On two optimization methods for optimal control problems

Sequence of Quadratic Problems // SemiSmooth Newton Method

Mariano Mateos
(Universidad de Oviedo, España)

Based on joint works with Eduardo Casas (U. de Cantabria)

Third COPI2A Meeting
Castro Urdiales, December 1, 2025

The author was supported by MICIU/AEI/10.13039/501100011033/ under research project PID2023-147610NB-I00

Outline

1. Introduction
   - Abstract framework
2. The methods
   - Warning
   - SemiSmooth Newton method
   - SQP method
3. More computational details
   - Solving PDEs
   - Finite dimensional optimization
   - A numerical experiment

Outline

The abstract optimization problem

Problem (P)

$$\text{(P)} \qquad \min_{u \in U_{\rm ad}} J(u) := \mathcal{J}(u) + \frac{\kappa}{2}\|u\|^2_{L^2(X)},$$

- $(X, \mathcal{S}, \mu)$ measure space with $\mu(X) < \infty$, $\qquad \kappa > 0$.
- $\mathcal{J}$ is a function of class C^2

$$\mathcal{J} : \mathcal{A} \subset L^p(X) \to \mathbb{R}$$

  for some $p \in [2, +\infty]$. Here $\mathcal{A} \subset L^p(X)$ is an open set.

- $U_{\rm ad} \subset \mathcal{A}$ and

$$U_{\rm ad} = \{u \in L^p(X) : \ \alpha \leq u \leq \beta \text{ a.e. } [\mu]\}$$

  $-\infty \leq \alpha < \beta \leq +\infty$. If $p > 2$, we also require $-\infty < \alpha < \beta < +\infty$.

Notation: $B^p_\rho(u) = \{v \in L^p(X) : \ \|v - u\|_{L^p(X)} \leq \rho\}$.

Some control problems that fit in this framework

Remark

The framework is easy to generalize to vector controls $\boldsymbol{u} \in \Pi_{j=1}^{n} L^{p^j}(X_j)$.

- Additive elliptic control problem governed by a semilinear equation, with distributed and/or boundary control.
- Distributed control of the instationary Navier-Stokes equations
- Distributed and/or boundary control of a parabolic equation
- Time dependent control of a parabolic quasilinear equation
- Boundary bilinear control of a semilinear parabolic equation ...

Eduardo Casas (2024). "Superlinear Convergence of a Semismooth Newton Method for Some Optimization Problems with Applications to Control Theory". In: *SIAM Journal on Optimization* 34.4, pp. 3681–3698. DOI: 10.1137/24M1644286

Eduardo Casas and Mariano Mateos (2025b). *Quadratic convergence of an SQP method for some optimization problems with applications to control theory.* (To appear in SICON). arXiv: 2505.22750

A prototypical example

Problem (E)

$$\min_{u\in U_{\rm ad}} J(u) := \frac{1}{2}\|y_u - y_d\|^2_{L^2(\Omega)} + \frac{\kappa}{2}\|u\|^2_{L^2(\Omega)}$$

where $y_u \in H^1_0(\Omega) \cap C(\bar{\Omega})$ is the solution of

$$-\Delta y + \boldsymbol{b}\cdot\nabla y + f(\cdot, y) = u \text{ in } \Omega,\ \ y = 0 \text{ on } \Gamma.$$

and

$$U_{\rm ad} = \{u \in L^2(\Omega) :\ \alpha \le u(x) \le \beta \text{ for a.a. } x \in \Omega\}$$

- $\Omega \subset \mathbb{R}^d$, $d \le 3$, bounded domain with Lipschitz boundary Γ.
- $\boldsymbol{b} \in L^{\bar{p}}(\Omega)^d$, $\bar{p} > 2$, $\operatorname{div} \boldsymbol{b} \in L^2(\Omega)$, $y_d \in L^2(\Omega)$
- $f : \Omega \times \mathbb{R} \to \mathbb{R}$ a Carathéodory function, $f(\cdot, 0) \in L^2(\Omega)$, monotone non-decreasing, of class C^2 and such that $\partial^2_{yy} f(x, y)$ is locally Lipschitz w.r.t. y .

Casas, Eduardo, Mateos, Mariano, and Rösch, Arnd (2020). “Analysis of control problems of nonmontone semilinear elliptic equations”. In: *ESAIM: COCV* 26, p. 80. DOI: 10.1051/cocv/2020032

Some remarks about the example problem (E)

- $X = \Omega$, μ is the Lebesgue measure, $p = 2$.
- For every $u \in \mathcal{A} = L^2(\Omega)$ there exists a unique $y_u \in H_0^1(\Omega) \cap C(\bar{\Omega})$ solution of

$$-\Delta y + \boldsymbol{b} \cdot \nabla y + f(\cdot, y) = u \text{ in } \Omega, \; y = 0 \text{ on } \Gamma.$$

- The control-to-state mapping $G(u) = y_u$ is of class C^2.

Derivative of the control-to-state mapping

For all $u, v \in L^2(\Omega)$, $z_{u,v} = G'(u)v \in H_0^1(\Omega) \cap C(\bar{\Omega})$ is the unique solution of

$$-\Delta z + \boldsymbol{b} \nabla z + \partial_y f(\cdot, y_u) z = v \text{ in } \Omega, \; z = 0 \text{ on } \Gamma.$$

- $\mathcal{J}(u) = \frac{1}{2}\|y_u - y_d\|^2_{L^2(\Omega)}$. $J(u) = \frac{1}{2}\|y_u - y_d\|^2_{L^2(\Omega)} + \frac{\kappa}{2}\|u\|^2_{L^2(\Omega)}$.
- Corollary: $\mathcal{J}$ is of class C^2

The first derivative of the objective functional

For the **abstract problem, we assume:**

1. There exists a C^1 mapping $\Phi : \mathcal{A} \to L^\infty(X)$ such that

$$\mathcal{J}'(u)v = \int_X \Phi(u) v \, d\mu \ \forall u \in \mathcal{A} \text{ and } \ \forall v \in L^p(X).$$

For the **example:** For every $u \in L^2(\Omega)$ there exists a unique adjoint state $\varphi_u \in H_0^1(\Omega) \cap C(\bar{\Omega})$ solution of

$$-\Delta\varphi - \operatorname{div}[\boldsymbol{b}\varphi] + \partial_y f(\cdot, y_u)\varphi = y_u - y_d \text{ in } \Omega, \ \varphi = 0 \text{ on } \Gamma.$$

The mapping $\Phi(u) = \varphi_u$ is of class C^1 from $L^2(\Omega)$ into $L^\infty(\Omega)$ and integration by parts shows that

$$J'(u)v = \int_\Omega (\varphi_u + \kappa u) v \, dx \ \forall u \in L^2(\Omega) \text{ and } \forall v \in L^2(\Omega).$$

The second derivative

Remark: $\mathcal{J}$ is of class C^2. $\forall u \in \mathcal{A}$, $\mathcal{J}''(u) : L^p(X) \times L^p(X) \longrightarrow \mathbb{R}$ is a symmetric and continuous bilinear form and satisfies

$$\mathcal{J}''(u)(v_1, v_2) = \int_X [\Phi'(u)v_1]v_2 \,\mathrm{d}\mu = \int_X [\Phi'(u)v_2]v_1 \,\mathrm{d}\mu \quad \forall v_1, v_2 \in L^p(X). \tag{1}$$

We will write $\mathcal{J}''(u)v^2 = \mathcal{J}''(u)(v, v)$.

For the example: Classical form of the second derivative:

$$J''(u)v^2 = \int_\Omega \left[\left(1 - \varphi_u \partial^2_{yy} f(\cdot, y_u)\right) z^2_{u,v} + \kappa v^2\right] \mathrm{d}x.$$

But ... we will use the so-called **second-adjoint-state**. For all $u, v \in L^2(\Omega)$, $\eta_{u,v} = \Phi'(u)v \in H^1_0(\Omega) \cap C(\bar{\Omega})$ is the unique solution of

$$-\Delta\eta - \mathrm{div}[\boldsymbol{b}\eta] + \partial_y f(\cdot, y_u)\eta = (1 - \varphi_u \partial^2_{yy} f(\cdot, y_u)) z_{u,v} \text{ in } \Omega, \;\; \eta = 0 \text{ on } \Gamma.$$

Using $\eta_{u,v}$ (very helpful for computations!. We'll see later why).

$$J''(u)v^2 = \int_\Omega (\eta_{u,v} + \kappa v) v \,\mathrm{d}x.$$

Assumptions regarding the derivative of Φ

2. For every $u \in \mathcal{A}$ the linear mapping
$$\Phi'(u) : L^p(X) \longrightarrow L^\infty(X)$$
has an **extension to a compact** operator
$$\Phi'(u) : L^2(X) \longrightarrow L^2(X).$$
For every $\varepsilon > 0$ there exists $\rho = \rho_{\varepsilon,u} > 0$ with $B^p_\rho(u) \subset \mathcal{A}$ such that
$$\|(\Phi'(u) - \Phi'(w))v\|_{L^2(X)} \le \varepsilon \|v\|_{L^2(X)} \; \forall w \in B^p_\rho(u) \text{ and } \forall v \in L^2(X).$$
3. There exist $N \ge 0$ and numbers $2 = p_0 \le p_1 \le \ldots \le p_N = p$ such that for every $u \in \mathcal{A}$, the linear mapping
$$\Phi'(u) : L^p(X) \longrightarrow L^\infty(X)$$
defines also a **continuous** operator
$$\Phi'(u) : L^{p_{i-1}}(X) \longrightarrow L^{p_i}(X)$$
for $i = 1, \ldots, N$ (denote $p_{N+1} = \infty$).
4. For every $u \in \mathcal{A}$, there exists $\rho_u > 0$ with $B^p_{\rho_u}(u) \subset \mathcal{A}$ and a constant $L_{u,\Phi'}$ such that
$$\|(\Phi'(w) - \Phi'(u))v\|_{L^\infty(X)} \le L_{u,\Phi'} \|w - u\|_{L^p(X)} \|v\|_{L^p(X)}$$
for all $w \in B^p_{\rho_u}(u)$ and $v \in L^p(X)$.

Local solution. First order optimality condition.

- Let $\bar{u}$ be a local solution
  (In the sense of $L^2(X)$ if $2 < p < \infty$, taking advantage of $-\infty < \alpha < \beta < \infty$.)
- Any local solution satisfies the first order optimality condition

$$\int_X (\Phi(\bar{u}) + \kappa\bar{u})(u - \bar{u})\, \mathrm{d}\mu \geq 0 \; \forall u \in U_{\mathrm{ad}}.$$

  We will use this form to write the Sequence of Quadratic Programs.
- Equivalently

$$\bar{u}(x) = \mathrm{Proj}_{[\alpha\beta]}\left(-\frac{\Phi(\bar{u})(x)}{\kappa}\right) \text{ for a.a.}[\mu] x \in X$$

  We will use this form to write the SemiSmooth Newton method.

Assumptions on the local solution

5. The local minimizer $\bar{u} \in U_{\mathrm{ad}}$ satisfies the second order sufficient optimality condition
$$J''(\bar{u})v^2 > 0 \;\forall v \in C_{\bar{u}} \setminus \{0\}$$
and the strict complementarity condition
$$\mu\{x \in X : \bar{u}(x) \in \{\alpha, \beta\} \text{ and } \kappa\bar{u}(x) + \Phi(\bar{u})(x) = 0\} = 0.$$

Here, $C_{\bar{u}}$ is the cone of critical directions

$$C_{\bar{u}} = \left\{ v \in L^2(X) : \begin{cases} v(x) \geq 0 \text{ if } \bar{u}(x) = \alpha, \\ v(x) \leq 0 \text{ if } \bar{u}(x) = \beta, \\ v(x) = 0 \text{ if } \Phi(\bar{u})(x) + \kappa\bar{u}(x) \neq 0 \end{cases} \quad \text{a.e. } [\mu] \right\}.$$

Remark

This is completely analog to the usual textbook assumption to obtain local quadratic convergence of the SQP for finite-dimensional problems.

Outline

Warning

- There are versions of these methods that exploit the complete optimality system.

$$(\text{SQP})\begin{cases}\text{State equation}\\ \text{Adjoint state equation}\\ \text{Variational inequality}\end{cases}\qquad(\text{SSN})\begin{cases}\text{State equation}\\ \text{Adjoint state equation}\\ \text{Projection equation}\end{cases}$$

  - The three variables (y, φ, u) are treated independently.
  - These versions are fully linearized. There is no need to solve non-linear PDEs.

Fredi Tröltzsch (1999). "On the Lagrange–Newton–SQP Method for the Optimal Control of Semilinear Parabolic Equations". In: *SIAM Journal on Control and Optimization* 38.1, pp. 294–312. DOI: 10.1137/S0363012998341423

- In contrast, we will use u as the unique optimization variable.
- A non-linear PDE must be solved at each step.
- Robustness is gained regarding the choice of the initial point.
- A smart combination of both worlds is possible to achieve the best performance.

Eduardo Casas and Mariano Mateos (2025a). "Boundary bilinear control of semilinear parabolic PDEs: quadratic convergence of the SQP method". In: arXiv: 2505.24237 [math.OC]

Warning

- There are versions of these methods that exploit the complete optimality system.

(SQP) $\left\{ \begin{array}{c} \text{State equation} \\ \text{Adjoint state equation} \\ \text{Variational inequality} \end{array} \right.$ (SSN) $\left\{ \begin{array}{c} \text{State equation} \\ \text{Adjoint state equation} \\ \text{Projection equation} \end{array} \right.$

  - The three variables (y, φ, u) are treated independently.
  - These versions are fully linearized. There is no need to solve non-linear PDEs.

Fredi Tröltzsch (1999). "On the Lagrange–Newton–SQP Method for the Optimal Control of Semilinear Parabolic Equations". In: *SIAM Journal on Control and Optimization* 38.1, pp. 294–312. DOI: 10.1137/S0363012998341423

- In contrast, we will use u as the unique optimization variable.
- A non-linear PDE must be solved at each step.
- Robustness is gained regarding the choice of the initial point.
- A smart combination of both worlds is possible to achieve the best performance.

Eduardo Casas and Mariano Mateos (2025a). "Boundary bilinear control of semilinear parabolic PDEs: quadratic convergence of the SQP method". In: arXiv: 2505.24237 [math.OC]

Warning

- There are versions of these methods that exploit the complete optimality system.

$$\text{(SQP)} \left\{ \begin{array}{c} \text{State equation} \\ \text{Adjoint state equation} \\ \text{Variational inequality} \end{array} \right. \qquad \text{(SSN)} \left\{ \begin{array}{c} \text{State equation} \\ \text{Adjoint state equation} \\ \text{Projection equation} \end{array} \right.$$

  - The three variables (y, φ, u) are treated independently.
  - These versions are fully linearized. There is no need to solve non-linear PDEs.

Fredi Tröltzsch (1999). "On the Lagrange–Newton–SQP Method for the Optimal Control of Semilinear Parabolic Equations". In: *SIAM Journal on Control and Optimization* 38.1, pp. 294–312. DOI: 10.1137/S0363012998341423

- In contrast, we will use u as the unique optimization variable.
- A non-linear PDE must be solved at each step.
- Robustness is gained regarding the choice of the initial point.
- A smart combination of both worlds is possible to achieve the best performance.

Eduardo Casas and Mariano Mateos (2025a). "Boundary bilinear control of semilinear parabolic PDEs: quadratic convergence of the SQP method". In: arXiv: 2505.24237 [math.OC]

Outline

Semismoothness

(A farther layer of abstraction)

Definition 1 (Semismooth function)

Given two Banach spaces U and X, an open subset $\mathcal{A}$ of U, a continuous function $F : \mathcal{A} \longrightarrow X$, and a set-value mapping $\partial F : \mathcal{A} \longrightarrow \mathcal{P}(\mathcal{L}(U, X))$ such that $\partial F(u) \neq \emptyset \; \forall u \in \mathcal{A}$, we say that F is ∂F-semismooth at $\bar{u} \in \mathcal{A}$ if

$$\lim_{v \longrightarrow 0} \sup_{M \in \partial F(\bar{u}+v)} \frac{\|F(\bar{u}+v) - F(\bar{u}) - Mv\|_X}{\|v\|_U} = 0.$$

F is said ∂F-semismooth at $\mathcal{A}$ if it is ∂F-semismooth at every $u \in \mathcal{A}$.

Algorithm 1: Semismooth Newton method to solve $F(u) = 0$.

```
Initialize. Choose u_0 ∈ A. Set j = 0.
for j ≥ 0 do
    Choose M_j ∈ ∂F(u_j) and solve M_j v_j = −F(u_j).
    Set u_{j+1} = u_j + v_j and j = j + 1.
end
```

Convergence conditions for the SSN

Theorem 2

Suppose F is semismooth at $\bar{u}$, a locally unique solution of $F(u) = 0$, and that for every $j \geq 0$, $M_j \in \partial F(u_j)$ is invertible and there exists $C > 0$ such that

$$\|M_j^{-1}\|_{\mathcal{L}(X,U)} \leq C \; \forall j \geq 0.$$

Then there exists $\delta > 0$ such that for all $u_0 \in U$ with $\|u_0 - u\|_X < \delta$ the sequence $\{u_j\}_{j \geq 0}$ converges superlinearly to $\bar{u}$.

In short

Semismoothness + Uniform boundness of the inverses $\Rightarrow$ superlinear convergence.

Building the SemiSmooth Newton method for (E)

$$F : L^2(\Omega) \to L^2(\Omega),\ F(u)(x) = u(x) - \mathrm{Proj}_{[\alpha,\beta]}\left(\frac{-\varphi_u(x)}{\kappa}\right)$$

- First order optimality conditions: A local solution $\bar{u}$ is a solution of $F(u) = 0$.
- In order to define $\partial F(u)$ we introduce some additional functions.

$$\Phi : L^2(\Omega) \longrightarrow L^2(\Omega), \quad \Phi(u) = \varphi_u,$$
$$\psi : \mathbb{R} \longrightarrow \mathbb{R}, \quad \psi(t) = \mathrm{Proj}_{[\alpha,\beta]}(t),$$
$$\Psi : L^2(\Omega) \longrightarrow L^2(\Omega), \quad \Psi(u)(x) = \psi(\frac{-\Phi(u)(x)}{\kappa}).$$

Notice that $F(u) = u - \Psi(u)$.

- Ψ is called a superposition operator.

Computing ∂F.

- $\psi : \mathbb{R} \longrightarrow \mathbb{R}, \quad \psi(t) = \mathrm{Proj}_{[\alpha,\beta]}(t)$ is a Lipschitz function. The subdifferential in Clarke's sense is

$$\partial\psi(t) = \{1\} \text{ if } t \in (\alpha,\beta),\ \partial\psi(t) = \{0\} \text{ if } t \notin [\alpha,\beta],\ \partial\psi(t) = [0,1] \text{ if } t \in \{\alpha,\beta\}.$$

- $\Psi(u) = \psi(\frac{-\Phi(u)}{\kappa})$. For every $u \in L^2(\Omega)$ we define (applying the chain rule)

$$\partial\Psi(u) = \Big\{N \in \mathcal{L}(L^2(\Omega), L^2(\Omega)) : Nv(x) = h(x) \cdot \frac{-[\Phi'(u)v](x)}{\kappa}\ \forall v \in L^2(\Omega)$$

$$\text{where } h : \Omega \longrightarrow \mathbb{R} \text{ is any measurable function}$$

$$\text{such that } h(x) \in \partial\psi(\frac{-\Phi(u)(x)}{\kappa})\Big\}.$$

Theorem 3

Ψ is $\partial\Psi$-semismooth in $L^2(\Omega)$ and hence the function $F : L^2(\Omega) \longrightarrow L^2(\Omega)$ is ∂F-semismooth in $L^2(\Omega)$, where

$$\partial F(u) = \{M = I - N : N \in \partial\Psi(u)\}.$$

Selection of the M_u

- Define $\lambda : \mathbb{R} \longrightarrow \mathbb{R}$ so that $\lambda(t) \in \partial\psi(t)$ for every $t \in \mathbb{R}$, by

$$\lambda(t) = \begin{cases} 1 & \text{if } t \in (\alpha, \beta), \\ 0 & \text{otherwise,} \end{cases}$$

- Given $u \in L^2(\Omega)$, define $h_u : \Omega \to \mathbb{R}$ as

$$h_u(x) = \lambda\left(\frac{-\Phi(u)(x)}{\kappa}\right) = \begin{cases} 1 & \text{if } \frac{-\varphi_u(x)}{\kappa} \in (\alpha, \beta), \\ 0 & \text{otherwise.} \end{cases}$$

- If we define the inactive and active sets as

$$\mathbb{I}_u = \{x \in \Omega : \frac{-\varphi_u(x)}{\kappa} \in (\alpha, \beta)\} \qquad \mathbb{A}_u = \Omega \setminus \mathbb{I}_u,$$

  then h_u is the characteristic function of the inactive set

$$h_u = \chi_{\mathbb{I}_u}.$$

- Select $M_u \in \partial F(u)$ as the linear operator $M_u : L^2(\Omega) \to L^2(\Omega)$ defined by

$$M_u v = v - \chi_{\mathbb{I}_u} \cdot \frac{-\Phi'(u)v}{\kappa} = v + \chi_{\mathbb{I}_u} \cdot \frac{\eta_{u,v}}{\kappa}$$

The algorithm up to now

Algorithm 2: Semismooth Newton method to solve $F(u) = 0$.

Initialize. Choose $u_0 \in L^2(\Omega)$. Set $j = 0$.
for $j \geq 0$ **do**
    Compute $y_j = G(u_j)$ solving the nonlinear state equation
    Compute $\varphi_j = \Phi(u_j)$ solving the linear adjoint equation
    Set $\mathbb{A}_j = \mathbb{A}_j^\beta \cup \mathbb{A}_j^\alpha$ and $\mathbb{I}_j = \Omega \setminus \mathbb{A}_j$, where

$$\mathbb{A}_j^\beta = \{x \in \Omega : -\varphi_j(x) \geq \kappa\beta\}, \quad \mathbb{A}_j^\alpha = \{x \in \Omega : -\varphi_j(x) \leq \kappa\alpha\}$$

    Set $w_j(x) = -F(u_j)(x) = \begin{cases} -u_j(x) + \beta & \text{if } x \in \mathbb{A}_j^\beta \\ -u_j(x) - \frac{\varphi_j(x)}{\kappa} & \text{if } x \in \mathbb{I}_j \\ -u_j(x) + \alpha & \text{if } x \in \mathbb{A}_j^\alpha \end{cases}$

    Solve $v + \chi_{\mathbb{I}_j} \frac{\eta_{u_j,v}}{\kappa} = w_j$. Name v_j the solution..
    Set $u_{j+1} = u_j + v_j$ and $j = j + 1$.
end

Solving the linear system $v + \chi_{\mathbb{I}_u} \frac{\eta_{u,v}}{\kappa} = w$

- In the active set, the equation is reduced to $v = w$.
- So we can write that $v = \chi_{\mathbb{I}_u} v + \chi_{\mathbb{A}_u} w$.
- The mapping $v \mapsto \eta_{u,v}$ is linear in v. So $\eta_{u,v} = \eta_{u,\chi_{\mathbb{I}_u} v} + \eta_{u,\chi_{\mathbb{A}_u} w}$.
- Name $b = \kappa w - \eta_{u,\chi_{\mathbb{A}_u} w}$. In the inactive set we can write the equation as

$$\kappa \chi_{\mathbb{I}_u} v + \chi_{\mathbb{I}_u} \eta_{u,\chi_{\mathbb{I}_u} v} = \chi_{\mathbb{I}_u} b.$$

- This equation is the first order optimality condition of the unconstrained quadratic problem [*Notation*: $E_\Omega \upsilon$ is the extension by zero.]

$$(Q) \quad \min_{\upsilon \in L^2(\mathbb{I}_u)} \frac{1}{2} \int_{\mathbb{I}_u} (\eta_{u,E_\Omega \upsilon} + \kappa \upsilon) \upsilon \, \mathrm{d}x - \int_{\mathbb{I}_u} b \upsilon \, \mathrm{d}x.$$

- Solve (Q) for $\bar{\upsilon}$. The solution of $v + \chi_{\mathbb{I}_u} \frac{\eta_{u,v}}{\kappa} = w$ is

$$v = \begin{cases} \bar{\upsilon} & \text{in } \mathbb{I}_u \\ w & \text{in } \mathbb{A}_u \end{cases}$$

The algorithm at this point

Algorithm 3: Semismooth Newton method to solve $F(u) = 0$.

Initialize. Choose $u_0 \in L^2(\Omega)$. Set $j = 0$.
for $j \geq 0$ **do**
  Compute $y_j = G(u_j)$ solving the nonlinear state equation
  Compute $\varphi_j = \Phi(u_j)$ solving the linear adjoint equation
  Compute $\mathbb{A}_j^\beta$ and $\mathbb{A}_j^\alpha$. Set $\mathbb{A}_j = \mathbb{A}_j^\beta \cup \mathbb{A}_j^\alpha$ and $\mathbb{I}_j = \Omega \setminus \mathbb{A}_j$,
  Set $w_j(x) = -F(u_j)(x)$
  Compute $z_j = G'(u_j)\chi_{\mathbb{A}_j} w_j$ solving the linearized state equation
  Compute $\eta_j = \Phi'(u_j)\chi_{\mathbb{A}_j} w_j$ solving the second adjoint state equation
  Name $b_j = \kappa w_j - \eta_j$
  Solve the unconstrained quadratic problem

$$(Q) \quad \min_{v \in L^2(\mathbb{I}_j)} \frac{1}{2} \int_{\mathbb{I}_j} (\eta_{u_j, E_\Omega v} + \kappa v) v \, \mathrm{d}x - \int_{\mathbb{I}_u} b_j v \, \mathrm{d}x.$$

  Name v_j the solution.
  Set $u_{j+1} = u_j + \begin{cases} v_j & \text{in } \mathbb{I}_j \\ w_j & \text{in } \mathbb{A}_j \end{cases}$ and $j = j + 1$.
end

Solving the unconstrained quadratic problem (Q)

$$(Q) \quad \min_{v \in L^2(\mathbb{I}_u)} \frac{1}{2} \int_{\mathbb{I}_u} (\eta_{u,E_\Omega v} + \kappa v) v \, \mathrm{d}x - \int_{\mathbb{I}_u} b v \, \mathrm{d}x.$$

Side remark: Before solving this, notice that, since $E_\Omega v = 0$ in $\mathbb{A}_u$, we have

$$J''(u)(E_\Omega v)^2 = \int_{\mathbb{I}_u} (\eta_{u,E_\Omega v} + \kappa v) v \, \mathrm{d}x.$$

In finite dimension, if we had H, the Hessian matrix at u, then (Q) would read as

$$\min_{\boldsymbol{v} \in \mathbb{R}^N} \frac{1}{2} \boldsymbol{v}^T (H + \kappa I) \boldsymbol{v} - \boldsymbol{b}^T \boldsymbol{v}$$

and this would be just a matter of solving the linear system

$$(H + \kappa I) \boldsymbol{v} = \boldsymbol{b}$$

But, as many authors point out at this point, H is too expensive to compute. Since

$$H_{ij} = \int_{\mathbb{I}_u} (\eta_{u,E_\Omega e_i}) e_j \, \mathrm{d}x,$$

for a problem with N dof, you have to solve N^2 linear pdes to get H.

Solving the unconstrained quadratic problem (Q)

$$(Q)\quad \min_{v\in L^2(\mathbb{I}_u)} \frac{1}{2}\int_{\mathbb{I}_u}(\eta_{u,E_\Omega v}+\kappa v)v\,\mathrm{d}x-\int_{\mathbb{I}_u} bv\,\mathrm{d}x.$$

- Define the linear operator $H: L^2(\mathbb{I}_u)\to L^2(\mathbb{I}_u)$, [$R_{\mathbb{I}_u}$ is a restriction operator].

$$Hv = R_{\mathbb{I}_u}\eta_{u,E_\Omega v}$$

- Denoting $(\cdot,\cdot)$ the scalar product in $L^2(\mathbb{I})$, our problem reads

$$(Q)\quad \min_{v\in L^2(\mathbb{I}_u)} \frac{1}{2}([H+\kappa I]v,v)-(b,v).$$

- SSC+strict complementarity $\Rightarrow$ $H+\kappa I$ is a symmetric, positive definite operator. **Use the conjugate gradient** method to solve (Q) at the price of one evaluation of Hv per iteration.
- Cost of evaluation of Hv. Solve two linear PDEs:

$$-\Delta z+\boldsymbol{b}\nabla z+\partial_y f(\cdot,y_u)z=E_\Omega v \text{ in } \Omega,\ z=0 \text{ on } \Gamma,$$

$$-\Delta\eta-\mathrm{div}[\boldsymbol{b}\eta]+\partial_y f(\cdot,y_u)\eta=(1-\varphi_u\partial^2_{yy}f(\cdot,y_u))z \text{ in } \Omega,\ \eta=0 \text{ on } \Gamma.$$

Further considerations

- Under no-gap second order conditions and a strict complementarity condition, the algorithm converges locally superlinearly.
- If the equation is linear, then the SemiSmooth Newton method is equivalent to a Primal Dual Active Set Strategy.
- When solving **finite dimensional approximations**, there are three remarkable facts:
  1. For linear equations –so called **linear-quadratic control problems**– the problem is solved in a **finite number of iterations**. Each iteration is uniquely characterized by its active and inactive sets, so if these do not change from one iteration to the next one, we stop.
  2. For **nonlinear equations**, once the active and inactive sets are localized, the observed order of convergence is **quadratic**.
  3. The number of iterations is independent of the number of variables (**mesh-independence principle**, observed in experiments; I don't think it has been proved in this context).

Outline

Generalized equations

- Let $\bar{u}$ be a local solution (In the sense of $L^2(X)$ if $2 < p < \infty$, taking advantage of $-\infty < \alpha < \beta < \infty$.)
- Any local solution satisfies the first order optimality condition

$$\int_X (\Phi(\bar{u}) + \kappa \bar{u})(u - \bar{u}) \, \mathrm{d}\mu \geq 0 \; \forall u \in U_{\mathrm{ad}}.$$

- Let $F : \mathcal{A} \to L^p(X)$ be given by $F(u) = \Phi(u) + \kappa u$.
- Normal cone of U_{ad} at u

$$N(u) = \begin{cases} \{w \in L^2(X) : \int_X w(v - u) d\mu \leq 0 \; \forall v \in U_{\mathrm{ad}}\} & \text{if } u \in U_{\mathrm{ad}}, \\ \emptyset & \text{if } u \notin U_{\mathrm{ad}}. \end{cases}$$

Generalized equation

$$0 \in F(\bar{u}) + N(\bar{u})$$

The SQP method

- Generalized Newton's method: Given $u_0 \in \mathcal{A}$, for $j \geq 0$ u_{j+1} solves

$$0 \in F(u_j) + F'(u_j)(u_{j+1} - u_j) + N(u_{j+1})$$

- This generalized equation is the first order optimality condition of the constrained quadratic problem

$$(\mathcal{Q}_j) \qquad \min_{u \in U_{\mathrm{ad}}} \frac{1}{2} J''(u_j)(u - u_j)^2 + J'(u_j)u.$$

- $(\mathcal{Q}_j)$ may have no solution or may have more than one solution. Instead, we will look for local solutions.

Remark: To solve $(\mathcal{Q}_j)$ we do the change $v = u - u_j$ and use

$$J''(u_j)v^2 = \int_\Omega \left(\kappa v + \eta_{u_j,v}\right) v \, \mathrm{d}x \qquad J'(u_j)v = \int_\Omega \left(\kappa u_j + \varphi_{u_j}\right) v \, \mathrm{d}x$$

The algorithm

Algorithm 4: SQP method to solve (E).

Initialize. Choose $u_0 \in L^2(\Omega)$. Set $j = 0$.
for $j \geq 0$ **do**
  Compute $y_j = G(u_j)$ solving the nonlinear state equation
  Compute $\varphi_j = \Phi(u_j)$ solving the linear adjoint equation
  Find a local solution of the constrained quadratic problem

$$(\mathcal{Q}_j') \qquad \min_{v \in U_{\mathrm{ad}} - \{u_j\}} \frac{1}{2} \int_\Omega \left(\kappa v + \eta_{u_j,v}\right) v \,\mathrm{d}x + \int_\Omega \left(\kappa u_j + \varphi_j\right) v \,\mathrm{d}x$$

  Name v_j the obtained solution.
  Set $u_{j+1} = u_j + v_j$ and $j = j + 1$.
end

Theorem 4

Under no-gap second order sufficient optimality conditions and a strict complementarity condition, the method converges quadratically to $\bar{u}$ both in $L^2(\Omega)$ and $L^\infty(\Omega)$ provided an initial point u_0 is given in a proper neighborhood of $\bar{u}$ in the sense of $L^2(\Omega)$.

Solving $(\mathcal{Q}'_j)$

$$(\mathcal{Q}') \qquad \min_{\alpha - u(x) \leq v(x) \leq \beta - u(x)} \frac{1}{2} \int_\Omega \big(\kappa v + \eta_{u,v}\big) v \, \mathrm{d}x + \int_\Omega \big(\kappa u + \varphi\big) v \, \mathrm{d}x$$

- Some authors write $(\mathcal{Q}')$ as a linear-quadratic optimal control problem. Let's do it more interesting, more general, and *easier*.
- Let $(X, \mathcal{S}, \mu)$ be measure space, $H \in \mathcal{L}(L^2(X))$ a self-adjoint operator, $b \in L^2(X)$, $\tilde{\alpha}(x)$ and $\tilde{\beta}(x)$ measurable functions (maybe taking $\pm\infty$ values), and $\kappa > 0$.

$$(\mathcal{Q}') \qquad \min_{\tilde{\alpha}(x) \leq v(x) \leq \tilde{\beta}(x)} \frac{1}{2}(Hv + \kappa v, v) + (b, v).$$

- In our case $Hv = \eta_{u,v}$ and $b = \kappa u + \varphi$, $\tilde{\alpha}(x) = \alpha - u(x)$, $\tilde{\beta}(x) = \beta - u(x)$.

To solve the constrained quadratic problem ...

Use SemiSmooth Newton method !!!!

Solving $(\mathcal{Q}_j')$

$$(\mathcal{Q}') \qquad \min_{\alpha - u(x) \leq v(x) \leq \beta - u(x)} \frac{1}{2} \int_\Omega \big(\kappa v + \eta_{u,v}\big) v \, dx + \int_\Omega \big(\kappa u + \varphi\big) v \, dx$$

- Some authors write $(\mathcal{Q}')$ as a linear-quadratic optimal control problem. Let's do it more interesting, more general, and *easier*.
- Let $(X, \mathcal{S}, \mu)$ be measure space, $H \in \mathcal{L}(L^2(X))$ a self-adjoint operator, $b \in L^2(X)$, $\tilde{\alpha}(x)$ and $\tilde{\beta}(x)$ measurable functions (maybe taking $\pm\infty$ values), and $\kappa > 0$.

$$(\mathcal{Q}') \qquad \min_{\tilde{\alpha}(x) \leq v(x) \leq \tilde{\beta}(x)} \frac{1}{2}(Hv + \kappa v, v) + (b, v).$$

- In our case $Hv = \eta_{u,v}$ and $b = \kappa u + \varphi$, $\tilde{\alpha}(x) = \alpha - u(x)$, $\tilde{\beta}(x) = \beta - u(x)$.

To solve the constrained quadratic problem ...

Use SemiSmooth Newton method !!!!

SemiSmooth Newton method to solve constrained quadratic problems

Algorithm 5: SemiSmooth method to solve a constrained quadratic problem.

Initialize. Choose $v_0 \in L^2(X)$. Set $n = 0$.

Compute $\Phi_n = Hv_n + b$; Set $\mathbb{A}_n = \mathbb{A}_n^\beta \cup \mathbb{A}_n^\alpha$ and $\mathbb{I}_n = X \setminus \mathbb{A}_n$, where

$$\mathbb{A}_n^\beta = \{x \in X : \; -\Phi_n(x) \geq \kappa\tilde{\beta}(x)\}, \; \mathbb{A}_n^\alpha = \{x \in X : \; -\Phi_n(x) \leq \kappa\tilde{\alpha}(x)\}.$$

for $n \geq 0$ **do**

Set $w_n(x) = \begin{cases} -v_n(x) + \tilde{\beta}(x) & \text{if } x \in \mathbb{A}_n^\beta \\ -v_n(x) - \frac{\Phi_n(x)}{\kappa} & \text{if } x \in \mathbb{I}_n \\ -v_n(x) + \tilde{\alpha}(x) & \text{if } x \in \mathbb{A}_n^\alpha \end{cases}$

Compute $\eta_n = H\chi_{\mathbb{A}_n} w_n$ and set $b_n = \chi_{\mathbb{I}_n} w_n - \chi_{\mathbb{I}_n}\eta_n$

Solve, for $\bar{\upsilon}_n$ the unconstrained quadratic problem $\min\limits_{\upsilon \in L^2(\mathbb{I}_n)} \frac{1}{2}(R_{\mathbb{I}_n} HE_X\upsilon + \kappa\upsilon, \upsilon) - (b, \upsilon)$

Set $v_{n+1} = w_n$ in $\mathbb{A}_n$ and $v_{n+1} = \bar{\upsilon}_n$ in $\mathbb{I}_n$.

Compute $\Phi_{n+1} = Hv_{n+1} + b$, $\mathbb{A}_{n+1}^\beta$, $\mathbb{A}_{n+1}^\alpha$, $\mathbb{A}_{n+1}$ and $\mathbb{I}_{n+1}$

For **finite-dimensional** problems: stop if $\mathbb{A}_n^\beta = \mathbb{A}_{n+1}^\beta$ and $\mathbb{A}_n^\alpha = \mathbb{A}_{n+1}^\alpha$

Set $n = n + 1$

end

Outline

Routines we need: (1) solving nonlinear PDEs

1. Given u, we need a routine to compute $y_u = G(u)$, this is, to solve the nonlinear equation

$$-\Delta y + \boldsymbol{b} \cdot \nabla y + f(\cdot, y) = u \text{ in } \Omega, \;\; y = 0 \text{ on } \Gamma.$$

I use Newton's method for the equation $\mathcal{F}(y) = 0$, where

$$\mathcal{F}(y) = -\Delta y + \boldsymbol{b} \cdot \nabla y + f(\cdot, y) - u = 0.$$

Notice that

$$\mathcal{F}'(y)z = -\Delta z + \boldsymbol{b} \cdot \nabla z + \partial_y f(\cdot, y)z = 0.$$

So, naming $z = y_{k+1} - y_k$, the Newton equation

$$\mathcal{F}'(y_k)z = -\mathcal{F}(y_k)$$

can be written, doing the proper cancellations in the linear terms, as

$$-\Delta y_{k+1} + \boldsymbol{b} \cdot \nabla y_{k+1} + \partial_y f(\cdot, y_k)y_{k+1} = u - f(\cdot, y_k) + \partial_y f(\cdot, y_k)y_k$$

FEM approximation

- Let $\mathcal{K}$ be the stiffness matrix such that $\mathcal{K}\boldsymbol{y}$ models $-\Delta y$
- Let $\mathcal{T}$ be the transport matrix such that $\mathcal{T}\boldsymbol{y}$ models $\boldsymbol{b} \cdot \nabla y$
  Notice that $\mathcal{T}^T\boldsymbol{y}$ models $-\operatorname{div}[\boldsymbol{b}\varphi]$
- Name $\mathcal{A} = \mathcal{K} + \mathcal{T}$
- Let $\mathcal{M}_0$ be the mass matrix such that $\mathcal{M}_0\boldsymbol{u}$ models the rhs u.
- **At each step**, we have to **assemble a mass matrix** $\mathcal{M}^k$ that models the action of $\partial_y f(x, y_k)$

$$\mathcal{M}^k_{i,j} = \int_\Omega e_i(x)\partial_y f(x, y^k_h(x))e_j(x)\,\mathrm{d}x.$$

And also **assemble a vector** $\mathfrak{f}^k$ to model $f(x, y_k)$

$$\mathfrak{f}^k_j = \int_\Omega f(x, y^k_h(x))e_j(x)\,\mathrm{d}x.$$

Solving the linear systems

At each step we have to solve the linear system

$$[\mathcal{A} + \mathcal{M}^k]\boldsymbol{y} = \mathcal{M}_0 \boldsymbol{u} + \mathcal{M}^k \boldsymbol{y}^k - \mathfrak{f}^k.$$

Since we have a transport term, the matrix is *not* symmetric. Use an LU decomposition. I use the following trick in MATLAB, which uses a scaling diagonal matrix `D`, row permutation `p` and column permutation `q`

```
[L,U,p,q,D] = lu(A+Mk,'vector'); D = spdiags(D); D = D(p);
```

We can solve `(A+Mk)y = b`, with a very efficient one-liner, that solves two very sparse triangular systems.

```
y(q,1) = U\(L\(D.\b(p)))
```

We stop the method when $\|y_h^k - y_h^{k+1}\|$ is “small”. This means that $\mathcal{M}^k \approx \mathcal{M}^{k+1}$, so we store in memory the last `L,U,p,q,D`. You will see now how helpful this is.

Routines we need: (2) solving the linearized PDE

Given u, y_u, and v, we need a routine to compute $z_{u,v} = G'(u)v$

$$-\Delta z + \boldsymbol{b} \cdot \nabla z + \partial_y f(\cdot, y_u) z = v$$

The y_u that appears here is the one that comes from Newton's method!. So the system to solve is

$$[\mathcal{A} + \mathcal{M}^{k+1}]\boldsymbol{z} = \mathcal{M}_0 \boldsymbol{v}$$

Since $\mathcal{M}^{k+1} \approx \mathcal{M}^k$, we use the same coefficient matrix that we have already used and factorized! Name $\boldsymbol{b} = \mathcal{M}_0 \boldsymbol{v}$ and do

```
z(q,:) = U\(L\(D.\b(p,:)))
```

Remark:

- We will also use this with Matlab `fmincon`, which will need to compute $z_{u,v}$ for several directions v at the same time, hence the `:`.

Routines we need: (3) solving the adjoint equations

Given u, y_u, we need a routine to compute φ_u

$$-\Delta\varphi - \mathrm{div}[\boldsymbol{b}\varphi] + \partial_y f(\cdot, y_u)\varphi = y_u - y_d$$

Given u, y_u, φ_u, v and $z_{u,v}$, we need a routine to compute $\eta_{u,v}$

$$-\Delta\eta - \mathrm{div}[\boldsymbol{b}\eta] + \partial_y f(\cdot, y_u)\eta = (1 - \varphi_u f''(y_u))z_{u,v}$$

The coefficient matrix that appears in the FEM approximation of these equations is the *transpose* of $[\mathcal{A} + \mathcal{M}^{k+1}]$. We can take advantage of the factorization that we have using MATLAB's forward slash (`mrdivide`).

Let $\mathcal{M}$ be the mass matrix such that $\|y_h\|^2_{L^2(\Omega)} = \boldsymbol{y}^T\mathcal{M}\boldsymbol{y}$.
For the adjoint state, name $\boldsymbol{b} = \mathcal{M}(\boldsymbol{y} - \boldsymbol{y}_d)$ and solve

```
phi(p,1) = ((b(q)'/U)/L)'./D;
```

Second adjoint state

For the second adjoint state, I **assemble** another mass matrix $\mathfrak{M}$

$$\mathfrak{M}_{i,j} = \int_\Omega e_i(x)\left(1 - \varphi_h(x)\partial^2_{yy} f(x, y_h(x))\right) e_j(x)\, \mathrm{d}x$$

and name $\boldsymbol{b} = \mathfrak{M}\boldsymbol{z}$. Then solve

```
eta(p,:) = ((b(q,:)'/U)/L)'./D;
```

Remark:

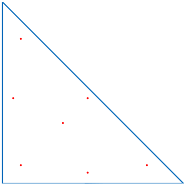
- We will have to compute $\mathfrak{M}\boldsymbol{z}$ for several $\boldsymbol{z}$, so it is better to assemble first $\mathfrak{M}$ than assembling $\boldsymbol{b} = \mathfrak{M}\boldsymbol{z}$ directly.
- We will also use this with Matlab `fmincon`, which needs to compute $\eta_{u,v}$ for several directions v at the same time, hence the `:`.

Assembling all those matrices

- One of the keys to fast computation is the ability to assemble efficiently all those mass matrices that change at every iteration.
- Numerical integration formulas must be used. Just the trapezoid or the midpoint formula may not do the work.
  - We may have to integrate difficult functions like

$$\int_T e_i(x)\varphi_h(x)\mathrm{e}^{y_h(x)}e_j(x)\,\mathrm{d}x, \text{ or } \int_T e_i(x)\varphi_h(x)y_h(x)|y_h(x)|e_j(x)\,\mathrm{d}x$$

  - We may have singularities near non-convex corners.
- An equilibrium must be found among the number of points and the accuracy. I use either the mid-edges formula (3 nodes, order 2) or a seven-point formula of order 5.



Ronald Cools (2022). *Encyclopaedia of Cubature Formulas.* URL: https://nines.cs.kuleuven.be/ecf/

Outline

Discretize, then optimize

Assume $\boldsymbol{y} \in \mathbb{R}^{N_y}$ and $\boldsymbol{u} \in \mathbb{R}^{N_u}$.

$$J(\boldsymbol{u}) = \frac{1}{2}(\boldsymbol{y} - \boldsymbol{y}_d)^T \mathcal{M}(\boldsymbol{y} - \boldsymbol{y}_d) + \frac{\kappa}{2}\boldsymbol{u}^T \mathcal{D}\boldsymbol{u},$$

where $\boldsymbol{u}^T\mathcal{D}\boldsymbol{u}$ is/approximates $\|u_h\|^2_{L^2(\Omega)}$

$$\begin{aligned} \mathcal{A}\boldsymbol{y} + \mathfrak{f}(\boldsymbol{y}) &= \mathcal{M}_0\boldsymbol{u} \\ \mathcal{A}\boldsymbol{z} + \mathcal{F}'(\boldsymbol{y})\boldsymbol{z} &= \mathcal{M}_0\boldsymbol{v} \\ \mathcal{A}^T\boldsymbol{\varphi} + \mathcal{F}'(\boldsymbol{y})\boldsymbol{\varphi} &= \mathcal{M}(\boldsymbol{y} - \boldsymbol{y}_d) \\ \mathcal{A}^T\boldsymbol{\eta} + \mathcal{F}'(\boldsymbol{y})\boldsymbol{\eta} &= \mathfrak{M}\boldsymbol{z} \end{aligned}$$

$$\begin{aligned} J'(\boldsymbol{u})\boldsymbol{v} &= \boldsymbol{v}^T(\mathcal{M}_0^T\boldsymbol{\varphi} + \kappa\mathcal{D}\boldsymbol{u}) \\ J''(\boldsymbol{u})\boldsymbol{v} &= \boldsymbol{v}^T(\mathcal{M}_0^T\boldsymbol{\eta} + \kappa\mathcal{D}\boldsymbol{v}) \end{aligned}$$

Can we use a diagonal $\mathcal{D}$?

- We can choose $\mathcal{D}$ diagonal in the following cases:
  1. $U_h = U_{h,0}$ is formed by piecewise constant approximations of the controls.
  2. $U_h = U_{h,1}$ is formed by piecewise linear approximations of the controls, and we use the composite trapezoid rule to approximate $\int_\Omega u_h(x)^2 \, dx$.
- The relation between discrete optimal control and discrete optimal adjoint state is respectively, as follows
  1. $\bar{u}_h(x) = \text{Proj}_{[\alpha,\beta]}\left(\frac{Q_h\bar{\varphi}_h}{\kappa}\right)$ where $Q_h : L^1(\Omega) \to U_h$ is the projection in the sense of $L^2(\Omega)$ onto U_h.
  2. $\bar{u}_h(x) = \text{Proj}_{[\alpha,\beta]}\left(\frac{C_h\bar{\varphi}_h}{\kappa}\right)$ where $C_h : L^1(\Omega) \to U_h$ is the Carstensen interpolation operator.
- In both cases we have that $Q_h\bar{\varphi}_h \neq \bar{\varphi}_h$ and $C_h\bar{\varphi}_h \neq \bar{\varphi}_h$. This has the following effects:
  - The approximation error $O(h^2)$ in $L^2(\Omega)$ of the FEM is lost. We will have $O(h)$ in the first case and $o(h)$ in the second case.
  - In the case $\alpha \leq 0 \leq \beta$, we have that $\bar{u} = 0$ on Γ. This is also lost. In general $\bar{u}_h \not\equiv 0$ on Γ.

Ideas about semismooth Newton

Usin that $\mathcal{D}$ is diagonal.
Instead of writing the first order optimality conditions written as

$$\bar{u}_i = \text{Proj}_{[\alpha,\beta]} \left(\frac{-[\mathcal{M}_0 \bar{\varphi}]_i}{\kappa d_{ii}} \right),$$

I prefer to denote $\alpha_i = \kappa d_{ii}\alpha$ and $\beta_i = \kappa d_{ii}\beta$, and write

$$\kappa d_{ii} \bar{u}_i = \text{Proj}_{[\alpha_i,\beta_i]} \left(-[\mathcal{M}_0 \bar{\varphi}]_i \right)$$

So the function $\boldsymbol{F}$ for the SemiSmooth Newton method is given by

$$[\boldsymbol{F}(\boldsymbol{u})]_i = \kappa d_{ii} u_i - \text{Proj}_{[\alpha_i,\beta_i]} \left(-[\mathcal{M}_0 \varphi]_i \right)$$

In the algorithm one has to be careful, because now w is defined by $-\kappa d_{ii} w_i = -[\boldsymbol{F}(\boldsymbol{u})]_i$. Also the solution of unconstrained quadratic program requires the use of a **preconditioner**. $\kappa\mathcal{D}$ does the work!

What if $\mathcal{D}$ is not diagonal?

- $U_h = U_{h,1}$ is formed by piecewise linear approximations of the controls, and we compute $\int_\Omega u_h(x)^2 \, dx = \boldsymbol{u}^T \mathcal{M} \boldsymbol{u}$, which is exact.
- In this case $\mathcal{M}_0 = \mathcal{M}$ and $\mathcal{D} = \mathcal{M}$. Order of convergence $O(h^{3/2})$ is obtained.
- The first order optimality condition

$$(\boldsymbol{u} - \bar{\boldsymbol{u}})^T (\mathcal{M} \bar{\boldsymbol{\varphi}} + \kappa \mathcal{M} \bar{\boldsymbol{u}}) \geq 0 \;\forall \boldsymbol{\alpha} \leq \boldsymbol{u} \leq \boldsymbol{\beta}$$

  cannot be written in a componentwise form. We lose the pointwise projection formula.
- We can write the first order optimality condition using Lagrange multipliers for the constraints $\boldsymbol{\alpha} \leq \boldsymbol{u} \leq \boldsymbol{\beta}$.
  There exists $\bar{\boldsymbol{\lambda}} = \bar{\boldsymbol{\lambda}}_\beta - \bar{\boldsymbol{\lambda}}_\alpha$ such that

$$\begin{aligned}
&\mathcal{M} \bar{\boldsymbol{\varphi}} + \kappa \mathcal{M} \bar{\boldsymbol{u}} + \bar{\boldsymbol{\lambda}} = 0 \\
&\boldsymbol{\alpha} \leq \bar{\boldsymbol{u}} \leq \boldsymbol{\beta} \\
&\bar{\boldsymbol{\lambda}}_\beta \geq \mathbf{0}, \; \bar{\boldsymbol{\lambda}}_\alpha \geq \mathbf{0} \\
&\bar{\boldsymbol{\lambda}}_\beta^T (\bar{\boldsymbol{u}} - \boldsymbol{\beta}) = 0 \\
&\bar{\boldsymbol{\lambda}}_\alpha^T (\bar{\boldsymbol{u}} - \boldsymbol{\alpha}) = 0
\end{aligned}$$

Semismooth equation for not diagonal $\mathcal{D}$.

- For any $c > 0$, the previous relations can be written as

$$\begin{aligned} &\mathcal{M}\bar{\varphi} + \kappa \mathcal{M}\bar{\boldsymbol{u}} + \bar{\boldsymbol{\lambda}} = 0 \\ &\bar{\boldsymbol{\lambda}} = \max\{\mathbf{0}, \bar{\boldsymbol{\lambda}} + c(\bar{\boldsymbol{u}} - \boldsymbol{\beta})\} + \min\{\mathbf{0}, \bar{\boldsymbol{\lambda}} + c(\bar{\boldsymbol{u}} - \boldsymbol{\alpha})\} \end{aligned}$$

- Define $\boldsymbol{F} : \mathbb{R}^N \times \mathbb{R}^N \longrightarrow \mathbb{R}^N \times \mathbb{R}^N$ as

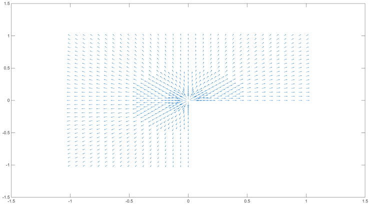
$$\boldsymbol{F}(\boldsymbol{u}, \boldsymbol{\lambda}) = \left[\begin{array}{l} \mathcal{M}\varphi + \kappa \mathcal{M}\boldsymbol{u} + \boldsymbol{\lambda} \\ \boldsymbol{\lambda} - \max\{\mathbf{0}, \boldsymbol{\lambda} + c(\boldsymbol{u} - \boldsymbol{\beta})\} - \min\{\mathbf{0}, \boldsymbol{\lambda} + c(\boldsymbol{u} - \boldsymbol{\alpha})\} \end{array} \right]$$

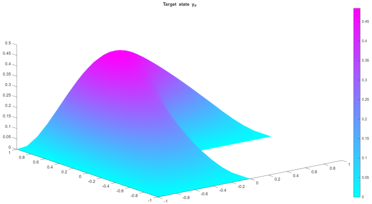
- Apply the SemiSmooth Newton method to this problem.
  (Exercise for the reader).

Outline

Data for the example

- Ω is the L-shaped domain $[-1,1]^2 \setminus [-1,0]^2$. Notice that its biggest interior angle is $\omega = 3\pi/2$. Name $\lambda = \pi/\omega = 2/3$.
- We select data to have a non-monotone and non-coercive nonlinear operator: $f(x,y) = y^3|y|$. $b(x) = \delta r^{1+\gamma}(\cos\theta, \sin(\theta))$. Here (r,θ) are the polar coordinates, and we select $\delta = 6$ and $\gamma = -1.25$.
- $y_d(x) = (1-x^2)(1-y^2)r^{\lambda}\sin(\lambda\theta)$.
- $\kappa = 10^{-4}$, $\alpha = 0$, $\beta = 10$ (really ill posed!).

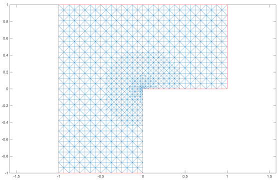




Discretization choices

- Finite element method. Lagrange P1 functions for the state and the adjoint state.
- Graded mesh family obtained by the “bisection method”. Grading parameter $\mu = 2/3$.
- The controls are discretized also with Lagrange P1 elements. The term $\|u\|^2_{L^2(\Omega)}$ is discretized using the composite trapezoid rule, which means $\boldsymbol{u}^T \mathcal{D} \boldsymbol{u}$ where $\mathcal{D}$ is the diagonal lumped mass matriz:

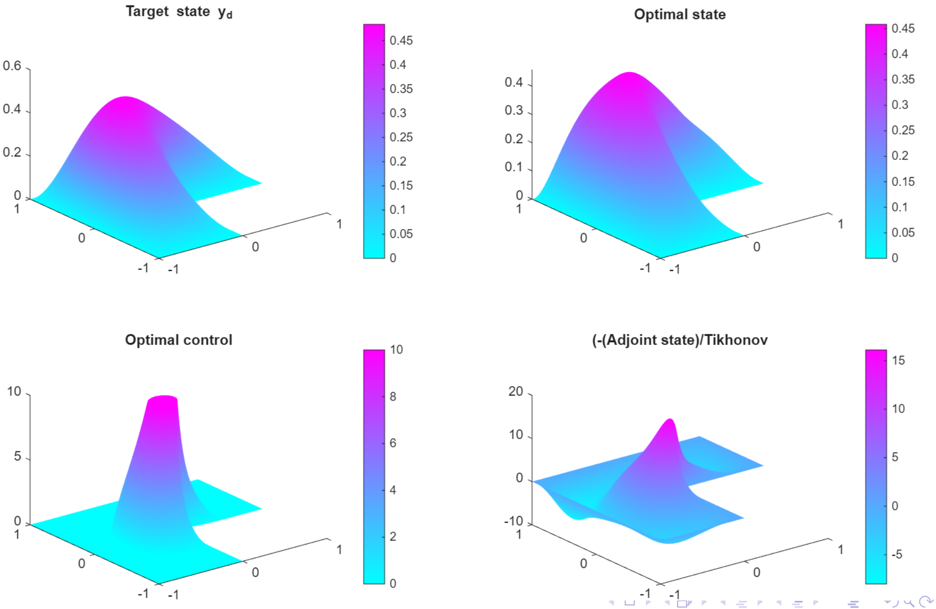
$$d_{ii} = \int_\Omega e_i \, \mathrm{d}x.$$



Choosing the initial point

- To solve the problem with meshsize h, use a sequence of (nested) meshes of sizes $h_1 \geq h_2 \geq \ldots \geq h_N = h$.
- Also, use a sequence $\kappa_1 \geq \kappa_2 \geq \ldots \kappa_N = \kappa$.
- For the problem of size h_i with Tikhonov parameter κ_i, use as initial point the solution of the problem of size h_{i-1}, with κ_{i-1}.
- The problem h_1 with κ_1 is of small size and not ill posed. In our case $u_0 \equiv 0$ works.

Solution



Target state y_d
Optimal state
Optimal control
(-(Adjoint state)/Tikhonov

Some results: (1) SemiSmooth Newton

```
kappa = 1.0000e+00
Mesh data: refinements = 1, dim Y_h = 8, dim U_h = 24.
SemiSmooth Newton method solving nonlinear equations
  j &       J(u)             &   delta_u    & Fact &   CG &    #J   &  #Ab &   #Aa \\

---- ----------------------- ------------- ------ ------ -------- ------ --------
  0 &  8.6591166282472015e-02 &       Inf    &    1 &    0 &      0      24        0 \\
  1 &  8.5977499674872490e-02 &  5.2e-02    &    2 &    2 &     24       0        0 \\
  2 &  8.5977499674872476e-02 &  2.5e-09    &    2 &    3 &     24       0        0 \\
.
.
.
kappa = 1.0000e-04
Mesh data: refinements = 8, dim Y_h = 237368, dim U_h = 239526.
SemiSmooth Newton method solving nonlinear equations
  j &       J(u)             &   delta_u    & Fact &   CG &    #J   &  #Ab &   #Aa \\

---- ----------------------- ------------- ------ ------ -------- ------ --------
  0 &  1.5050899562244532e-03 &       Inf    &    2 &    0 &  86646  138045   14835 \\
  1 &  1.5050875185621496e-03 &  3.1e-02    &    3 &   11 &  85947  138643   14936 \\
  2 &  1.5050875082492838e-03 &  4.7e-03    &    3 &   14 &  85995  138452   15079 \\
  3 &  1.5050875081594375e-03 &  7.6e-05    &    3 &   13 &  85993  138452   15081 \\
  4 &  1.5050875081590892e-03 &  7.4e-10    &    3 &   18 &  85993  138452   15081 \\
  5 &  1.5050875081590843e-03 &  9.6e-12    &    2 &   22 &  85993  138452   15081 \\
  6 &  1.5050875081590817e-03 &  1.2e-13    &    1 &   22 &  85993  138452   15081 \\
  7 &  1.5050875081590821e-03 &  1.8e-15    &    1 &   26 &  85993  138452   15081 \\

tiempo =

  44.360280099999997
```

Some results: (2) Sequence of Quadratic Programs

```
kappa = 1.0000e+00
Mesh data: refinements = 1, dim Y_h = 8, dim U_h = 24.
SQP method solving nonlinear equations
  n &       J(u_n)             &  delta_u   & Fact &  QP  &  CG  &   #J  &   #Aa  &  #Ab \\

---- ------------------------   ---------- ------ ------ ------- -------- -------
  0 &  8.6591166282472015e-02 &       Inf  &    1 &    0 &    0 &      0      24       0 \\
  1 &  8.5977499674872490e-02 &  5.2e-02   &    2 &    1 &    2 &     24       0       0 \\
  2 &  8.5977499674872476e-02 &  2.5e-09   &    2 &    1 &    3 &     24       0       0 \\
.
.
.
kappa = 1.0000e-04
Mesh data: refinements = 8, dim Y_h = 237368, dim U_h = 239526.
SQP method solving nonlinear equations
  n &       J(u_n)             &  delta_u   & Fact &  QP  &  CG  &   #J  &   #Aa  &  #Ab \\

---- ------------------------   ---------- ------ ------ ------- -------- -------
  0 &  1.5050899562244540e-03 &       Inf  &    2 &    0 &    0 &  86646  138045   14835 \\
  1 &  1.5050875096717187e-03 &  3.1e-02   &    3 &    2 &   22 &  85993  138452   15081 \\
  2 &  1.5050875081708251e-03 &  2.2e-06   &    4 &    1 &   13 &  85993  138452   15081 \\
  3 &  1.5050875081591864e-03 &  3.0e-08   &    4 &    1 &   21 &  85993  138452   15081 \\
  4 &  1.5050875081590721e-03 &  3.5e-10   &    3 &    1 &   22 &  85993  138452   15081 \\
  5 &  1.5050875081590713e-03 &  4.1e-12   &    2 &    1 &   22 &  85993  138452   15081 \\
  6 &  1.5050875081590962e-03 &  4.9e-14   &    1 &    1 &   24 &  85993  138452   15081 \\

tiempo =

  46.425035999999999
```

Some results: (3) SemiSmooth Newton using Lagrange multipliers. No mass lumping!

Slightly different problem, hence slightly different solution, probably more accurate

```
kappa = 1.0000e+00
Mesh data: refinements = 1, dim Y_h = 8, dim U_h = 24.
SemiSmooth Newton method. Using multipliers and solving nonlinear equations
  j &       J(u)             &   delta_u    & Fact &    CG &    #J   &  #Ab &    #Aa \\
____ _______________________ _____________ ______ ______ ________ ______ ________
  0 &  8.6591166282472015e-02 &       Inf   &    1 &     0 &       0       24        0 \\
  1 &  8.6591166282472015e-02 &  0.0e+00    &    1 &     0 &       0       24        0 \\
.
.
.
kappa = 1.0000e-04
Mesh data: refinements = 8, dim Y_h = 237368, dim U_h = 239526.
SemiSmooth Newton method. Using multipliers and solving nonlinear equations
  j &       J(u)             &   delta_u    & Fact &    CG &    #J   &  #Ab &    #Aa \\
____ _______________________ _____________ ______ ______ ________ ______ ________
  0 &  1.5050391203894268e-03 &       Inf   &    2 &     0 &  85902  138602   15013 \\
  1 &  1.5050370629104556e-03 &  4.2e-02    &    3 &    22 &  85901  138612   15013 \\
  2 &  1.5050374656000616e-03 &  4.3e-02    &    2 &    29 &  85643  138721   15162 \\
  3 &  1.5050374665947280e-03 &  9.7e-03    &    2 &    32 &  85434  138910   15182 \\
  4 &  1.5050374665958775e-03 &  1.4e-09    &    3 &    25 &  85514  138830   15182 \\
  5 &  1.5050374665958525e-03 &  2.8e-11    &    2 &    45 &  85469  138875   15182 \\
  6 &  1.5050374665958462e-03 &  3.3e-13    &    1 &    45 &  85571  138773   15182 \\
  7 &  1.5050374665958462e-03 &  1.2e-14    &    1 &    50 &  85531  138813   15182 \\

tiempo =

  51.489184500000000
```

Some results: (4a) Matlab fmincon. Mass lumping.

Trust region method. Provide HessiaMultiply with a smart trick to pass a diagonal preconditioner.

Use zero initial point and let Matlab do all the work.

```
kappa = 1.0000e-04
Mesh data: refinements = 8, dim Y_h = 237368, dim U_h = 239526.
Matlab fmincon

                                  Norm of      First-order
 Iteration          f(x)          step          optimality   CG-iterations
     0           0.0854467                       1.01e-05
     1           0.0124817         1.55336       1.62e-06           14
     2          0.00466646        0.715332       2.69e-07           19
     3          0.00310525        0.954632       1.49e-07           19
     4          0.00282616         0.22572       1.46e-07           19
     5          0.00180709         1.35091       3.25e-08           18
     6           0.0015767        0.871314       6.25e-09           20
     .
    32          0.00150509     5.07681e-14       6.06e-14            0
    33          0.00150509      1.2692e-14       6.06e-14            0

Optimization stopped because the norm of the current step, 1.269202e-14, is
less than options.StepTolerance = 5.000000e-14.


tiempo =

     1.376478881000000e+02
```

Some results: (4b) Matlab fmincon. Mass lumping.

Trust region method. Provide HessiaMultiply with a smart trick to pass a diagonal preconditioner. Use a –refinement in h and continuation in $\kappa-$ technique

```
kappa = 1.0000e+00
Mesh data: refinements = 1, dim Y_h = 8, dim U_h = 24.
Matlab fmincon
                                    Norm of      First-order
 Iteration        f(x)              step          optimality   CG-iterations
     0          0.0865912                           0.0705
     1          0.0859775        0.0179072        0.000245           4
     2          0.0859775       6.6269e-05        6.02e-09           4
     3          0.0859775      1.38571e-09        2.59e-17           4

Optimization completed: The first-order optimality measure, 2.592113e-17,
is less than options.OptimalityTolerance = 5.000000e-14, and no negative/zero
curvature is detected in the trust-region model.
 .
 .
kappa = 1.0000e-04
Mesh data: refinements = 8, dim Y_h = 237368, dim U_h = 239526.
Matlab fmincon
                                    Norm of      First-order
 Iteration        f(x)              step          optimality   CG-iterations
     0         0.00150509                           5.4e-10
     1         0.00150509      0.000325074          3.8e-10          21
     2         0.00150509        0.0209348        6.64e-12          21
     3         0.00150509        0.0115229        9.45e-13          20
     4         0.00150509       0.00591895        9.45e-13          21
     .
     .
    20         0.00150509      2.31679e-13        9.45e-13           0
    21         0.00150509      5.79199e-14        9.45e-13           0
    22         0.00150509        1.448e-14        9.45e-13           0

Optimization stopped because the norm of the current step, 1.447997e-14, is
less than options.StepTolerance = 5.000000e-14.

tiempo =
  66.902753500000003
```

Some results: (5) Matlab fmincon. No Mass Lumping

Trust region method. Provide HessiaMultiply with a smart trick to pass a diagonal preconditioner. Use a –refinement in h and continuation in κ– technique

```
kappa = 1.0000e+00
Mesh data: refinements = 1, dim Y_h = 8, dim U_h = 24.
Matlab fmincon
                                  Norm of      First-order
 Iteration        f(x)            step          optimality   CG-iterations
     0          0.0865912                         0.0705
     1          0.0859153       0.0192294          0.011          12
     2          0.0858572        0.101693       2.95e-05          12
     .
     .
    16          0.0858497     0.000752885        1.6e-14          12

Optimization completed: The first-order optimality measure, 1.597934e-14,
is less than options.OptimalityTolerance = 5.000000e-14, and no negative/zero
curvature is detected in the trust-region model.
.
.
kappa = 1.0000e-04
Mesh data: refinements = 8, dim Y_h = 237368, dim U_h = 239526.
Matlab fmincon
                                  Norm of      First-order
 Iteration        f(x)            step          optimality   CG-iterations
     0         0.00150504                       7.52e-10
     1         0.00150504     0.000323823       4.83e-10          51
     2         0.00150504     0.000580233       1.29e-10          50
     3         0.00150504     6.11812e-05       1.29e-10          50
     4         0.00150504     1.52953e-05       1.29e-10           0
     .
     .
    19         0.00150504     1.42449e-14       1.29e-10           0

Optimization stopped because the norm of the current step, 1.424486e-14, is
less than options.StepTolerance = 5.000000e-14.

tiempo =
  81.967342200000004
```