AWESCO Winter School on Numerical Optimal Control with DAE - University of Freiburg

# Exercise 5: Sequential Quadratic Programming

Andrea Zanelli, Gianluca Frison, Elena Malz,
Joel Andersson, Joris Gillis, Sebastien Gros, Moritz Diehl

In the exercises so far, we solved the NLPs with IPOPT which is a popular open-source primal-dual interior point solver. Other NLP solvers can be used from CasADi including SNOPT, WORHP and KNITRO. In the following, we will write our own simple NLP solver implementing sequential quadratic programming (SQP).

Starting from a given initial guess for the primal and dual variables $(x^0, \lambda^0)$, SQP solves the NLP by iteratively computing local convex quadratic approximations of the NLP at the current iterate $\left(x^k, \lambda^k\right)$ and solving them by using a convex quadratic programming (QP) solver. For an NLP of the form:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & \underline{x} \le x \le \overline{x}, \\ & \underline{g} \le g(x) \le \overline{g}, \end{aligned} \tag{1}$$

these approximations take the form:

$$\begin{aligned} \min_{\Delta x} \quad & \frac{1}{2}\Delta x^T \nabla_x^2 \mathcal{L}(x^k, \lambda^k)\Delta x + \nabla_x f(x^k)^T \Delta x \\ \text{s.t.} \quad & \underline{x} \le x^k + \Delta x \le \overline{x}, \\ & \underline{g} \le g(x^k) + \nabla g(x^k)^T \Delta x \le \overline{g}, \end{aligned} \tag{2}$$

where $(x^k, \lambda^k)$ is the current approximation of the primal-dual solution to the NLP in (4) and $\mathcal{L}(x, \lambda) := f(x) + \lambda_g^T g(x) + \mu^T(x - \overline{x}) + \nu^T(\underline{x} - x)$ is the Lagrangian. The solution of this QP gives the step $\Delta x$ and new approximation of the multipliers $(\lambda, \mu, \nu)$.

For nonlinear least square objectives of the form $f(x) = \frac{1}{2}\|R(x)\|_2^2$, a popular variant is to use the so called *Gauss-Newton* approximation of the Hessian of the Lagrangian:

$$\nabla_x^2 \mathcal{L}(x^k, \lambda^k) \approx \nabla R(x^k) \nabla R(x^k)^T. \tag{3}$$

In this exercise we will consider the optimization problem

$$\begin{aligned} \min_{x} \quad & f(x) := \frac{1}{2}(x_1 - 1)^2 + \frac{1}{2}(10(x_2 - x_1^2))^2 + \frac{1}{2}x_2^2 \\ \text{s.t.} \quad & g(x) := x_1 + (1 - x_2)^2 = 0 \\ & h(x) := 0.2 + x_1^2 - x_2 \le 0 \end{aligned} \tag{4}$$

Tasks:

5.1 Re-write on paper the objective function in nonlinear least-square form $F(x) = \frac{1}{2} \|R(x)\|_2^2$ and derive the Gauss-Newton approximation of the Hessian of the Lagrangian.

5.2 We will start by implementing an SQP solver for the unconstrained problem obtained by removing both $g$ and $h$ from (4). Using the template provided with this exercise, implement the CasADi functions `f` and `Jf` that return evaluations of $f$ and its Jacobian. Use the numerical values given in the template to check that your implementation is correct. Do the same for the residual function $R$ and its Jacobian.

5.3 Using the Jacobian of $f$ and $R$ build the Gauss-Newton objective function

$$f_{gn} = \frac{1}{2}\Delta x^T \nabla R(x^k) \nabla R(x^k)^T \Delta x + \nabla_x f(x^k)^T \Delta x.$$

Then, allocate an instance of the QP solver qpOASES using CasADi and use it to solve the local quadratic approximations in the SQP iterations. Plot the results using the template. Where do the iterates converge to?

5.4 Include now the equality constraints. Define two CasADi functions `G` and `Jg` that return evaluations of $g$ and its Jacobian and use them to define the linearized equality constraint

$$g_l = g(x^k) + \nabla g_x(x^k)^T \Delta x.$$

Include this constraint in the QP formulation and run the simulation again. Does the solution change?

5.5 Finally, include the inequality constraints. As in Task 5.4, define `H` and `Jh` and use them to define the linearized inequality constraints. Include them in the QP formulation and run the finalized version of the SQP solver.