TEMPO Summer School on Numerical Optimal Control and Embedded Optimization
University of Freiburg, July 27 - August 7, 2015

Exercise 12: Splitting Methods

Giorgos Stathopolous Colin N. Jones

http://syscop.de/teaching/numerical-optimal-control/

The inverted pendulum - Controlled via AMA Consider again the problem of stabilizing the inverted pendulum in the upright position, as discussed in Ex.1 of the course. The system was linearized around the desired equilibrium and an LQR controller applied. Consider now the actuator constraints $-3.25 \leq u \leq 0.75$.

In this exercise, you will use the AMA algorithm to solve the problem in a rolling horizon fashion with sampling time $T_s = 0.05s$, a total simulation time of $T = 5s$ and a horizon of $N = 40$.

The problem to solve at each time step has the form

$$\begin{array}{ll} \text{minimize} & (z - z_s)^T H (z - z_s) \\ \text{subject to} & Cz = d \\ & -Lz + l \geq 0 \ , \end{array} \tag{1}$$

where $z = (x_0, x_1, \ldots, x_N, u_0, \ldots, u_{N-1}) \in \mathbb{R}^{N(n+m)+n}$, $H = \mathbf{diag}(Q, \ldots, Q, P, R, \ldots, R)$ and $z_s = (x_s, u_s)$ the zero equilibrium.

1. Define the functions f and g and the matrix A and vector b so that you can write problem (1) in the standard form

$$\begin{aligned} &\min\ f(z) + g(y) \\ &\text{s.t. } Az + b = y \end{aligned}$$

   where f is strongly convex. Hint : Embed $Cz = d$ into f.

2. Derive the expressions required to compute each step of the AMA algorithm

$$\begin{aligned} x^{k+1} &= \arg\min_x \ f(x) + \langle \lambda^k, Ax \rangle \\ y^{k+1} &= \mathrm{prox}_{g,\rho}\left(Ax^{k+1} + b + \lambda^k/\rho\right) \\ \lambda^{k+1} &= \lambda^k + \rho(Ax^{k+1} + b - y^{k+1}) \end{aligned}$$

3. Complete the file `templateAMA.m` with the three steps of the algorithm above. Run `mainPendulum` to run the simulation. Notice that during transitions the algorithm takes more than the allowed 150 iterations.

4. Extend your AMA algorithm to utilize Nesterov's acceleration

$$\begin{aligned} x^{k+1} &= \arg\min_x \ f(x) + \langle \hat{\lambda}^k, Ax \rangle \\ y^{k+1} &= \mathrm{prox}_{g,\rho}\left(Ax^{k+1} + b + \hat{\lambda}^k/\rho\right) \\ \lambda^k &= \hat{\lambda}^k + \rho(Ax^{k+1} + b - y^{k+1}) \\ \hat{\lambda}^{k+1} &= \lambda^k + ((\alpha^k - 1)/\alpha^{k+1})(\lambda^k - \lambda^{k-1}) \end{aligned}$$

   Calling `mainPendulum('acceleration',true)` will enable the computation of the α sequence above (called β in the code), and will utilize a restarting scheme that has been written for you. Simply change the `lam` variables in your AMA code to `hat.lam` where appropriate. Compare the number of iterations taken by the algorithms - do you see a difference?

5. Develop a preconditioned matrix by modifying the second step of the algorithm to

$$y^{k+1} = E\,\mathrm{prox}^{P}_{g,\rho}(E^{-1}(A_d x^{k+1} + b + \lambda^k/\rho) \tag{2}$$

where the matrix E is available in the code as `prec.E`.

Run a preconditioned version of the algorithm by calling

```
mainPendulum('acceleration',true,'preCondition',true)
```

and compare to the first two versions.