INIS for Optimization of PDE

Justin Pearse-Danker

Systems Control and Optimization laboratory

Master thesis presentation
March 23, 2020

Motivation

Goal:

- Implement an efficient solver for PDE constrained optimal control problems (OCP) with boundary controls.

Motivation

Goal:

- Implement an efficient solver for PDE constrained optimal control problems (OCP) with boundary controls.

$\Rightarrow$ Combine the MG method with the INIS method $\rightarrow$ INIS-MG Algorithm.

Motivation

Goal:

- Implement an efficient solver for PDE constrained optimal control problems (OCP) with boundary controls.

$\Rightarrow$ Combine the MG method with the INIS method $\rightarrow$ INIS-MG Algorithm.

Why PDE constrained OCPs:

- Relevant in the context of industrial and medical applications
  - optimal cooling of steel profiles
  - optimal local heating of tumor tissue

Motivation

Goal:

- Implement an efficient solver for PDE constrained optimal control problems (OCP) with boundary controls.

$\Rightarrow$ Combine the MG method with the INIS method $\rightarrow$ INIS-MG Algorithm.

Why PDE constrained OCPs:

- Relevant in the context of industrial and medical applications
  - optimal cooling of steel profiles
  - optimal local heating of tumor tissue

Why use the combination of INIS and MG:

- Boundary Controls
- PDE constraints

Problem Formulation

$$\begin{aligned} \min_{z\in\mathbb{R}^{n_\mathrm{z}},\, w\in\mathbb{R}^{n_\mathrm{w}}} \quad & f(z,w), \\ \text{subject to} \quad & g(z,w) = 0. \end{aligned}$$

- $g : \mathbb{R}^{n_\mathrm{z}} \times \mathbb{R}^{n_\mathrm{w}} \to \mathbb{R}^{n_g}$,
- $n_\mathrm{z} = n_g$,
- Jacobian $g_z(\cdot)$ invertible.
- $y = [z^\top, w^\top]^\top$

Forward Problem

$$g(z, w) = 0$$

Assumptions:

- $n_{\mathrm{z}} = n_g$,
- Jacobian $g_z(\cdot)$ is invertible.

Forward Problem

$$g(z, w) = 0$$

Assumptions:

- $n_{\mathrm{z}} = n_g$,
- Jacobian $g_z(\cdot)$ is invertible.

$\Rightarrow$ The variables z are implicitly defined as function of w.

Forward Problem

For a given w^* solve

$$g(z, w^*) = 0$$

with Newton's method:

- Current iterate z^k,
- $\Delta z^k = -g_z(z^k, w^*)^{-1} g(z^k, w^*)$,
- $z^{k+1} = z^k + \Delta z^k$.

Forward Problem

Use full-rank approximation

$$M \approx g_z$$

for an inexact Newton method:

- Current iterate z^k,
- $\Delta z^k = -M^{-1} g(z^k, w^*)$,
- $z^{k+1} = z^k + \Delta z^k$.

In order to solve the whole NLP we can apply a SQP method:

- Current iterate (y^k, λ^k)
- Solve following QP:

$$\begin{aligned} \min_{\Delta y \in \mathbb{R}^{n_y}} \quad & \frac{1}{2} \Delta y^\top \tilde{H} \Delta y + \nabla_y \mathcal{L}(y^k, \lambda^k) \Delta y \\ \text{subject to} \quad & g_z(y^k) \Delta z + g_w(y^k) \Delta w + g(y^k) = 0. \end{aligned}$$

- $y^{k+1} = y^k + \Delta y$ and $\lambda^{k+1} = \lambda^k + \Delta \lambda^k$.

Forward Problem

SQP method

In order to solve the whole NLP we can apply a SQP method:

- Current iterate (y^k, λ^k)
- Solve following QP:

$$\begin{aligned} \min_{\Delta y \in \mathbb{R}^{n_y}} \quad & \frac{1}{2}\Delta y^\top \tilde{H} \Delta y + \nabla_y \mathcal{L}(y^k, \lambda^k)\Delta y \\ \text{subject to} \quad & M\Delta z + g_w(y^k)\Delta w + g(y^k) = 0. \end{aligned}$$

- $y^{k+1} = y^k + \Delta y$ and $\lambda^{k+1} = \lambda^k + \Delta\lambda^k$.

Question: Is there a connection between the contraction of inexact Newton method applied to the forward problem and the contraction of the inexact method of the NLP?

Question: Is there a connection between the contraction of inexact Newton method applied to the forward problem and the contraction of the inexact method of the NLP?

Answer: No, there are examples, where the inexact Newton method of the forward problem converges, but the inexact method applied to the whole NLP with the same approximation M diverges.

Introduce sensitivity matrix $D \in \mathbb{R}^{n_\mathrm{z} \times n_\mathrm{w}}$ which is implicitly defined by the equation

$$g_z(y)D - g_w(y) = 0.$$

Introduce sensitivity matrix $D \in \mathbb{R}^{n_\mathrm{z} \times n_\mathrm{w}}$ which is implicitly defined by the equation

$$g_z(y)D - g_w(y) = 0.$$

Applying Newton's method yields:

- Current iterate D^k,
- $\Delta D^k = -M^{-1}(g_z(y^k)D^k - g_w(y^k))$,
- $D^{k+1} = D^k + \Delta D^k$.

With the approximation

$$MD^k \approx g_w(y^k),$$

the SQP method solves the QP:

$$\begin{aligned} \min_{\Delta y \in \mathbb{R}^{n_\mathrm{y}}} \quad & \frac{1}{2}\Delta y^\top \tilde{H} \Delta y + \nabla_y \mathcal{L}(y^k, \lambda^k) \Delta y \\ \text{subject to} \quad & M \Delta z + {\color{red} MD^k} \Delta w + g(y^k) = 0. \end{aligned}$$

Contraction rate of INIS method:

$$\kappa^*_{\text{INIS}} = \max\left(\kappa^*_F, \rho\left(\tilde{H}_Z^{-1} H_Z - \mathbb{1}_{n_\text{w}}\right)\right)$$

Contraction rate of INIS method:

$$\kappa_{\mathrm{INIS}}^* = \max\left(\kappa_F^*, \rho\left(\tilde{H}_Z^{-1} H_Z - \mathbb{1}_{n_\mathrm{w}}\right)\right)$$

- Local contraction of the forward problem is a necessary condition for local contraction of the INIS algorithm.
- Sufficient Condition, if the Hessian approximation is good enough.

Poisson equation:

$$\begin{aligned} -\Delta z &= f \quad t \in \Omega = (0,1)^2, \\ z &= 0 \quad t \in \partial\Omega. \end{aligned}$$

with $f : \Omega \to \mathbb{R}$.

Finite differences discretization of the Laplacian:

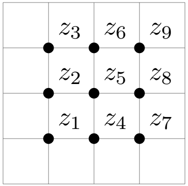
$$-\partial_{t_1}^{+}\partial_{t_1}^{-}z_{i,j} - \partial_{t_2}^{+}\partial_{t_2}^{-}z_{i,j} = -h^{-2}(z_{i,j-1} + z_{i-1,j} - 4z_{i,j} + z_{i+1,j} + z_{i,j+1})$$

Finite differences discretization of the Laplacian:

$$-\partial_{t_1}^{+}\partial_{t_1}^{-} z_{i,j} - \partial_{t_2}^{+}\partial_{t_2}^{-} z_{i,j} = -h^{-2}(z_{i,j-1} + z_{i-1,j} - 4z_{i,j} + z_{i+1,j} + z_{i,j+1})$$

Lexicographic enumeration of interior points:

$$(i,j) \equiv i + (j-1)(J-1) = m$$



.

Reduced linear system:

$$\underbrace{h^{-2}\begin{bmatrix} X & -\mathbb{1} & & \\ -\mathbb{1} & \ddots & \ddots & \\ & \ddots & \ddots & -\mathbb{1} \\ & & -\mathbb{1} & X \end{bmatrix}}_{=:A} \underbrace{\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_N \end{bmatrix}}_{=:Z} = \underbrace{\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{bmatrix}}_{=:F},$$

with

$$X = \begin{bmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix}$$

.

Sequence of grid sizes

$$h_0 > h_1 > \ldots > h_l > \ldots > h_L \quad \text{with} \;\; h_l = 2^{-l-1}$$

for a given $L > 0$.

Corresponding interior grid:

$$\Omega_l = \{(ih_l, jh_l) : 1 \leq i, j \leq J_l\},$$

with $J_l = h_l^{-1} - 1$.

Goal: Reduce high frequent part of the error $e_l = Z_l - Z_l^*$

Richardson iteration:

$$Z_l^k = Z_l^{k-1} - \omega(A_l Z_l^{k-1} - F_l),$$

with $\omega \in (0, 2/\xi_{\max})$.

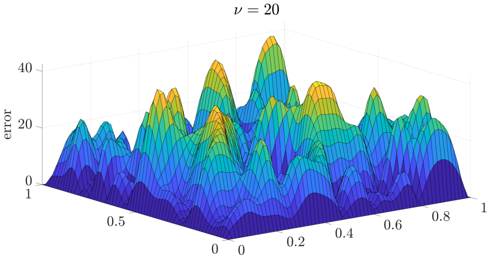


Figure: Error $e_l^\nu = |Z_l^\nu - Z_l^*|$ after $\nu = 20$ Richardson iterations.

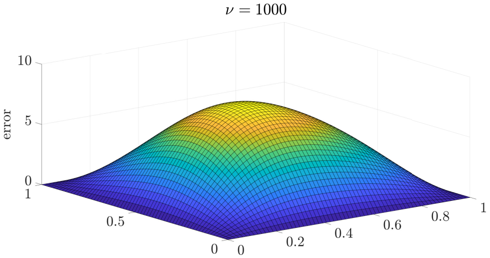


Figure: Error $e_l^\nu = |Z_l^\nu - Z_l^*|$ after $\nu = 200$ Richardson iterations.

With the residuum $r_l = A_l Z_l^\nu - F_l$ we can formulate the *defect problem*

$$A_l d_l = r_l,$$

with its "smooth" solution $d_l^* = Z_l^\nu - Z_l^*$.

$\Rightarrow$ d_l^* can be approximated on a coarse grid better than Z_l^*.

Restriction operator:

$$R_l : \mathbb{R}^{J_l^2} \to \mathbb{R}^{J_{l-1}^2}$$
$$r_l \mapsto R_l \, r_l .$$

Prolongation operator:

$$P_l : \mathbb{R}^{J_{l-1}^2} \to \mathbb{R}^{J_l^2}$$
$$d_{l-1} \mapsto P_l \, d_{l-1} = R_l^\top d_{l-1} .$$

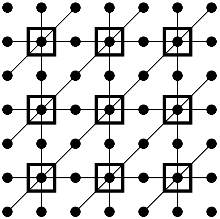


Figure: Restriction and prolongation for gridlevel $l = 3$

Coarse grid correction:

$$Z_l^\nu \mapsto Z_l^\nu - P_l A_{l-1}^{-1} R_l (A_l Z_l^\nu - F_l),$$

Coarse grid correction:

$$Z_l^\nu \mapsto Z_l^\nu - P_l A_{l-1}^{-1} R_l (A_l Z_l^\nu - F_l),$$

Algorithm 2: `two_grid`$(A_l, F_l, Z_l^0, \omega, \nu)$

$Z_l^\nu =$ `richardson`$(A_l, F_l, Z_l^0, \omega, \nu)$ `// smoothing inital guess`
$r_l = A_l Z_l^\nu - F_l$ `// calculation of the residuum`
$r_{l-1} = R_l r_l$ `// restriction of the residuum`
$d_{l-1} = A_{l-1}^{-1} r_{l-1}$ `// exact solution of the coarse-grid equation`
$Z_l = Z_l^\nu - R_l^\top d_{l-1}$ `// correction step`
return Z_l

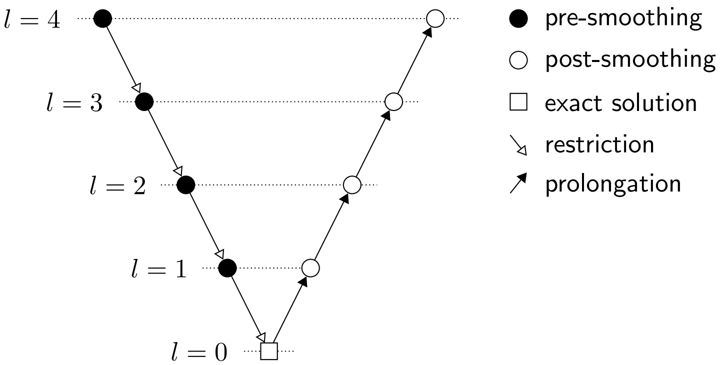


Figure: Graphical illustration of the recursive MG strategy.

Lemma (Linearity of the V-cycle)

The mapping φ_l is linear in Z_l and F_l, i.e. for $l \geq 0$ there exist matrices $S_l^{\mathrm{MG}}, T_l^{\mathrm{MG}} \in \mathbb{R}^{J_l^2 \times J_l^2}$ such that

$$\varphi_l(Z_l, F_l) = S_l^{\mathrm{MG}} Z_l + T_l^{\mathrm{MG}} F_l$$

for all $Z_l, F_l \in \mathbb{R}^{J_l^2}$. For $l = 0$ these matrices are

$$S_l^{\mathrm{MG}} = \mathbb{0},$$
$$T_l^{\mathrm{MG}} = A_l^{-1}$$

and for $l > 0$ they are recursively defined as

$$S_l^{\mathrm{MG}} = S_l^{\nu_{\mathrm{post}}}(S_l^{\nu_{\mathrm{pre}}} + R_l^T T_{l-1}^{\mathrm{MG}} R_l A_l S_l^{\nu_{\mathrm{pre}}}),$$
$$T_l^{\mathrm{MG}} = S_l^{\nu_{\mathrm{post}}}(T_l^{\nu_{\mathrm{pre}}} + R_l^T (T_{l-1}^{\mathrm{MG}} R_l A_l T_l^{\nu_{\mathrm{pre}}} - T_{l-1}^{\mathrm{MG}} R_l)) + T_l^{\nu_{\mathrm{post}}}.$$

INIS-Multi-Grid (INIS-MG) for Optimization PDE

PDE Constrained Optimal Control Test Problem

$$
\begin{aligned}
\underset{z(\cdot),\,u(\cdot)}{\text{minimize}} \quad & \frac{1-\alpha}{2}\int_\Omega \left\|z - f_{\text{ref}}^{\gamma}\right\|^2 \mathrm{d}t + \frac{\alpha}{2}\int_{\partial\Omega} \|u\|^2 \,\mathrm{d}s, \\
\text{subject to} \quad & -\Delta z = \beta z^3 \quad t \in \Omega = (0,1)^2, \\
& u \in \mathcal{C}(\partial\Omega), \\
& u|_{\partial\Omega_i} = u_i \quad i = 1,\ldots,4\,, \\
& u_i \in \mathscr{P}_5(\partial\Omega_i) \quad i = 1,\ldots,4\,, \\
& z|_{\partial\Omega_i} = u_i \quad i = 1,\ldots,4\,,
\end{aligned}
$$

with $\beta \in \mathbb{R}$ and $\alpha \in [0,1]$.

PDE Constrained Optimal Control Test Problem

$$f_{\text{ref}}^{\gamma}(t) = \begin{cases} \gamma & \text{for } t \in [0.2, 0.3]^2, \\ 0 & \text{otherwise,} \end{cases}$$

with $\gamma \in \mathbb{R}$.

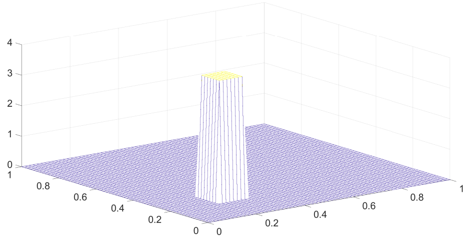


Figure: Reference function $f_{\text{ref}}^{\gamma}(\cdot)$ with $\gamma = 4$.

$$u_i(t) = \sum_{j=0}^{5} w_i^j t^j \quad \text{for } i = 1, 2$$

$$u_i(t) = \sum_{j=0}^{5} w_i^j (1-t)^j \quad \text{for } i = 3, 4$$

with $t \in [0, 1]$

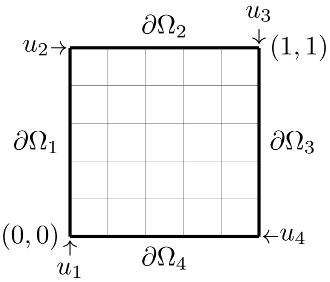


Figure: Discretization of Ω with uniform grid and boundary polynomials u_i for $i = 1, \ldots, 4$.

Eliminating boundary states:

$$z_{i,0} := u_1(ih) \qquad z_{i,J} := u_3(ih)$$
$$z_{J,j} := u_2(jh) \qquad z_{0,j} := u_4(jh)$$

for $i, j = 0, \ldots, J$.

INIS-Multi-Grid (INIS-MG) for Optimization PDE

Equality Constraints

$$\underbrace{\begin{bmatrix} X^1_{L,\beta}[Z_L] & -\mathbb{1} & & \\ -\mathbb{1} & \ddots & \ddots & \\ & \ddots & \ddots & \mathbb{1} \\ & & -\mathbb{1} & X^I_{L,\beta}[Z_L] \end{bmatrix}}_{=:A_{L,\beta}[Z_L]} \underbrace{\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_{N-1} \\ z_N \end{bmatrix}}_{=:Z_L} = \underbrace{\begin{bmatrix} d_1[w] \\ d_2[w] \\ \vdots \\ d_{I-1}[w] \\ d_I[w] \end{bmatrix}}_{b_L[w]}$$

with

$$X^i_{L,\beta}[Z_L] = \begin{bmatrix} 4 - h^2\beta z^2_{(i-1)I+1} & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 - h^2\beta z^2_{iI} \end{bmatrix}$$

for $i = 1, \ldots, I$.

INIS-Multi-Grid (INIS-MG) for Optimization PDE

Reduced NLP

$$\begin{aligned} \underset{Z_L \in \mathbb{R}^N,\, w \in \mathbb{R}^{n_\mathrm{w}}}{\text{minimize}} \quad & \frac{1-\alpha}{2} h^2 \sum_{i=1}^{N} (Z_L^i - f_{\mathrm{ref}}^{\gamma}(t_i))^2 + \frac{\alpha}{2} h \sum_{i=1}^{4} \sum_{j=0}^{J} u_i(jh)^2 \\ \text{subject to} \quad & A_{L,\beta}[Z_L] Z_L = b_L[w] \end{aligned}$$

INIS-Multi-Grid (INIS-MG) for Optimization PDE

Jacobian Approximation

Constraint Jacobian:

$$g_{Z_L}(Z_L, w) = \begin{bmatrix} \tilde{X}^1_{L,\beta}[Z_L] & -\mathbb{1} & & \\ -\mathbb{1} & \ddots & \ddots & \\ & \ddots & \ddots & -\mathbb{1} \\ & & -\mathbb{1} & \tilde{X}^I_{L,\beta}[Z_L] \end{bmatrix}$$

with

$$\tilde{X}^i_{L,\beta}[Z_L] = \begin{bmatrix} 4 - 3h^2\beta z^2_{i(I)+1} & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 - 3h^2\beta z^2_{iI+I} \end{bmatrix}$$

Jacobian Approximation:

$$M := g_{Z_L}(\mathbb{0}, w) \approx g_{Z_L}(Z_L, w).$$

INIS-Multi-Grid (INIS-MG) for Optimization PDE

Algorithm

Algorithm 3: $\texttt{INIS\_MG}(M, D, y_L^k, \Delta z^0, \Delta\lambda^0, \Delta D^0, L)$

$$\Delta\bar{z} = -\texttt{multi\_grid}(M, g(y_L^k), \Delta z^0, L)$$
$$b = Z^\top \nabla_y \mathcal{L}(y_L^k, \lambda_L^k) - Z^\top \tilde{H} \begin{bmatrix} \Delta\bar{z} \\ \mathbb{0} \end{bmatrix}$$
$$\Delta w = -(Z^\top \tilde{H} Z)^{-1} b$$
$$\Delta z = \Delta\bar{z} - D^k \Delta w$$
$$b = [\mathbb{1}_N \quad \mathbb{0}] \left(\nabla_y \mathcal{L}(y_L^k, \lambda_L^k) + \tilde{H} \Delta y \right)$$
$$\Delta\lambda = -\texttt{multi\_grid}(M^\top, b, \Delta\lambda^0, L)$$
$$y_L^{k+1} = y_L^k + (\Delta z^\top, \Delta w^\top)^\top$$
$$\lambda^{k+1} = \lambda^k + \Delta\lambda$$
$$B = g_z(y_L^k) D^k - g_w(y_L^k)$$
$$\Delta D = -\texttt{multi\_grid}(M, B, \Delta D^0, L)$$
$$D^{k+1} = D^k + \Delta D$$

Algorithm 4: $\texttt{INIS\_MG}(M, D, y_L^k, \Delta z^0, \Delta\lambda^0, \Delta D^0, L)$

$\Delta\bar{z} = -\texttt{multi\_grid}(M, g(y_L^k), \Delta z^0, L)$

$b = Z^\top \nabla_y \mathcal{L}(y_L^k, \lambda_L^k) - Z^\top \tilde{H} \begin{bmatrix} \Delta\bar{z} \\ \mathbb{0} \end{bmatrix}$

$\Delta w = -(Z^\top \tilde{H} Z)^{-1} b$

$\Delta z = \Delta\bar{z} - D^k \Delta w$

$b = [\mathbb{1}_N \quad \mathbb{0}] \left(\nabla_y \mathcal{L}(y_L^k, \lambda_L^k) + \tilde{H}\Delta y \right)$

$\Delta\lambda = -\texttt{multi\_grid}(M^\top, b, \Delta\lambda^0, L)$

$y_L^{k+1} = y_L^k + (\Delta z^\top, \Delta w^\top)^\top$

$\lambda^{k+1} = \lambda^k + \Delta\lambda$

$B = g_z(y_L^k) D^k - g_w(y_L^k)$

$\Delta D = -\texttt{multi\_grid}(M, B, \Delta D^0, L)$

$D^{k+1} = D^k + \Delta D$

- Laptop running Windows 10 equipped with an Intel i7.8565U and 16GB of RAM.
- `MATLAB`
- `CasADi`
  - Computation of Jacobians and Hessians.
- `ipopt`
  - State of the art large-scale NLP solver.

$$\begin{aligned} \underset{Z_L \in \mathbb{R}^N,\, w \in \mathbb{R}^{n_\mathrm{w}}}{\text{minimize}} \quad & \frac{1-\alpha}{2} h^2 \sum_{i=1}^{N} (Z_L^i - f_{\mathrm{ref}}^{\gamma}(t_i))^2 + \frac{\alpha}{2} h \sum_{i=1}^{4} \sum_{j=0}^{J} u_i(jh)^2 \\ \text{subject to} \quad & A_{L,\beta}[Z_L] Z_L - b_L[w] = 0 \end{aligned}$$

NLP parameters:

$$\alpha = 0.5, \quad \beta = 80, \quad \gamma = 4.$$

MG parameters:

$$\nu_{\mathrm{pre}} = 2, \quad \nu_{\mathrm{post}} = 2, \quad l_{\min} = 0$$

Numerical Experiments
Controls

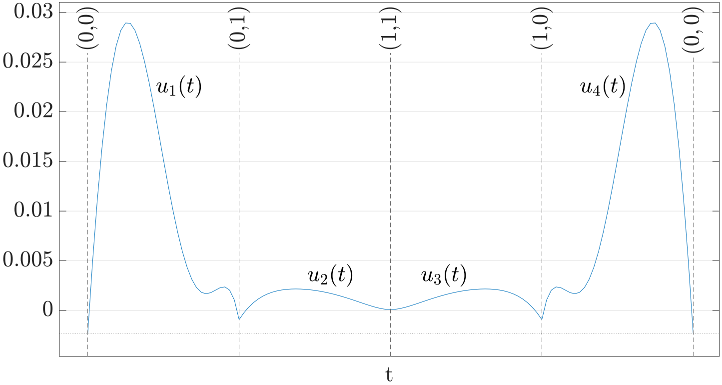


Figure: Polynomials $u_1(\cdot), \ldots, u_4(\cdot)$ with coefficients $w^*_{\texttt{ipopt}}$.

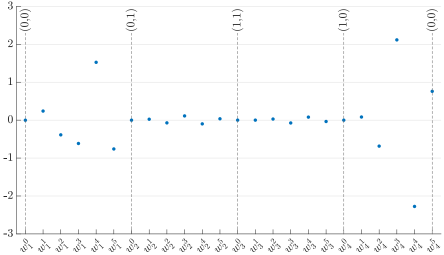


Figure: Plot of the expanded coefficients $w^*_{\texttt{ipopt}}$.

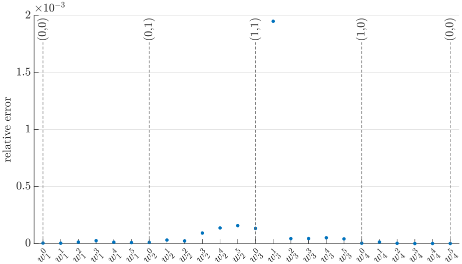


Figure: Relative error $e^{\text{rel}}(w^*_{\texttt{ipopt}}, w^*_{\text{INIS}})$ of expanded coefficients w^*_{INIS}.

Numerical Experiments

States

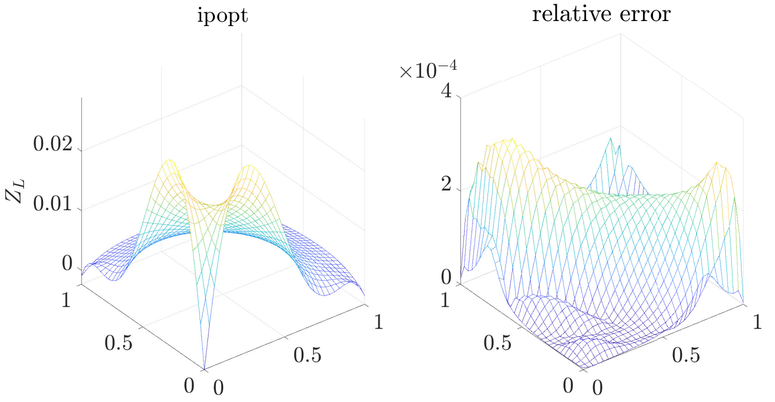


Figure: Plot of the expanded solution $Z^*_{\mathtt{ipopt}}$ for gridlevel $L = 5$ and the relative error $e^{\mathrm{rel}}(Z^*_{\mathtt{ipopt}}, Z^*_{\mathrm{INIS}})$.

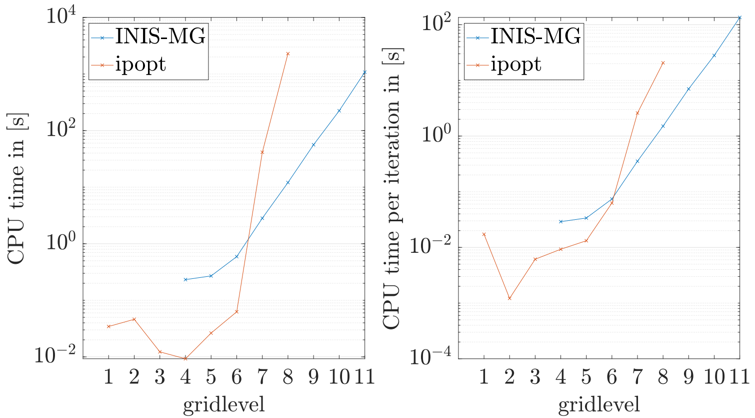


Figure: CPU time to compute NLP solution with INIS-MG and `ipopt` on different gridlevels.

Numerical Experiments

Number of Iterations

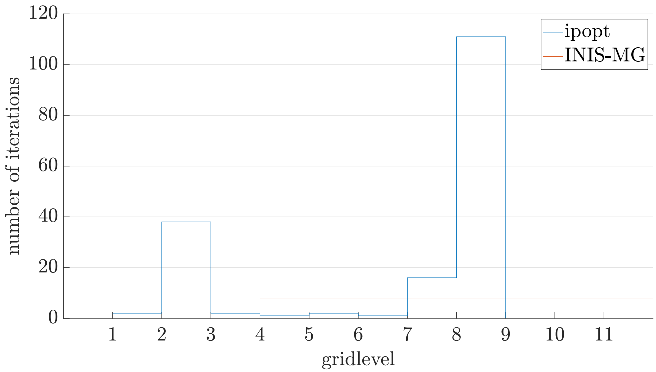


Figure: Number of iterations needed for convergence of the algorithm `ipopt` and INIS-MG solving the test problem on gridlevels $L = 1, \ldots, 11$.

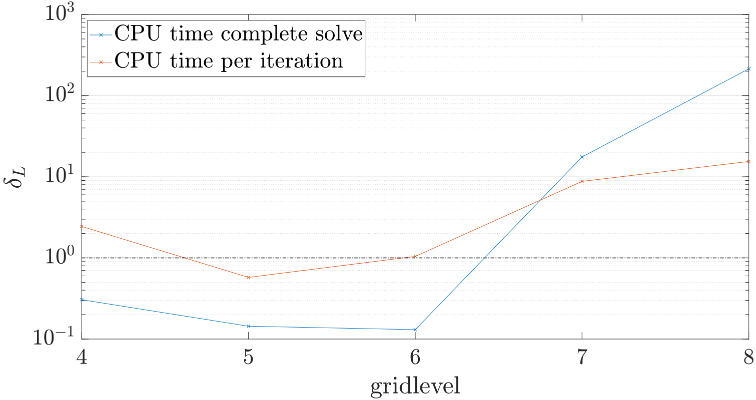


Figure: Factor $\delta_L = t_L^{\text{ipopt}} / t_L^{\text{INIS-MG}}$ on gridlevels $L = 4, \ldots, 8$

Numerical Experiments

Local Contraction of Forward Problem and INIS-MG

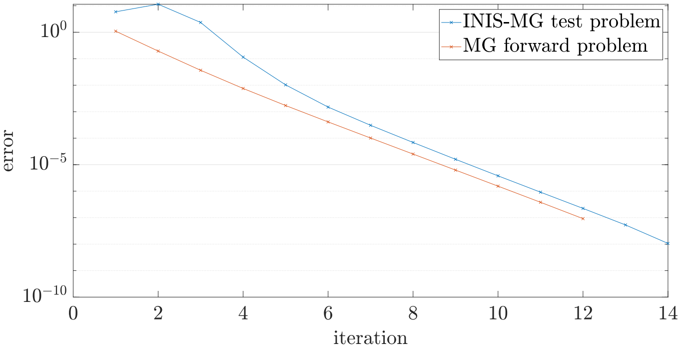


Figure: Plot of the error $|y^k - y^*|$ for the iterates of the INIS-MG method and the forward problem performed on gridlevel $L = 7$.

Numerical Experiments

Local Contraction of Forward Problem and INIS-MG

Contraction rate via slope:

$$\kappa^*_{\text{INIS-MG}} \approx \exp(-1.4446) = 0.2358$$
$$\kappa^*_F \approx \exp(-1.4019) = 0.2461$$

Contraction rate via definition:

$$\kappa^*_F = \rho(A_{L,\beta}[0]^{-1} g_z - \mathbb{1}_{n_z}) = 0.28232$$

Conclusion

- The INIS-MG method preserves the local contraction properties of the INIS method.
- INIS-MG method outperformed `ipopt` by a factor up to 200.

Outlook

- Extend the presented INIS-MG method with respect to
  - more general PDEs
  - inequality constraints
  - 3-dimensional problems
- Investigate different versions, such as a version with an inexact hessian.
- Combine the MG method with the IN method.