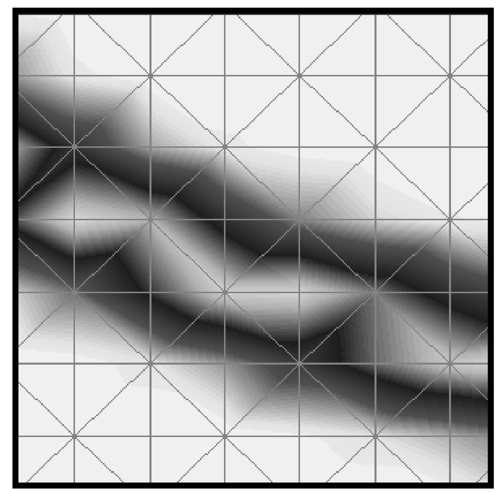
Towards a purely Eulerian method for shape optimization

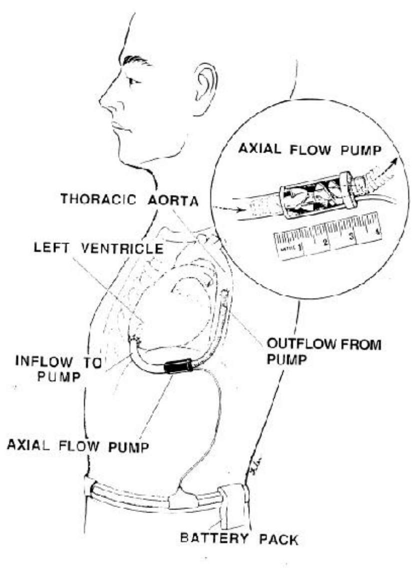
Volkan Akcelik, Alexandre Cunha, Omar Ghattas
Carnegie Mellon University

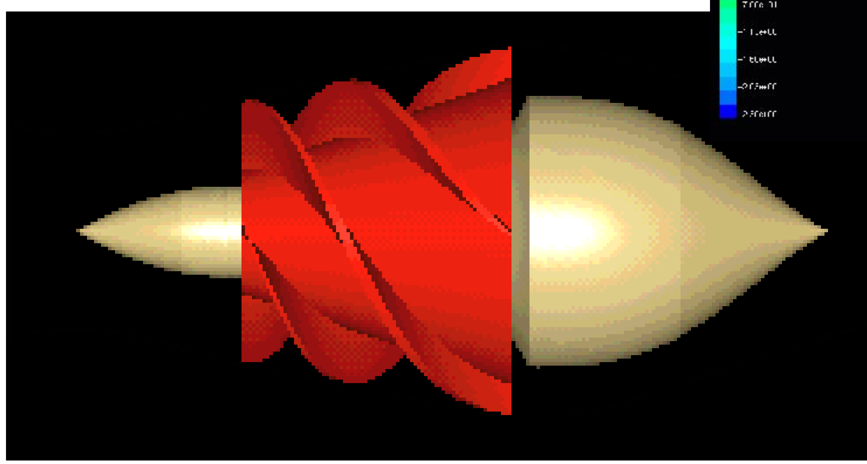
George Biros
University of Pennsylvania

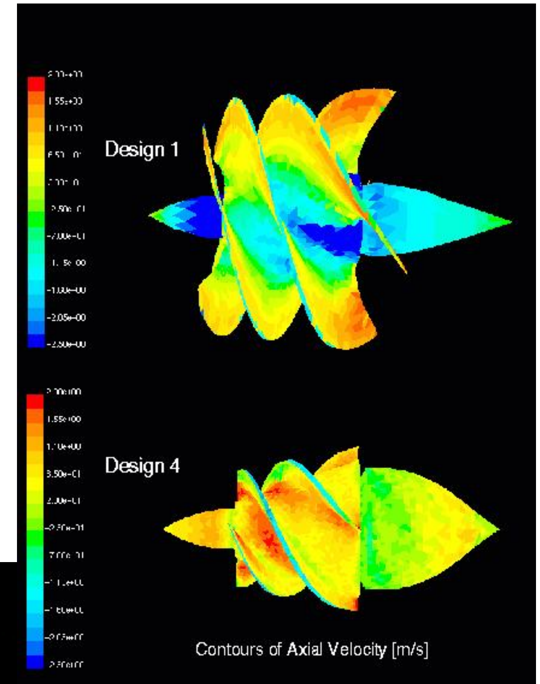


CFD example: shape optimization of artificial heart ventricular assist devices

- development of artificial heart device at UPMC by Jim Antaki, Brad Paden (UCSB), et al.
- numerous advantages (size, power, reliability, invasiveness)
- design challenge: overcome tendency to damage RBCs

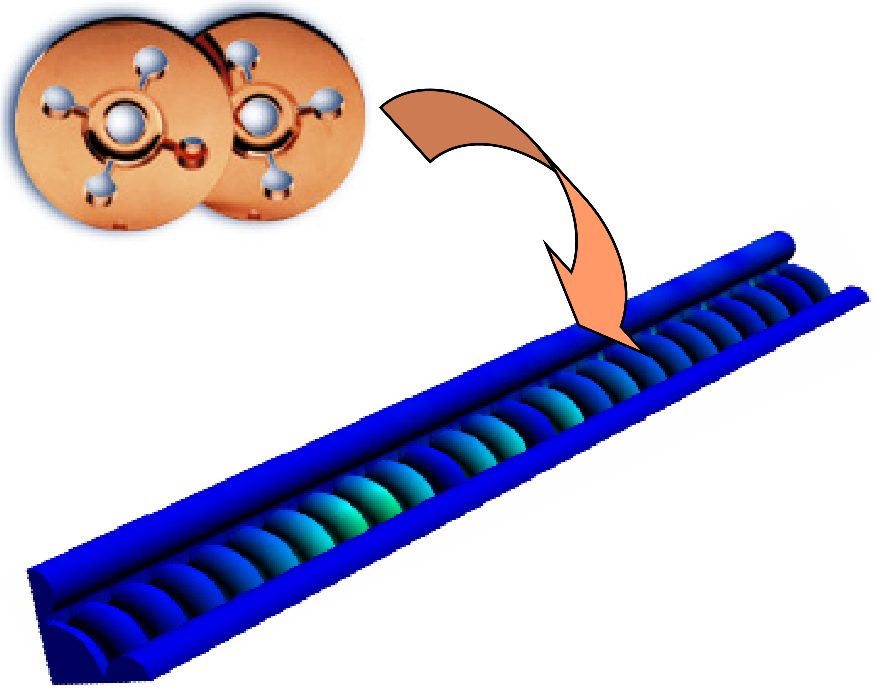




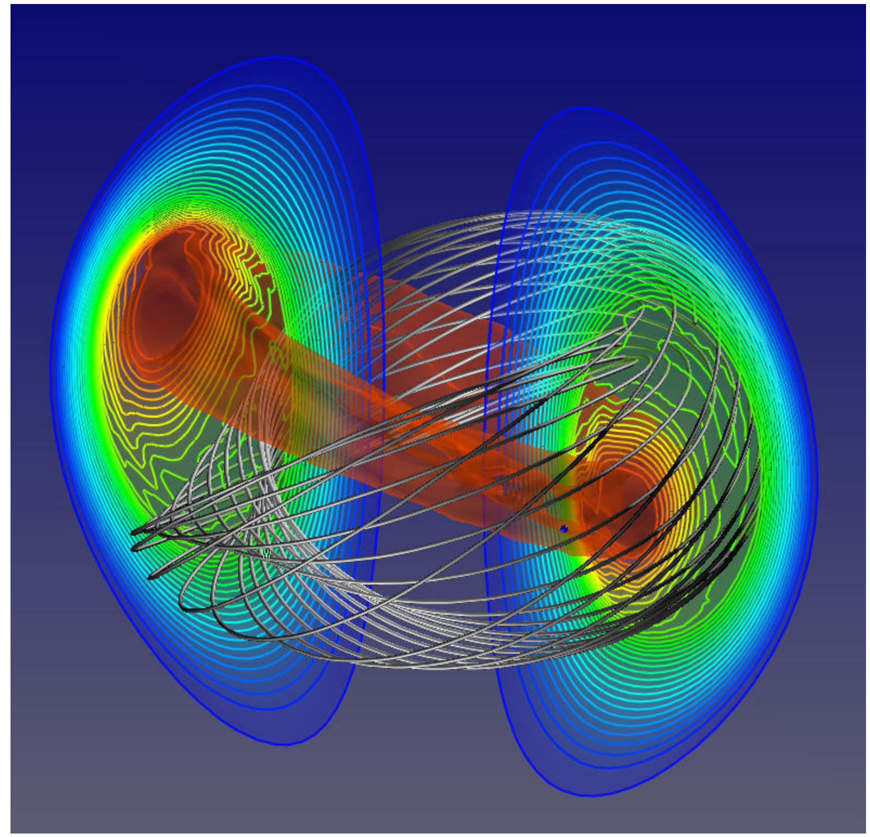


G Burgreen & JF Antaki, UPMC

EM/MHD example: shape optimization problems in SciDAC program



design of next linear collider
(K. Ko et al., Stanford)



design of plasma fusion device
(S. Jardin et al., Princeton)

outline

Overall goal: an efficient purely Eulerian method for shape optimization

- current approaches
- nonlinear solver: LNKS
- level set representation of shape
- shape matching problem
- distributed Lagrange multiplier fictitious domain treatment of boundary conditions
- displacement matching problem

discrete approach

continuous problem:

$$\begin{aligned} \text{minimize} \quad & \mathcal{F}(u,z) := \int \phi(u)\, d\Omega(z) \\ \text{subject to} \quad & \mathcal{L}(u) = f \text{ in } \Omega(z) \end{aligned}$$

discretized problem:

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{u},\boldsymbol{z}) \\ \text{subject to} \quad & \boldsymbol{c}(\boldsymbol{u},\boldsymbol{z}) = \boldsymbol{0} \end{aligned}$$

Lagrangian function:

$$L(\boldsymbol{u},\boldsymbol{z},\boldsymbol{\lambda}) := f(\boldsymbol{u},\boldsymbol{z}) + \boldsymbol{\lambda}^{\mathsf{T}} \boldsymbol{c}(\boldsymbol{u},\boldsymbol{z})$$

optimality conditions ($\boldsymbol{g} := \nabla f$; $\boldsymbol{A} := \nabla \boldsymbol{c}$):

$$\begin{aligned} \boldsymbol{c}(\boldsymbol{u},\boldsymbol{z}) &= \boldsymbol{0} && \text{state equation} \\ \boldsymbol{A}_u^{\mathsf{T}}(\boldsymbol{u},\boldsymbol{z})\boldsymbol{\lambda} &= -\boldsymbol{g}_u(\boldsymbol{u},\boldsymbol{z}) && \text{adjoint equation} \\ \boldsymbol{A}_z^{\mathsf{T}}(\boldsymbol{u},\boldsymbol{z})\boldsymbol{\lambda} &= -\boldsymbol{g}_z(\boldsymbol{u},\boldsymbol{z}) && \text{shape equation} \end{aligned}$$

Difficulties:
- Shape parameterization
- CAD sensitivity
- Mesh movement
- Mesh sensitivities

sidebar: Lagrange-Newton-Krylov-Schur nonlinear solver

optimality conditions:

$$
\begin{aligned}
\boldsymbol{c}(\boldsymbol{u},\boldsymbol{z}) &= \boldsymbol{0} && \text{state equation}\\
\boldsymbol{A}_u^{\mathsf{T}}(\boldsymbol{u},\boldsymbol{z})\boldsymbol{\lambda} &= -\boldsymbol{g}_u(\boldsymbol{u},\boldsymbol{z}) && \text{adjoint equation}\\
\boldsymbol{A}_z^{\mathsf{T}}(\boldsymbol{u},\boldsymbol{z})\boldsymbol{\lambda} &= -\boldsymbol{g}_z(\boldsymbol{u},\boldsymbol{z}) && \text{shape equation}
\end{aligned}
$$

nonlinear elimination

Newton step ($\boldsymbol{H} := \nabla^2 L$):

$$
\begin{bmatrix} \boldsymbol{H}_{uu} & \boldsymbol{H}_{uz} & \boldsymbol{A}_u^{\mathsf{T}} \\ \boldsymbol{H}_{zu} & \boldsymbol{H}_{zz} & \boldsymbol{A}_z^{\mathsf{T}} \\ \boldsymbol{A}_u & \boldsymbol{A}_z & 0 \end{bmatrix}
\begin{Bmatrix} \delta\boldsymbol{u} \\ \delta\boldsymbol{z} \\ \delta\boldsymbol{\lambda} \end{Bmatrix}
= -\begin{Bmatrix} \boldsymbol{A}_u^{\mathsf{T}}\boldsymbol{\lambda} + \boldsymbol{g}_u \\ \boldsymbol{A}_z^{\mathsf{T}}\boldsymbol{\lambda} + \boldsymbol{g}_z \\ \boldsymbol{c} \end{Bmatrix}
$$

block elimination

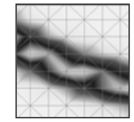
$\tilde{\boldsymbol{A}}_u$

Schur complement of shape block:

block preconditioning

$$
\boldsymbol{A}_z^{\mathsf{T}}\boldsymbol{A}_u^{-\mathsf{T}}\boldsymbol{H}_{uu}\boldsymbol{A}_u^{-1}\boldsymbol{A}_z - \boldsymbol{H}_{zu}\boldsymbol{A}_u^{-1}\boldsymbol{A}_z - \boldsymbol{A}_z^{\mathsf{T}}\boldsymbol{A}_u^{-\mathsf{T}}\boldsymbol{H}_{uz} + \boldsymbol{H}_{zz} \longleftarrow \boldsymbol{B}_z
$$

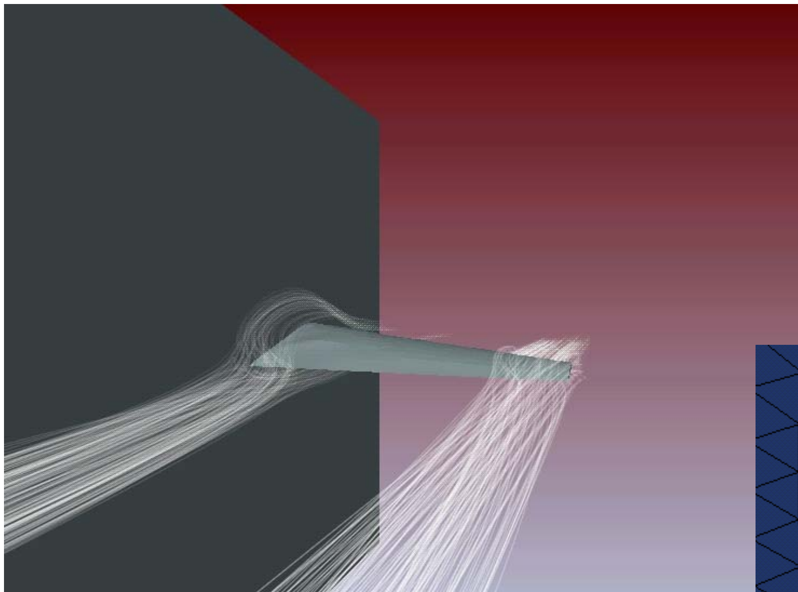
Lagrange-Newton-Krylov-Schur method

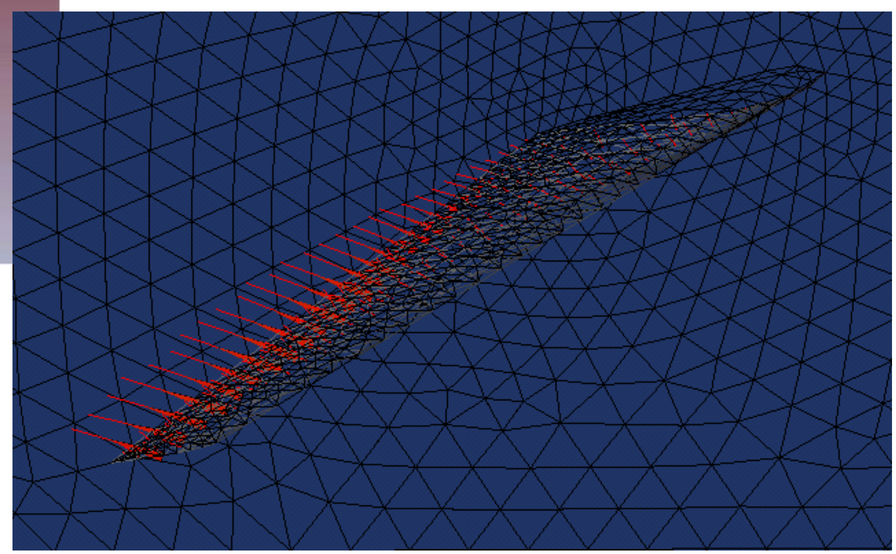


PETSc implementation: Veltisto

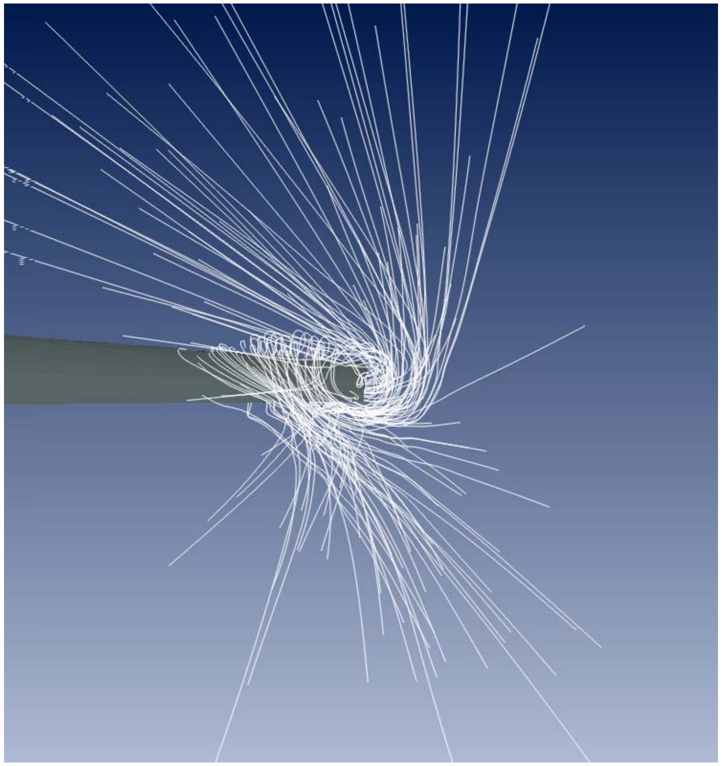
- Continuation loop
  - Optimization iteration (*Lagrange-Newton*)
    - Estimate extremal eigenvalues of (approximate) reduced Hessian using Lanczos (retreat to QN-RSQP if negative)
    - Inexact KKT solution via symmetric QMR (*Krylov*)
      - Quasi-Newton RSQP preconditioner (*Schur*)
        - » 2-step stationary iterations+ L-BFGS approximation of inverse reduced Hessian
        - » PDE solve replaced by PDE preconditioner
    - Backtracking line search on augmented Lagrangian or l_1 merit function
    - If no sufficient descent use QN-RSQP
    - Compute derivatives, objective function, and residuals
    - Update solution, tighten tolerances

optimal boundary flow control

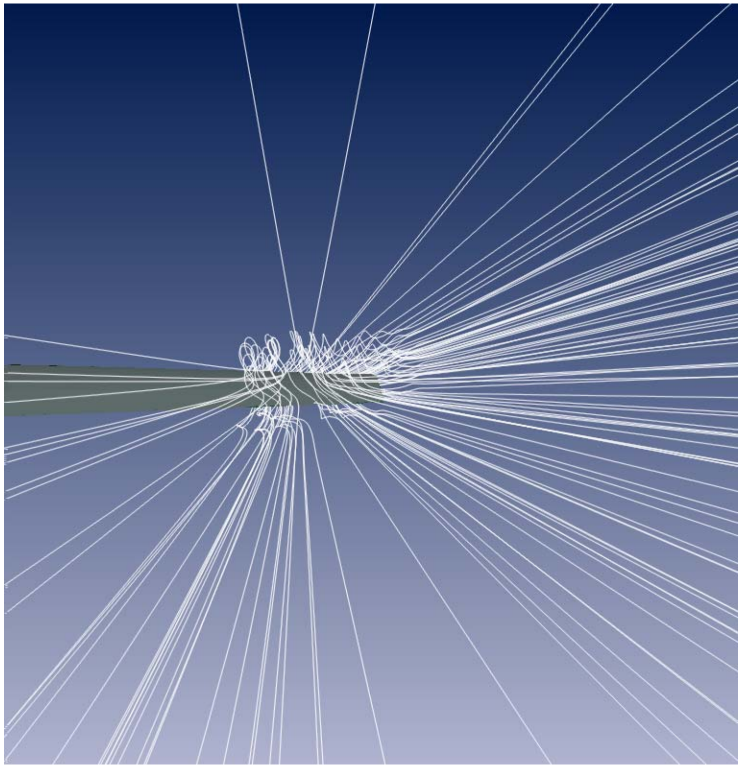




optimal boundary flow control

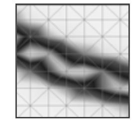


no control



optimal control

Isogranular algorithmic efficiency



Mesh independence of Krylov iteration with exact PDE solves (implies reduced Hessian preconditioner is effective)

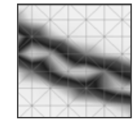
states controls	preconditioning	Newton iter	average KKT iter	time (hours)
117,048	QN-RSQP	161	—	32.1
2,925	LNKS-EX	5	18	22,8
(32 procs)	LNKS-PR	6	1,367	5,7
	LNKS-PR-TR	11	163	1.4
389,440	QN-RSQP	189	—	46.3
6,549	LNKS-EX	6	19	27.4
(64 procs)	LNKS-PR	6	2,153	15.7
	LNKS-PR-TR	13	238	3.8
615,981	QN-RSQP	204	—	53.1
8,901	LNKS-EX	7	20	33.8
(128 procs)	LNKS-PR	6	3,583	16.8
	LNKS-PR-TR	12	379	4.1

4x cost of PDE solve

"textbook" Newton mesh independence

Moderate growth of Krylov iterations with approximate PDE solves (implies PDE precond is moderately effective)

some continuous shape optimization approaches



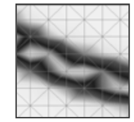
- “speed method” (Cea, Zolesio, ...)
  - \+ take “derivatives” at continuous level
  - \+ avoids mesh sensitivity
  - \+ high accuracy
  - – Newton methods are complicated
  - – Lagrangian method: still requires remeshing

Difficulties:
- ~~Shape parameterization~~
- ~~CAD sensitivity~~
- Mesh movement
- ~~Mesh sensitivities~~

- “level set” methods (Osher & Santosa, ...)
  - \+ level set description of shape (avoid parameterizing shape)
  - \+ has robustifying properties
  - – “controlled evolution” via Hamilton-Jacobi is too slow, equivalent to steepest descent
  - – still need to remesh for boundary shape optimization

$$\boxed{\frac{\partial \phi}{\partial t} + v \cdot \nabla \phi = 0 \text{ in } \Omega}$$

a purely Eulerian optimization approach



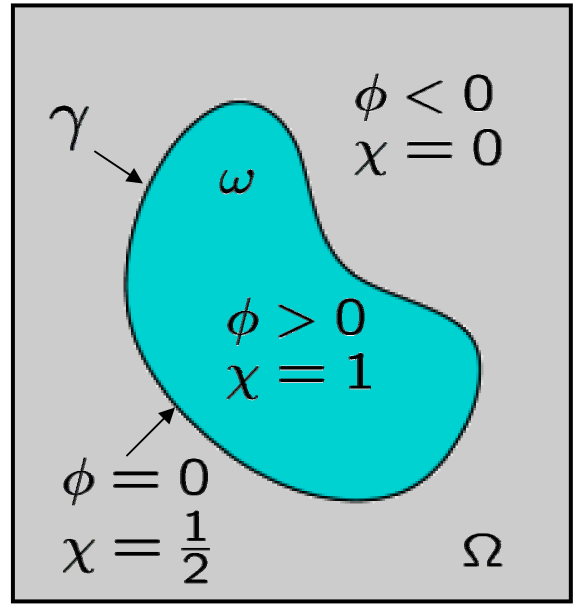
+ employ level set description of shape (need regularization)
+ solve via Newton's method (robustify conventionally)
+ employ fictitious domain method to avoid remeshing
- lower accuracy (but fast solvers possible on regular grids)

level set function $\phi(x)$ and heaviside function $\chi(\phi)$

$\phi = 0$ defines boundary

$$\chi(\phi) = \begin{cases} 1 & \text{if } \phi > 0 \\ \frac{1}{2} & \text{if } \phi = 0 \\ 0 & \text{if } \phi < 0 \end{cases}$$

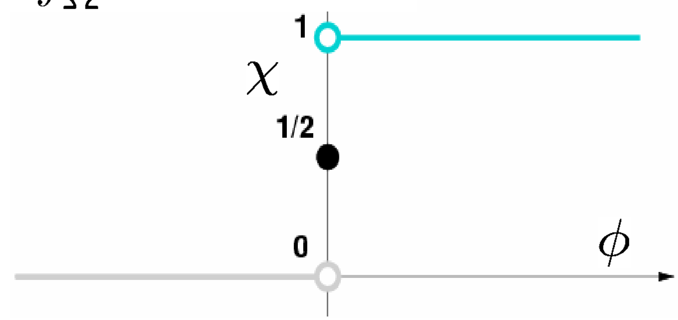
$$\text{Area}(\omega) = \int_\omega d\omega = \int_\Omega \chi(\phi)\, d\Omega$$



Γ

$$\text{Length}(\gamma) = \int_\omega d\gamma = \int_\Omega |\nabla \chi(\phi)|\, d\Omega = \int_\Omega \delta(\phi) |\nabla \phi|\, d\Omega$$

where δ is the delta function

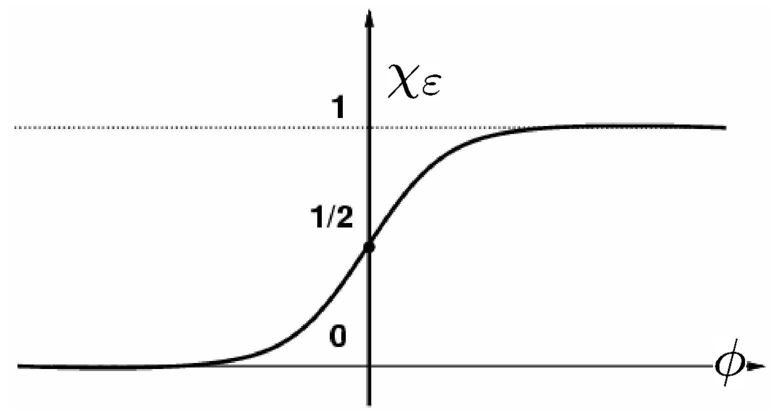


heaviside approximation

- smoothed approximation of heaviside function:

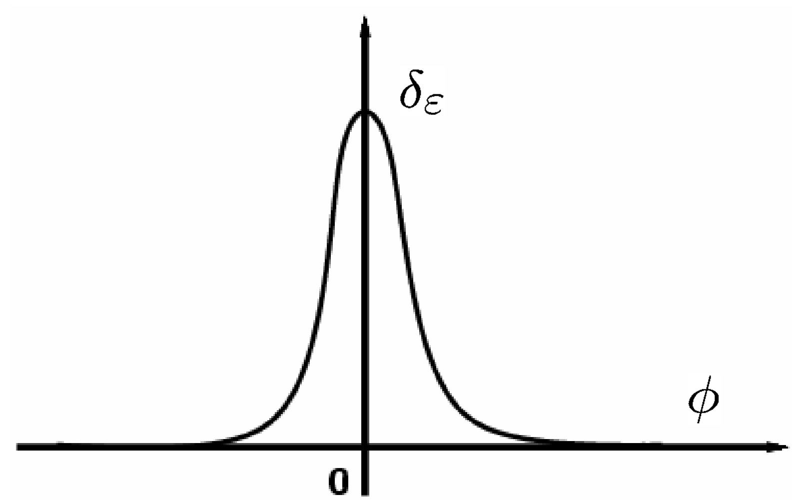
$$\chi_\varepsilon(\phi) = \frac{1}{1 + e^{-\phi/\varepsilon}}$$

$$\chi(\phi) = \lim_{\varepsilon \to 0} \chi_\varepsilon(\phi)$$



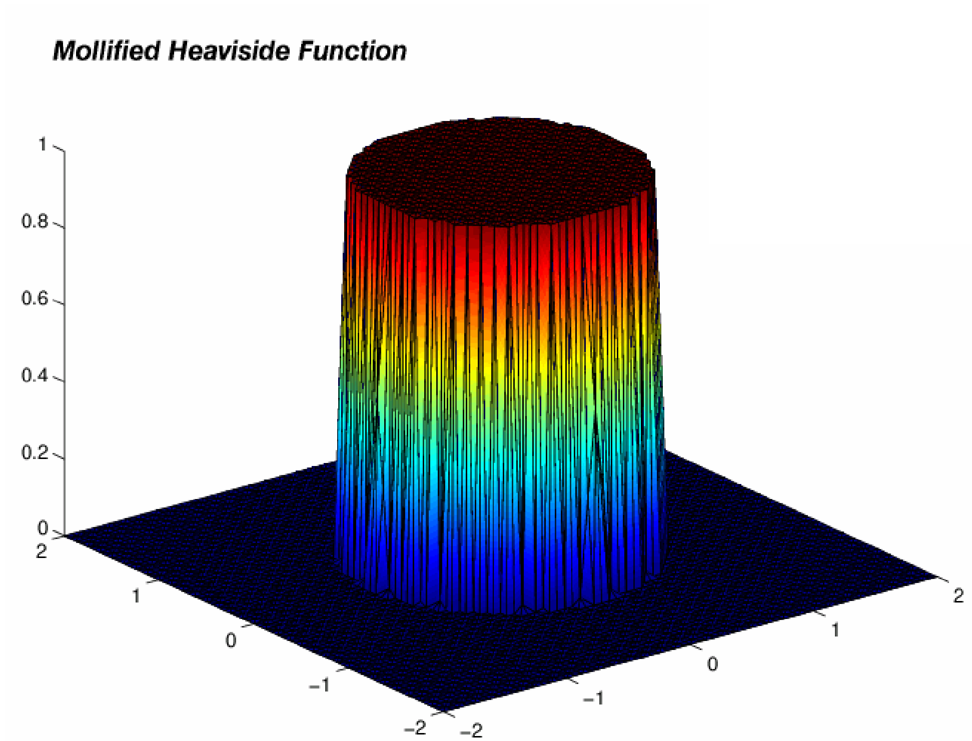
$$\delta_\varepsilon(\phi) = \frac{1}{\varepsilon}(1 - \chi_\varepsilon(\phi))\chi_\varepsilon(\phi)$$

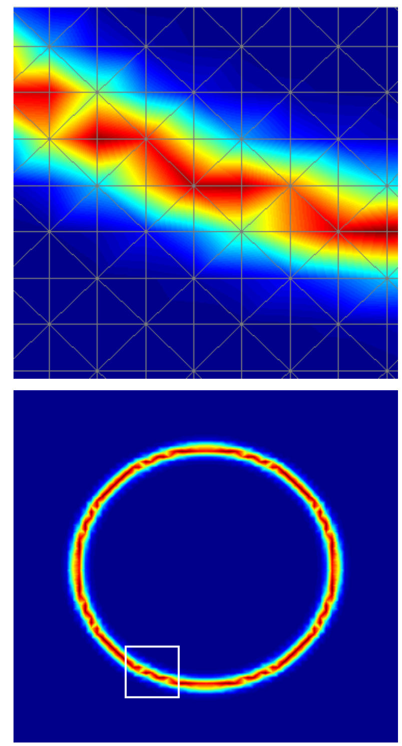
$$\delta(\phi) = \lim_{\varepsilon \to 0} \delta_\varepsilon(\phi)$$



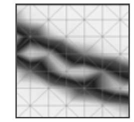
delta function approximation

- we chose ε so that the width of the delta function covers a multiple of the mesh size (typically 3h)



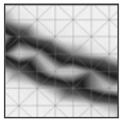


Sundance



- Implementation with Sundance library for PDE solution (Kevin Long, Sandia)
- powerful capabilities:
  - symbolic interface via variational forms
  - built in mesh generation
  - implicit geometric modeling via functional expressions
  - use of Trilinos linear solvers
  - rapid prototyping

implicit modeling in Sundance

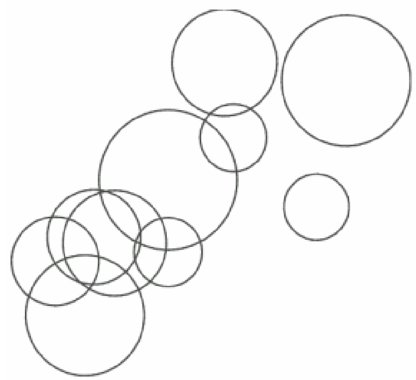


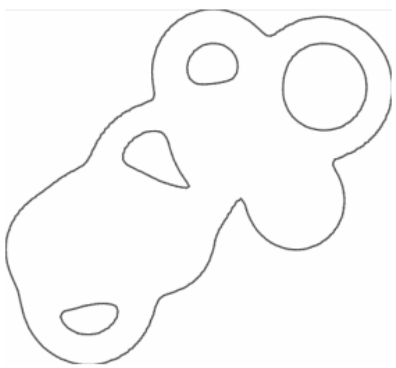
- Sundance's symbolic engine facilitates building geometric models as implicit algebraic functions
- if F and G are any two level set functions, both instances of the Expr (expression) class of Sundance, we can easily perform CSG boolean operations on them to build a more complex model:

$$F \cap G = F^n + G^n$$
$$F \cup G = (F^{-n} + G^{-n})^{-1}$$
$$F - G = F^n + G^{-n}$$

$$F = F(x, y)$$
$$G = G(x, y)$$
$$n = \text{blending factor}$$

blob model





circles as geometric primitives:

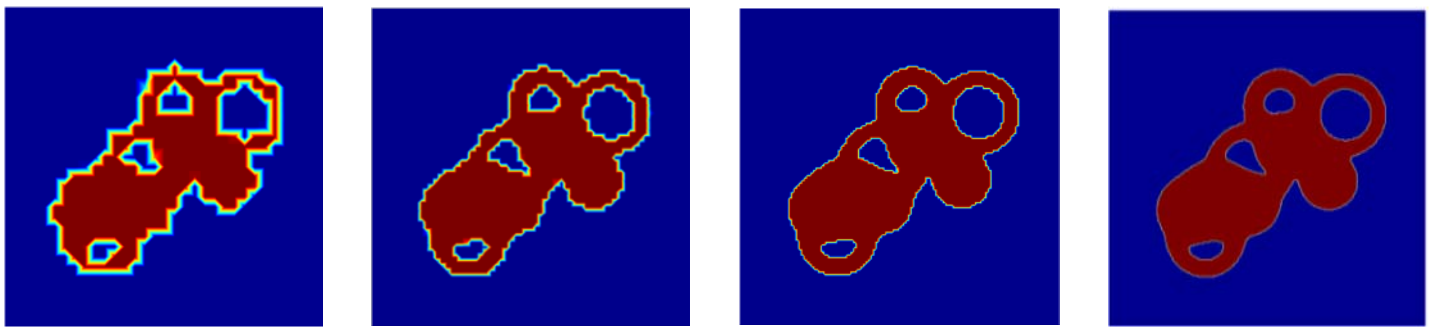
$$f_i = r_i^2 - (x - x_i)^2 - (y - y_i)^2$$

blob model: $$f = 2 - \frac{1}{\{\sum_i \frac{1}{f_i^n}\}^{1/n}}$$

```
case BLOOB:
  shape = 0.0;
  for (int i = 0; i < num_bloobs; i++)
    shape += 1.0/pow(rj[i]*rj[i] - pow(x - xj[i],2) - pow(y - yj[i],2),blend);
  shape = cutoff - 1.0/pow(shape, 1.0/blend);
  break;
```

area computation - blob

1/h= 32 64 128 256

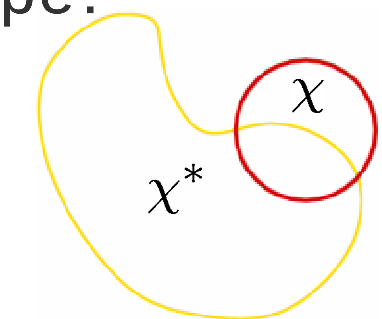


a shape matching problem

least squares problem for a target shape:

$$\min_{\phi} \mathcal{J}(\phi) := \frac{1}{2}\int_{\Omega} (\chi_{\varepsilon}(\phi) - \chi^*)^2$$

where χ^* defines the target shape



ill-posed problem: $\nabla\phi$ arbitrary for a given $\chi(\phi)$

$$\min_{\phi} \mathcal{J}(\phi) := \frac{1}{2}\int_{\Omega} (\chi_{\varepsilon}(\phi) - \chi^*)^2 + \mathcal{R}(\phi)$$

One choice is to penalize against a signed distance function:

$$\mathcal{R}(\phi) = \frac{\beta}{4}\int_{\Omega}(|\nabla\phi|^2 - 1)^2 + \frac{\varepsilon}{2}\int_{\Omega}|\nabla\phi|^2$$

optimality conditions

first order optimality condition

find $\phi \in H^1$ s.t.

$$\int_\Omega (\chi_\varepsilon(\phi) - \chi^*)\delta_\varepsilon(\phi)\hat{\phi} \; + (\beta(|\nabla\phi|^2 - 1) + \varepsilon)\nabla\phi \cdot \nabla\hat{\phi} \; = 0, \; \forall \hat{\phi} \in H^1$$

in strong form:

$$-\nabla \cdot \left[(\beta(|\nabla\phi|^2 - 1) + \varepsilon)\nabla\phi\right] + (\chi_\varepsilon(\phi) - \chi^*)\delta_\varepsilon(\phi) = 0 \;\; x \in \Omega$$

second variation (Hessian for Newton’s method):

$$\partial^2_\phi \mathcal{L} = \int_\Omega [\delta_\varepsilon(\phi)^2 + (\chi_\varepsilon(\phi) - \chi^*)\delta'_\varepsilon(\phi)]\hat{\phi}\tilde{\phi} \quad +$$

$$\int_\Omega [(\beta(|\nabla\phi|^2 - 1) + \varepsilon)\nabla\hat{\phi} \cdot \nabla\tilde{\phi} + 2\beta(\nabla\phi \cdot \nabla\hat{\phi})(\nabla\phi \cdot \nabla\tilde{\phi})]$$

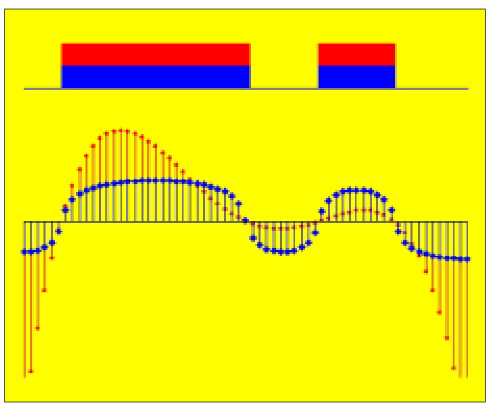
numerical examples

solution with Sundance:

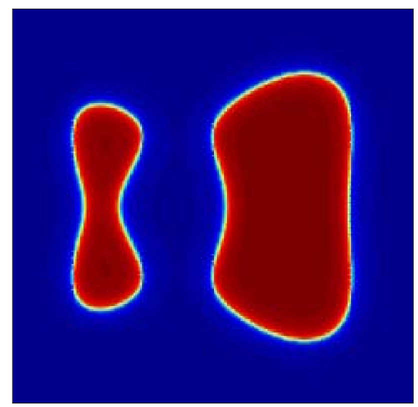
- FE approximation, linear triangles
- Gauss-Newton solver with backtracking line search
- BiCGSTAB Krylov solver with ILU(k) preconditioner

examples:

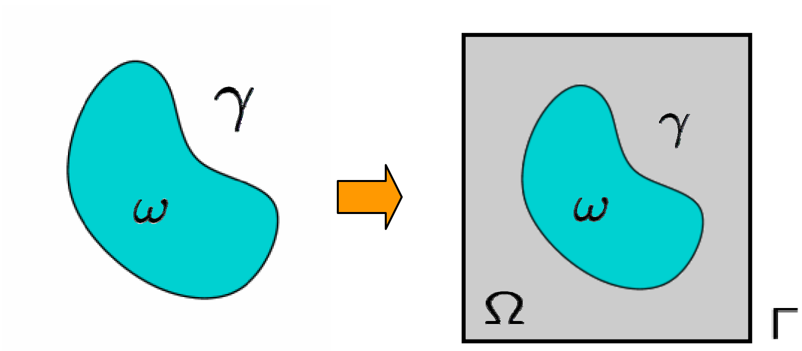
1D



2D



forward problem, penalized distributed fictitious domain method



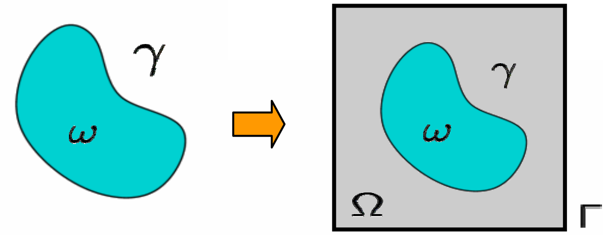
$$-\Delta u = f, \quad u \in \omega$$
$$u = 0, \quad u \in \gamma$$

$$-\Delta u_\tau + \frac{1}{\tau}(1-\chi(\phi))u_\tau = f, \quad u_\tau \in \Omega$$
$$u_\tau = 0, \quad u_\tau \in \Gamma$$

it can be shown that $u_\tau \to u$ as $\tau \to 0$ (Pironneau, 1984)

forward problem, penalized distributed fictitious domain method

$$-\Delta u = f, \quad u \in \omega$$
$$u = u_0, \quad u \in \gamma$$

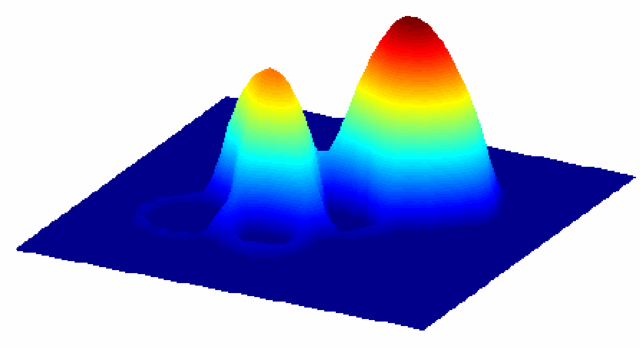


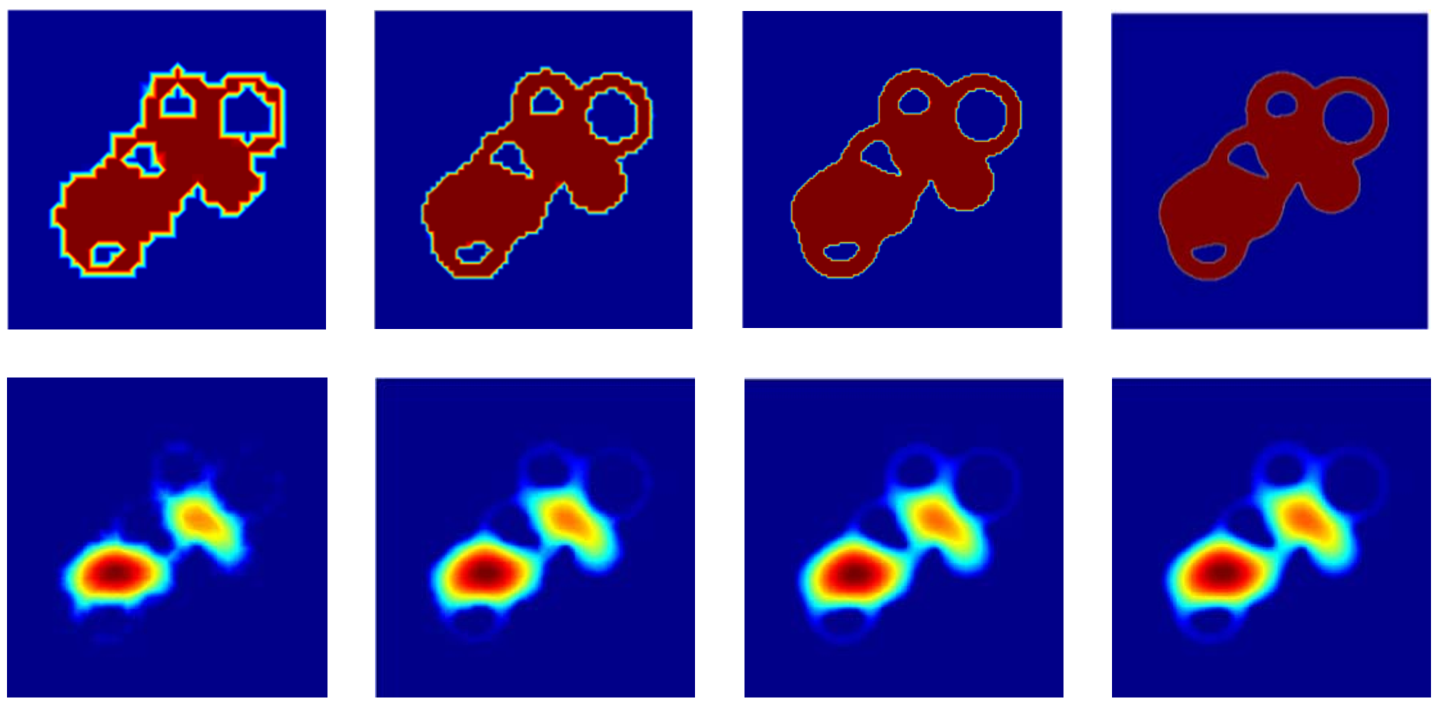
$$\min_u \int_\Omega \left(\frac{|\nabla u|^2}{2} - fu \right) \text{ s.t. } u = u_0, u \in \gamma$$

$$\min_u \int_\Omega \left(\frac{|\nabla u|^2}{2} - fu \right) + \frac{1}{2\tau} \int_{\Omega/\omega} (u - u_0)^2$$

$$\min_u \int_\Omega \left(\frac{|\nabla u|^2}{2} - fu \right) + \frac{1}{2\tau} \int_\Omega (1 - \chi(\omega))(u - u_0)^2$$

Poisson on the blob





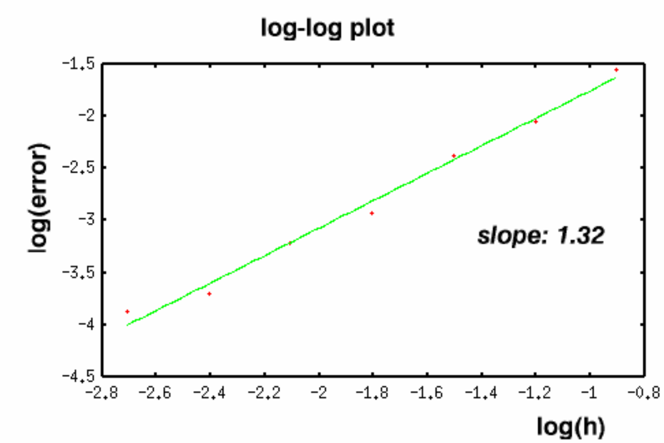
1/h: 32 64 128 256

forward problem, convergence rate

- Circle problem: convergence rate of 1.32 for linear triangles
- Rate is 2 (optimal) when γ is grid-aligned

Convergence analysis

1/h	beta	L2 error norm	log(1/h)	log(error)
8	+6.15642E+02	+2.79190547E-02	-0.903089986991943	-1.55409929040145
16	+1.56883E+03	+8.89118808E-03	-1.20411998265592	-2.0510402027966
32	+5.68378E+03	+4.25871730E-03	-1.50514997831991	-2.37072118808425
64	+2.89034E+04	+1.18665312E-03	-1.80617997398389	-2.92567621456378
128	+1.36059E+05	+6.09600036E-04	-2.10720996964787	-3.21495501602114
256	+3.90441E+05	+1.95880041E-04	-2.40823996531185	-3.70800981375086
512	+1.80273E+06	+1.32541788E-04	-2.70926996097583	-3.87764717503588



a PDE constrained problem

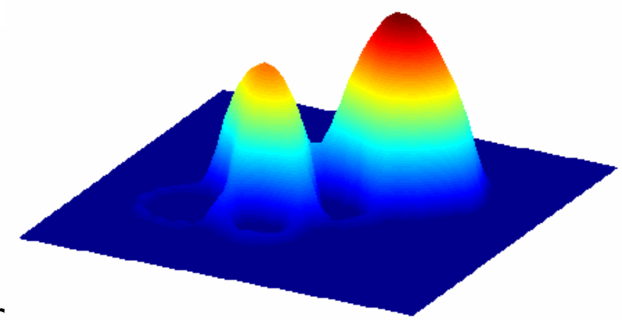
displacement matching for a membrane:

$$\min_{u_\mathcal{T},\phi} \mathcal{J}(\phi, u_\mathcal{T}) := \frac{1}{2}\int_\Omega (u_\mathcal{T} - u^*)^2 + \mathcal{R}(\phi)$$

$$\text{s.t.} \quad -\Delta u_\mathcal{T} + \frac{1}{\tau}(1 - \chi_\varepsilon(\phi))u_\mathcal{T} = f \quad \text{in}$$

$$u_\mathcal{T} = 0 \quad \text{or}$$

$$u^* = \text{ target displacement}$$



regularization: same as before

$$\mathcal{R}(\phi) = \frac{\beta}{4}\int_\Omega (|\nabla\phi|^2 - 1)^2 + \frac{\varepsilon}{2}\int_\Omega |\nabla\phi|^2$$

optimality conditions

Lagrangian:

$$\mathcal{L}(u_\mathcal{T},\phi,\lambda) = \frac{1}{2}\int_\Omega (u_\mathcal{T}-u^*)^2 + \frac{\beta}{4}\int_\Omega (|\nabla\phi|^2-1)^2 + \frac{\varepsilon}{2}\int_\Omega |\nabla\phi|^2$$
$$+ \int_\Omega \left(\nabla u_\mathcal{T}\cdot\nabla\lambda + \frac{1}{\tau}(1-\chi_\varepsilon(\phi))u_\mathcal{T}\lambda - f\lambda\right)$$

first-order necessary conditions:

find $(u_\mathcal{T},\lambda,\phi)\in(H_0^1\times H_0^1\times H^1)$ s.t.:

$$\int_\Omega (u_\mathcal{T}-u^*)\hat{u}_\mathcal{T} + \nabla\hat{u}_\mathcal{T}\cdot\nabla\lambda + \frac{1}{\tau}(1-\chi_\varepsilon(\phi))\hat{u}_\mathcal{T}\lambda = 0 \quad \forall \hat{u}_\mathcal{T}\in H_0^1$$

$$\int_\Omega \nabla u_\mathcal{T}\cdot\nabla\hat{\lambda} + \frac{1}{\tau}(1-\chi_\varepsilon(\phi))u_\mathcal{T}\hat{\lambda} - f\hat{\lambda} = 0 \quad \forall\hat{\lambda}\in H_0^1$$

$$\int_\Omega (\beta(\nabla\phi\cdot\nabla\phi-1)+\varepsilon)\nabla\phi\cdot\nabla\hat{\phi} - \frac{1}{\tau}u_\mathcal{T}\lambda\delta_\varepsilon(\phi)\hat{\phi} = 0 \quad \forall\hat{\phi}\in H^1$$

displacement matching

in strong form:

$$-\Delta\lambda + \frac{1}{\tau}(1-\chi_\varepsilon(\phi))\lambda = u_\tau - u^* \text{ in } \Omega$$

$$-\Delta u_\tau + \frac{1}{\tau}(1-\chi_\varepsilon(\phi))u_\tau = f \text{ in } \Omega$$

$$-\nabla\cdot[(\beta(\nabla\phi\cdot\nabla\phi - 1)+\varepsilon)\nabla\phi] - \frac{1}{\tau}u_\tau\lambda\delta_\varepsilon(\phi) = 0 \text{ in } \Omega$$

Hessian of the Lagrangian:

$$\delta^2_{u_\tau}\mathcal{L} = \int_\Omega \hat{u}_\tau\tilde{u}_\tau + \nabla\hat{u}_\tau\cdot\nabla\tilde{\lambda} + \frac{1}{\tau}(1-\chi_\varepsilon(\phi))\hat{u}_\tau\tilde{\lambda} - \frac{1}{\tau}\lambda\delta_\varepsilon(\phi)\hat{u}_\tau\tilde{\phi}$$

$$\delta^2_{\lambda}\mathcal{L} = \int_\Omega \nabla\tilde{u}_\tau\cdot\nabla\hat{\lambda} + \frac{1}{\tau}(1-\chi_\varepsilon(\phi))\tilde{u}_\tau\hat{\lambda} - \frac{1}{\tau}u_\tau\delta_\varepsilon(\phi)\hat{\lambda}\tilde{\phi}$$

$$\delta^2_{\phi}\mathcal{L} = \int_\Omega -\frac{1}{\tau}(\tilde{u}_\tau\lambda + u_\tau\tilde{\lambda})\delta_\varepsilon(\phi)\hat{\phi} - \frac{1}{\tau}u_\tau\lambda\delta'_\varepsilon(\phi)\hat{\phi}\tilde{\phi} +$$
$$\int_\Omega (\beta(|\nabla\phi|^2-1) + \varepsilon)\nabla\hat{\phi}\cdot\nabla\tilde{\phi} + 2\beta(\nabla\phi\cdot\nabla\hat{\phi})(\nabla\phi\cdot\nabla\tilde{\phi})$$