ODE flows and volume constraints in shape optimization

Amaury Bélières–Frendo[1], Charles Dapogny[2],
Victor Michel-Dansac[3], Yannick Privat[4,5]

[1] Institut de Recherche Mathématique Avancée, Université de Strasbourg, Strasbourg, France
[2] Laboratoire Jean Kuntzmann, CNRS, Grenoble, France
[3] MACARON project-team, Université de Strasbourg, CNRS, Inria, IRMA, Strasbourg, France
[4] Institut Élie Cartan de Lorraine, École des Mines de Nancy, Nancy, France
[5] Institut Universitaire de France

6th May, 2025 - ANR Stoiques - Autrans

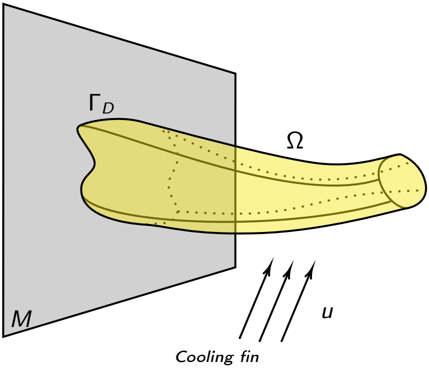
1. Foreword
   - Constraints in shape optimization
   - Intrinsic volume preservation

2. Mathematical framework for volume-preserving shape-optimization
   - Background on shape derivatives
   - Volume-preserving ODE flows
   - Reformulation of the shape optimization problem
   - The special case of dimension 2 and symplectic geometry

3. Neural networks can solve shape optimization problems
   - Neural symplectic maps
   - PDE resolution
   - Combinaison of PINNs and SympNets

Constraints in shape optimization

- Objectives of the PhD: neural methods for reach constrained shape optimization problems
- Could neural networks solve turbulent Navier-Stokes shape optimization problems?
- Existence and regularity of optimal shapes depend on the the admissible set, i.e. on the constraints of the problem.
- Numerical methods are designed to enforce these constraints.



Setting and objectives (I)

We aim to solve the following volume-constrained shape optimization problem

$$\inf_{\substack{\Omega \in \mathscr{O}_{\mathrm{ad}} \\ |\Omega| \leq V_0}} \mathscr{J}(\Omega),$$

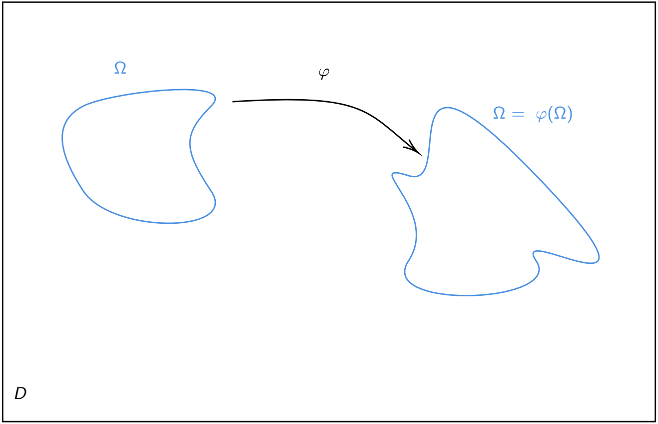
with $\Omega \subset D$ a shape in $\mathscr{O}_{\mathrm{ad}}$, an admissible space to be specified, $V_0 \in \mathbb{R}_+^*$, and $\mathscr{J}$ a shape functional defined by

$$\mathscr{J}(\Omega) = \int_\Omega j(u_\Omega)\, \mathrm{dx},$$

where j is regular, and u_Ω is the solution of the Poisson problem, with $f \in L^2(\Omega)$

$$\begin{cases} -\Delta u_\Omega = f & \text{in } \Omega; \\ u_\Omega = 0 & \text{on } \partial\Omega. \end{cases}$$

Setting and objectives (II)



We want to build a volume-preserving mapping φ that sends a given shape onto the optimal one.

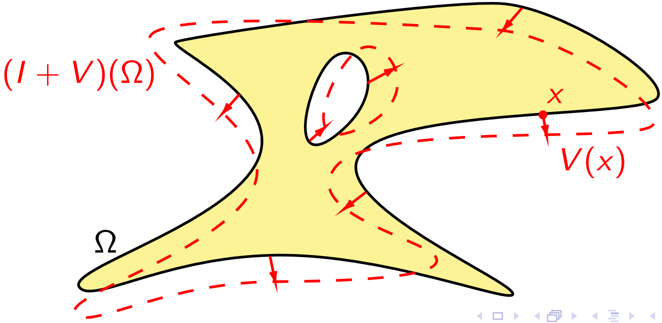
Setting and objectives (III)

- Build a parametrization of the mapping φ that preserves the volume of the shape.
- Compute the shape-derivative of the objective function with respect to these parameters introduced before.

Definition 1 (Shape derivative in the sense of Hadamard).

One function $\mathscr{J}(\Omega)$ of the domain is said to be shape differentiable at Ω if the underlying mapping $V \mapsto \mathscr{J}((\mathrm{I}+V)(\Omega))$, from $W^{1,\infty}(\mathbb{R}^d,\mathbb{R}^d)$ into $\mathbb{R}$ is Fréchet differentiable at $V=0$. The corresponding Fréchet differential is denoted by $\langle \mathrm{d}\mathscr{J}(\Omega), V\rangle$ and the following expansion holds

$$\mathscr{J}((\mathrm{I}+V)(\Omega)) = \mathscr{J}(\Omega) + \langle \mathrm{d}\mathscr{J}(\Omega), V\rangle + \mathrm{o}(||V||_{B^{1,\infty}(0,1)}).$$



Theorem 1 (Hadamard shape derivative).

Let $\mathscr{J}$ be a shape functional defined by

$$\mathscr{J}(\Omega) = \int_{\Omega} j(u_\Omega)\, \mathrm{dx},$$

where j is regular, and u_Ω is the solution of the Poisson problem, with $f \in L^2(\Omega)$

$$\begin{cases} -\Delta u_\Omega = f & \text{in } \Omega; \\ u_\Omega = 0 & \text{on } \partial\Omega. \end{cases}$$

Thus

$$\langle \mathrm{d}\mathscr{J}(\Omega),\ V \rangle = \int_{\partial\Omega} v_\Omega\, V \cdot n \, \mathrm{d}\sigma,$$

with $v_\Omega := (j(u_\Omega) - \nabla u_\Omega \cdot \nabla p_\Omega)$ where p_Ω solves

$$\begin{cases} -\Delta p_\Omega = -j'(u_\Omega) & \text{in } \Omega; \\ p_\Omega = 0 & \text{on } \partial\Omega, \end{cases}$$

p_Ω is called adjoint state.

Volume-preserving ODE flows

The flow associated with a divergence-free vector field V in $D \subset \mathbb{R}^n$ is defined by the ODE

$$\begin{cases} \partial_t \varphi(t,x) = V \circ \varphi(t,x) & \forall (t,x) \in [0,1] \times D; \\ \varphi(0,x) = x & \forall x \in D, \end{cases}$$

and φ is volume-preserving, i.e. $\forall t, \Omega,\ |\Omega| = |\varphi(t,\Omega)|$.

Helmoltz decomposition of divergence-free vector fields

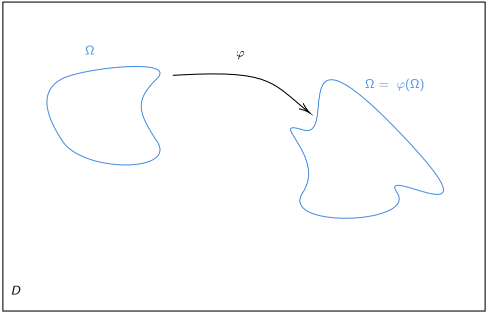
Any vector field $V \in L^2(D)^d$ can be decomposed in the following way

$$V = \mathrm{curl}\phi + \nabla z.$$

If $V \in L^2(D)^d$ is assumed to be divergence-free, z solves

$$\begin{cases} -\Delta z = 0 & \text{in } D; \\ z = g & \text{on } \partial D, \end{cases}$$

with $g \in H^{1/2}(D)$.



We call $\varphi_{phi,g}$ the solution of

$$\begin{cases} \partial_t \varphi_{\phi,g}(t,x) = (\mathrm{curl}\phi + \nabla z(g)) \circ \varphi_{\phi,g}(t,x) & \forall (t,x) \in [0,1] \times D; \\ \varphi_{\phi,g}(0,x) = x & \forall x \in D, \end{cases}$$

A reformulated optimization problem

The shape functional under consideration is

$$\mathscr{J}(\Omega) = \int_{\Omega} j(u_{\Omega})\,\mathrm{dx},$$

To compute a gradient descent on the parameters of the volume preserving map, we introduce the new shape functional

$$\widehat{\mathscr{J}}(\phi,\, g) := \mathscr{J}(\varphi_{\phi,g}(1,\, \Omega)) = \int_{\Omega_{\phi,g}} j(u_{\phi,g})\,\mathrm{dx},$$

with $\Omega_{\phi,g} = \varphi_{\phi,g}(1,\Omega)$ and $u_{\phi,g} \in H_0^1(\Omega_{\phi,g})$ solution of

$$\begin{cases} -\Delta u_{\phi,g} = f & \text{in } \Omega_{\phi,g}; \\ u_{\phi,g} = 0 & \text{on } \partial\Omega_{\phi,g}. \end{cases}$$

Remark: by searching the minimum of $\widehat{\mathscr{J}}$, we are now solving a constraint-free optimization problem. We now have to compute the differential of $\widehat{\mathscr{J}}$ with respect to ϕ and g.

Computation of the functionnal derivative (I)

As $\varphi_{\phi,g}$ replaces $\mathrm{I}+V$ in the shape derivative in the sens of Hadamard, we use the chain rule applied in $(\phi,\, g)=0$, in the direction $(\widehat{\phi},\, \widehat{g})$ in the Hadamard formulae

$$\widehat{\mathscr{J}}(\phi,\, g) = \int_\Omega j(u_{\phi,g})\,\mathrm{dx},$$

becomes

$$\langle \mathrm{d}\widehat{\mathscr{J}}(0),\, \widehat{\phi}\rangle = \int_{\partial\Omega} v_\Omega\, \mathrm{curl}\widehat{\phi}\cdot n\,\mathrm{d}\sigma,$$

and

$$\langle \mathrm{d}\widehat{\mathscr{J}}(0),\, \widehat{g}\rangle = \int_{\partial\Omega} v_\Omega\, \nabla z(\widehat{g})\cdot n\,\mathrm{d}\sigma.$$

With $v_\Omega = j(u_\Omega) - \nabla u_\Omega \cdot \nabla p_\Omega$.

We want to express the shape derivatives as a scalar product exploitable to compute a gradient descent.

Computation of the functionnal derivative (II)

After some computations, the shape derivatives are given by

$$\left\langle \frac{\partial \widehat{\mathscr{J}}}{\partial g}(0,0), \widehat{g} \right\rangle = \left\langle \frac{\partial y_\Omega}{\partial n}, \widehat{g} \right\rangle_{L^2(\partial D)},$$

with $y_\Omega \in H_0^1(\Omega)$, such that for all $w \in H_0^1(\Omega)$,

$$\int_D \nabla y_\Omega \cdot \nabla w \, \mathrm{dx} = \int_D \nabla w \cdot \nabla q_\Omega \, \mathrm{dx},$$

and $q_\Omega \in H^1(D)$ the orthogonal projection of $v_\Omega \in H^1(\Omega)$ on $H^1(D)$, such that for all $w \in H^1(D)$,

$$\int_D \nabla q_\Omega \cdot \nabla w \, \mathrm{dx} = -\int_\Omega \nabla v_\Omega \cdot \nabla w \, \mathrm{dx}.$$

Computation of the functionnal derivative (III)

$$\left\langle \frac{\partial \widehat{\mathscr{J}}}{\partial \phi}(0,0),\ \widehat{\phi} \right\rangle = \left\langle \mathrm{curl}\xi_\Omega,\ \widehat{\phi} \right\rangle_{L^2(\partial D)},$$

with $\xi_\Omega \in H^1(D)$, for all $w \in H^1(D)$,

$$\langle \xi_\Omega,\ w \rangle_{H^1(D)} = -\int_{\partial\Omega} v_\Omega \langle \mathrm{curl} w,\ n \rangle \,\mathrm{dx}.$$

Algorithm

- **Solve the partial differential equations (PDEs):**
  - Solve the following PDEs on the domain Ω: $u_\Omega, p_\Omega, q_\Omega, y_\Omega, \xi_\Omega$.
- **Construct the divergence-free vector field:**
  - Compute $\nabla z(g)$ and $\mathrm{curl}\, \xi_\Omega$.
  - Combine these quantities to form a divergence-free vector field.
- **Compute the associated flow (ordinary differential equation):**
  - Integrate the flow $\varphi(t, x)$ defined by the previous vector field, for $t \in [0, 1]$, with the initial condition $\varphi(0, x) = x$.
- **Domain deformation:**
  - Transport the domain Ω by the obtained flow: $\Omega_{\text{final}} := \varphi(1, \Omega)$.

Remark: all the equations can be solved using any numerical method (FEM, NNs). The shape can be represented either by a mesh or a level set function. This is not an issue, as for now it remains an abstract method that can be implemented with any numerical technique.

plots

lien vidéo

1. Foreword
   - Constraints in shape optimization
   - Intrinsic volume preservation

2. Mathematical framework for volume-preserving shape-optimization
   - Background on shape derivatives
   - Volume-preserving ODE flows
   - Reformulation of the shape optimization problem
   - The special case of dimension 2 and symplectic geometry

3. Neural networks can solve shape optimization problems
   - Neural symplectic maps
   - PDE resolution
   - Combinaison of PINNs and SympNets

Hamiltonian ODEs

$$\partial_t \phi(t, (x, p)) = J \nabla H(\phi)(t, (x, p))$$

- (x, p) belongs to the phase space
- x can represent the position of the system, while p can represent the momentum of the system
- $\dim(x) = \dim(p) := d$
- H is the Hamiltonian function, i.e. the energy of the system
- J is the symplectic form in the canonical bases of $\mathbb{R}^{2d}$, i.e.

$$J = \begin{pmatrix} 0 & I_d \\ -I_d & 0 \end{pmatrix}$$

- J is in $\mathbb{R}^2$ the $-\pi/2$ rotation matrix

How to adapt Hamiltonian mechanics to our problem?

- Hamiltonian potential physically reprents an energy that is conserved in time.
- The flow associated with the Hamiltonian ODE, called symplectic maps, preserves the volume of the phase space.
- We want to make it preserve the volume of our shape, because symplectic maps have a lot of structure properties that we can leverage to introduce a smarter parametrization of the volume-presereving maps of even dimensions.

Some useful properties [Arnold]

Shear maps

Any symplectic map in $C^1(\mathbb{R}^{2d})$ can be approximated by the composition of several shear maps, defined as follows, with $x = (x_1, x_2) \in \mathbb{R}^{2d}$

$$f_{\mathrm{up}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 + \nabla V_{\mathrm{up}}(x_2) \\ x_2 \end{pmatrix} \quad \text{and} \quad f_{\mathrm{down}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 + \nabla V_{\mathrm{down}}(x_1) \end{pmatrix},$$

where $V_{\mathrm{up/down}} \in C^2(\mathbb{R}^d, \mathbb{R})$, and $\nabla V : \mathbb{R}^d \to \mathbb{R}^d$ is the gradient of V.

How could we parametrize efficiently the shear maps?

SympNets [Jin et al, 2020]

Theorem

Let $q > 0$ be the depth of the neural network. In practice, we set $q > 2n$. We define the approximation $\widehat{\sigma_{K,a,b}}$ of ∇V by an activation function $\sigma : \mathbb{R} \to \mathbb{R}$, two vectors $a, b \in \mathbb{R}^q$, a matrix $K \in \mathcal{M}_{q,n}(\mathbb{R})$, and $\mathrm{diag}(a) = (a_i \delta_{ij})_{1 \leq i,j \leq q}$, as follows

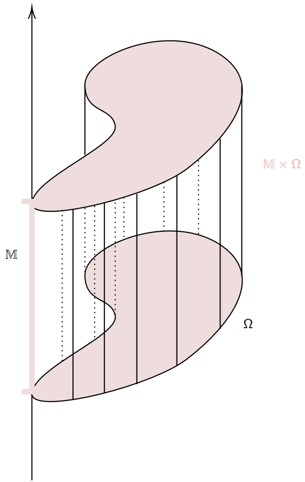
$$\widehat{\sigma_{K,a,b}}(x) = K^\intercal \mathrm{diag}(a) \sigma(Kx + b),$$

Then, gradient modules $\mathcal{G}_{\text{up}}$ and $\mathcal{G}_{\text{down}}$ are defined to approximate f_{up} and f_{down}, by

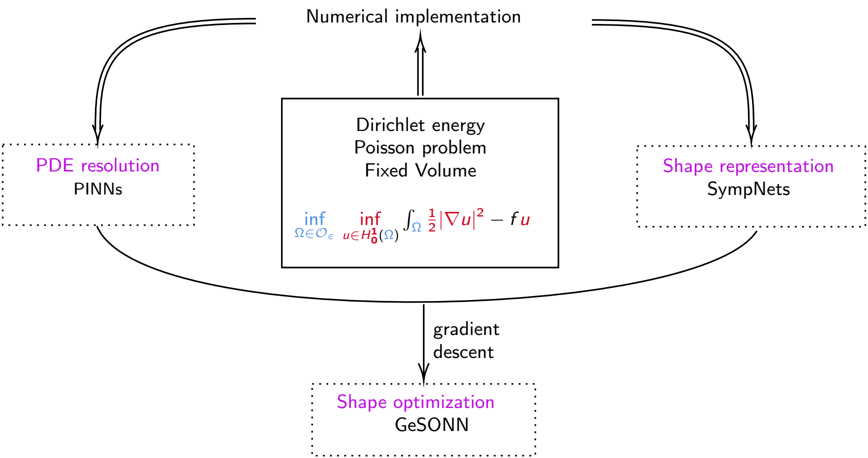
$$\mathcal{G}_{\text{up}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 + \widehat{\sigma_{K,a,b}}(x_2) \\ x_2 \end{pmatrix} \quad \text{and} \quad \mathcal{G}_{\text{down}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 + \widehat{\sigma_{K,a,b}}(x_1) \end{pmatrix}.$$

Parametric problems

- Examples: solving a parameter dependant PDE for a wide range of parameters
  - Stokes equation for different viscosity coefficients,
  - elasticity equation for different elasticity or shear moduli or Poisson coefficients.
- The dimension of the approximation space increases with the dimension of the parameter space.
- Accuracy of the Monte-Carlo integral only depends on the collocation points which will be in the cross space of the domain and of the parameter space.



Objective


Numerical implementation
PDE resolution
PINNs
Dirichlet energy
Poisson problem
Fixed Volume
$\inf_{\Omega \in \mathcal{O}_\varepsilon} \inf_{u \in H_0^1(\Omega)} \int_\Omega \frac{1}{2} |\nabla u|^2 - f u$
Shape representation
SympNets
gradient
descent
Shape optimization
GeSONN

Physics-informed neural networks (PINNs) (I)

Paramteric function: $\theta = \left\{(W^k,\, b^k)\right\}_{k=0}^{l}$ designates the set of parameters to be trained by the neural network, which propagates the input data through its l different layers, according to the sequence of operations:

$$\begin{cases} z^0 & = x, \\ z^k & = \sigma(W^k z^{k-1} + b^k),\ 1 \leq k \leq l, \\ z^l & = W^l z^{l-1} + b^l. \end{cases}$$

Each layer has as an ouput a vector $z_k \in \mathbb{R}^{q_k}$, where q_k is the number of "neurons", and is defined by a weights matrix $W^k \in \mathbb{R}^{q_k} \times \mathbb{R}^{q_k - 1}$, a bias vector $b \in \mathbb{R}^{q_k}$ and a non-linear activation function $\sigma(.)$.
Activation functions: hyperbolic tangent, sigmoid, relu...

PINNS (II)

Find an approximation of u_Ω the solution of the Poisson problem

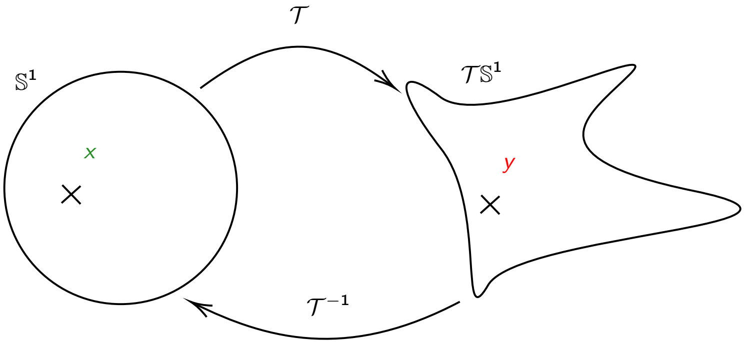
$$(\mathcal{P}) : \begin{cases} -\Delta u_\Omega = f & \text{in } \Omega; \\ u_\Omega = 0 & \text{on } \partial\Omega. \end{cases}$$

Our problem can be directly seen as an optimization problem

$$\inf_{v \in H_0^1(\Omega)} \frac{1}{2} \int_\Omega |\nabla v|^2 \, \mathrm{dx} - \int_\Omega f v \, \mathrm{dx}.$$

Joint formulation of the Dirichlet energy (I)

Here is a domain generated by a symplectic mapping of the unit disk of $\mathbb{R}^2$.



$$\begin{aligned} -\Delta u_{\mathcal{T}}(y) &= f(y), \quad y \in \mathcal{T}\mathbb{S}^1 \\ u_{\mathcal{T}}(y) &= 0, \qquad y \in \partial\mathcal{T}\mathbb{S}^1 \end{aligned}$$

we introduce $w : \mathbb{S}^1 \to \mathbb{R}$, defined for a.e. $x \in \Omega$ by

$$w(x) = (u_{\mathcal{T}} \circ \mathcal{T})(x).$$

Joint formulation of the Dirichlet energy (II)

PDE

$$\begin{cases} -\mathrm{div}\big(A\nabla w\big) = f \circ \mathcal{T}, & \text{in } \mathbb{S}^1; \\ w = 0, & \text{on } \mathbb{S}^1, \end{cases}$$

with $A : \mathbb{S}^1 \to \mathbb{R}$ a uniformly elliptic metric tensor, defined by

$$A = J_{\mathcal{T}}^{-1} \cdot J_{\mathcal{T}}^{-\intercal}.$$

Optimization problem

$$\inf \left\{ \frac{1}{2} \int_{\mathbb{S}^1} A\nabla w \cdot \nabla w - \int_{\mathbb{S}^1} \widetilde{f} w, \quad \exists u_{\mathcal{T}} \in H_0^1(\mathcal{T}\mathbb{S}^1), \quad w = u_{\mathcal{T}} \circ \mathcal{T} \in H_0^1(\mathbb{S}^1) \right\}$$

Joint formulation of the Dirichlet energy (III) [Bélières et al, 2025]

The Dirichlet energy as a loss function

$$\mathcal{J}_{P/S}\left(\theta, \omega; \{x_i\}^N\right) = \frac{V_0}{N} \sum_{i=1}^{N} \left\{ \frac{1}{2} \left| A_\omega \nabla v_{\theta,\omega} \cdot \nabla v_{\theta,\omega} \right|^2 - \widetilde{f_\omega} v_{\theta,\omega} \right\} (x_i)$$

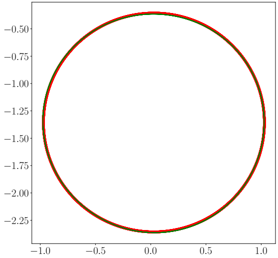
- θ the trainable weights of the PINN, ω the trainable weights of the SympNet;
- $v_{\theta,\omega} : \mathbb{S}^1 \to \mathbb{R}; x \mapsto \alpha(x) u_\theta(T_\omega x) + \beta(x)$ a test function;
- $T_\omega : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ the SympNet;
- $u_\theta : T_\omega \mathbb{S}^1 \to \mathbb{R}$ the PINN.

Algorithm

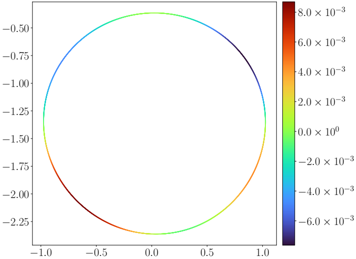
- **Problem setup:**
  - Consider a parameterized PDE defined on a reference domain $B^2 \times \mathcal{M}$.
  - Use a transformation T_ω to map to the physical domain and push the PDE accordingly.
- **Train a composite model (PINN + SympNet):**
  - For each training iteration:
    - Sample N collocation points (x_i, μ_i) in $B^2 \times \mathcal{M}$.
    - Compute the loss associated to the PDE residual at each collocation point
    - Update model parameters (θ, ω) via a gradient descent.

Remark: The method jointly optimizes the PDE solution and the transformation using PINNs and a SympNet. The PDE is solved only at the end of the training procedure. This method only applies for min min problems.

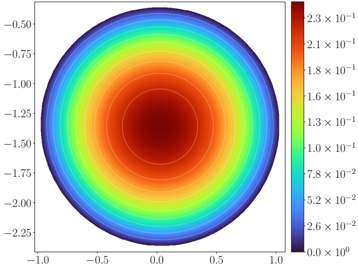
Source term $f = 1$



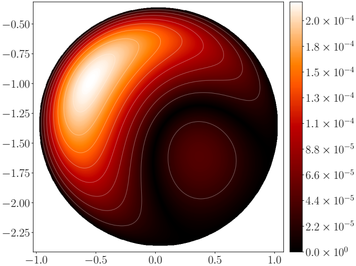
(a) Optimal shape



(b) optimality condition

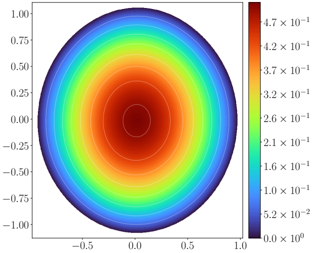


(c) PDE solution

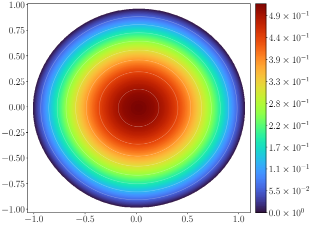


(d) point wise error of the PDE

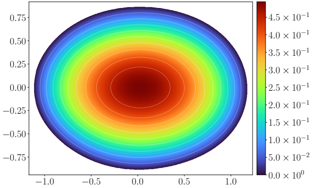
Parametric family of source terms, $f = \exp\left(1 - (\frac{x_1}{\mu})^2 - (\mu x_2)^2\right)$



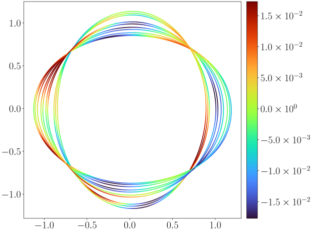
(a) solution, $\mu = 0.83$



(b) solution, $\mu = 1.13$



(c) solution, $\mu = 1.61$



(d) optimality condition

Conclusion

2 strategies

- Parametrize a divergence-free vector field.
- Parametrize a symplectic flow.

Thank you!

Thank you for your attention!