1 The Generalized Gauss-Newton Method for Optimal Control

In this exercise, we will learn how to solve an optimal control problem using direct multiple shooting and a Gauss-Newton Hessian approximation.

The multiple shooting discretization: After defining a time grid \( t_k \leq t_{k+1} \), the discrete-time system dynamics are defined by simulating the continuous-time dynamics on each interval of length \( T_s \) separately

\[
x_{k+1} = \Phi(x_k, u_k), \quad k = 1, \ldots, N - 1
\]  

(1)

The nonlinear programming problem (NLP): The NLP resulting from the multiple shooting discretization is

\[
\begin{align*}
&\text{minimize} & & \sum_{k=0}^{N-1} \|x_k - x^r\|_Q^2 + \|u_k - u^r\|_R^2 \\
&\text{subject to} & & x_0 = \hat{x}(t_0), \\
& & & x_{k+1} = \Phi(x_k, u_k), \quad k = 0, \ldots, N - 1 \\
& & & x_N = \hat{x}_N, \\
& & & x^L \leq x_k \leq x^U, \quad k = 0, \ldots, N, \\
& & & u^L \leq u_k \leq u^U, \quad k = 0, \ldots, N - 1, 
\end{align*}
\]  

(2)

where \( Q \) and \( R \) are the weighting matrices.

The Gauss-Newton Hessian approximation: For a detailed description of the Gauss-Newton approximation, we refer to the lecture notes and we will now focus on the specific problem (2). Computing the exact Hessian for the NLP (2) involves computing the second-order deriva-
tives of the discrete-time model equations. The Lagrangian function of this problem is

\[ L(w, \lambda, \mu) = f(w) - \lambda^T g(w) - \mu^T h(w) \] (3)

\[ = w^T B w - \sum_{k=0}^{N-1} \|x_k - x^r\|^2_Q + \|u_k - u^r\|^2_R \] (4)

where \( w = [x_0, u_0, x_1, u_1, \ldots, x_N] \) and \( B = \text{diag}\{Q, R, Q, R, \ldots, Q\} \). This implies \( w^T B w = \sum_{k=0}^{N-1} x_k^T Q x_k + \sum_{k=0}^{N-1} u_k^T R u_k \).

The Hessian of the Lagrangian is

\[ \nabla_w^2 L(w, \lambda, \mu) = 2B + \sum_{j=1}^{N} \lambda_j \nabla_w^2 \Phi (x_{j-1}, u_{j-1}) \] (5)

The Gauss-Newton Hessian approximation is obtained by dropping the second term of (5), which gives

\[ \nabla_w^2 L(w, \lambda, \mu) \approx B. \] (6)

It can be shown that this Hessian approximation is particularly good when the residuals of the least-squares objective function are small.

**Sequential Quadratic Programming (SQP):** SQP consists in considering quadratic approximations of the NLP (2) and iteratively solving the quadratic programming (QP) problem

\[ \begin{align*}
\text{minimize} & \quad \sum_{k=0}^{N-1} \|x_k - x^r\|^2_Q + \|u_k - u^r\|^2_R \\
\text{subject to} & \quad x_0 = \hat{x}(t_0), \\
& \quad x_{k+1} = A_k x_k + B_k u_k, \quad k = 0, \ldots, N - 1 \\
& \quad x_N = \hat{x}_N, \\
& \quad \underline{x} \leq x_k \leq \bar{x}, \quad k = 0, \ldots, N, \\
& \quad \underline{u} \leq u_k \leq \bar{u}, \quad k = 0, \ldots, N - 1,
\end{align*} \] (7)

with \( A_k = \frac{\partial \Phi}{\partial x}(x_k, u_k) \) and \( B_k = \frac{\partial \Phi}{\partial u}(x_k, u_k) \).
The Real-Time Iteration Scheme: In order to speedup computations and have real-time feasible Nonlinear Mode Predictive Control (NMPC) implementations, the real time iteration (RTI) scheme proposes to only take a single full Newton step per sampling time. The initial value embedding consists in keeping the constraint on the initial state in the NLP: this allows us to initialize the solver at the previous trajectory and integration and sensitivity computation can be done before the initial state is known. In the moment at which the initial state $\hat{x}_0$ is known, only a QP needs to be solved and the feedback control can be applied immediately.

2 SQP for Optimal Control of a Pendulum

Let’s consider the overhead crane from the previous exercises.

1. Write a loop that recomputes the quadratic approximation (7) of problem (2) and solves the QP (7) at each iteration. Stop when the following criterion is met

   $$\left\| 2Bw - \nabla g(w)\lambda - \nabla h(w)\mu \right\|_{g(w)} \leq \min(0, h(w)) \leq \epsilon.$$

2. Starting from the code you have written, write a function called `MPCstep` that, given the current guess, reference trajectory and initial state, computes a full Newton step and returns the control to be applied to the system. If you want you can split this function into the two functions `MPCpreparation` and `MPCfeedback`. The first one linearizes the system and does all computations that can be prepared before $\hat{x}(t_0)$ is known, while the second one only updates the value of $\hat{x}(t_0)$ in the function $g(w)$ and solves the QP.

3. Write a closed-loop simulation that uses the functions that you just wrote for controlling the overhead crane used in the previous examples.

4. Add some step changes in the reference and simulate the system again.