Optimization Algorithms
Exercise 8

Marc Toussaint
Learning & Intelligent Systems Lab, TU Berlin
Marchstr. 23, 10587 Berlin, Germany

Winter 2020/21

The numbers of ex 2 are preliminary

1 Coding: Solving the Minimum fuel optimal control with bound constraints

In the last exercise you discussed *Minimum fuel optimal control*. In this exercise, you will implement a practical example of that. Assume a rocket wants to land on the surface of the moon. Let $x(0) = (10, -1)^T$ be the initial state of the rocket, where the first component is the height over the surface of the moon and the second one its initial velocity towards the surface. The dynamics of the rocket can be written as $x(t+1) = Ax(t) + bu$

$$A = \begin{pmatrix} 1 & 0.1 \\ 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 0 \\ 0.1 \end{pmatrix}. \tag{1}$$

We assume no gravity, vacuum and a point mass rocket. Sine we want to land on the surface without crashing into it, $x(N) = (0, 0)^T$ and $0 \leq x_1$. Further, we assume bounds on the control input $0 \leq u \leq 0.2$ and $N = 50$. For the cost function, consider

$$f(u) = u^2. \tag{2}$$

Implement the problem and use one of the reference solvers you implemented previously (Log Barrier, Augmented Lagrangian, Primal-Dual Newton) to solve it numerically by additionally adding the bound constraint optimization techniques you learned in the last exercise.

- Plot the evolution of u, and x_1, x_2.
- What happens if you remove the bound constraints on x_1 and/or u?

2 Facility Location

There are n facilities with which to satisfy the needs of m clients. The cost for opening facility j is f_j, and the cost for servicing client i through facility j is c_{ij}. You have to find an optimal way to open facilities and to associate clients to facilities.

Formulate as an ILP and then define a relaxation. If possible, come up with an interpretation for the relaxed problem.

3 Branch and bound for Integer Programming

This is a discussion exericise – no quantitative or coding work. Be prepared to discuss and report in the tutorials.

We will not discuss in detail branch-and-bound methods in the lecture – please learn about how to use branch-and-bound for Integer Programming yourself:

a) Read the Wikipedia page https://en.wikipedia.org/wiki/Branch_and_cut. Warning: they maximize instead of minimizing. The first 2 paragraphs (Algorithm) are somewhat fuzzy, but the pseudo-code is rather helpful.

b) What is the purpose of the upper bounds? (Lower bounds on their convention.)

c) What concrete pieces of code would be missing for you to realize such a solver based on our reference solvers?

d) Relate the branch-and-bound method to potentially all methods you learned about while studying. How it is related to CSP solvers, to A*, to Dynamic Programming?