

## Exercise 2: Optimization

Dr. Lilli Frison, Jochem De Schutter, Prof. Dr. Moritz Diehl

---

### Introduction to CasADi

The aim of this part is to deepen the understanding of optimization problems and in particular, to learn how to formulate and solve an optimization problem using CasADi.

**CasADi:** CasADi (see <https://web.casadi.org>) is an open-source software tool for numerical optimization in general and optimal control (i.e. optimization involving differential equations) in particular. The project was started by Joel Andersson and Joris Gillis while PhD students at the Optimization in Engineering Center (OPTEC) of the KU Leuven under supervision of Prof. Moritz Diehl. CasADi started out as a tool for algorithmic differentiation (AD) using a syntax borrowed from computer algebra systems (CAS), which explains its name. While AD still forms one of the core functionalities of the tool, the scope of the tool has since been considerably broadened, with the addition of support for ODE/DAE integration and sensitivity analysis, nonlinear programming and interfaces to other numerical tools. In its current form, it is a general-purpose tool for gradient-based numerical optimization – with a strong focus on optimal control. Note that CasADi is not an “optimal control problem solver”, that allows the user to enter an OCP and then gives the solution back. Instead, it tries to provide the user with a set of “building blocks” that can be used to implement general-purpose or specific-purpose OCP solvers efficiently with a modest programming effort.

We will use CasADi’s *Opti stack* because it provides a syntax close to paper notation. For the documentation and further examples see <https://web.casadi.org/docs/#document-opti> and <https://web.casadi.org/blog/opti/>. *Opti stack* is just a compact syntax from CasADi for NLP modeling, using a set of helper classes. It is also possible to formulate and solve the optimization problem using CasADi directly.

*Note:* CasADi is mainly a symbolic framework to formulate optimization problems and generate derivatives. To solve the problems it needs some underlying solvers installed, such as the NLP solver IPOPT or the MILP solver BONMIN or SCIP (some of which are already included).

To download and install CasADi for Python use the one-line command: `pip install casadi`

*Precaution for Conda (Anaconda / Miniconda) users:* avoid `conda install casadi`. Regardless of which channel, you will miss necessary solver plugins. Instead, use “pip” inside a conda environment, which you are able to do by running `conda install pip` inside the environment first.

You can use CasADi in the same way as any other package. Here is a simple example.

```
import casadi as ca
x = ca.MX.sym("x")
print(ca.jacobian(ca.sin(x), x))
```

Your output should be `cos(x)`.

### Exercise Tasks

1. **A tutorial example with CasADi:** Let’s first look at the following unconstrained optimization problem

$$\underset{x \in \mathbb{R}}{\text{minimize}} \quad x^2 - 2x$$

- (a) Derive first the optimal value for  $x$  on paper. Then, download the code provided for exercise 1 from the course homepage and run `ex2_toy_example.py` to solve the same problem with CasADi. Is the result the same? What is it?

$x^* =$

- (b) Have a closer look at the template and adapt it to include the inequality constraint  $x \geq 1.5$ . What is the new result? Is it what you would intuitively expect?

$x^* =$

- (c) Now modify the template to solve the two-dimensional problem:

$$\begin{aligned} &\underset{x, y \in \mathbb{R}}{\text{minimize}} \quad x^2 - 2x + y^2 + y \\ &x \geq 1.5 \\ &x + y \geq 0 \end{aligned}$$

What are the optimal values for  $x$  and  $y$  returned by CasADi?

$x =$   $y =$

2. **Optimal control with CasADi** Now that you have run the tutorial example, you know how to solve an optimization problem with CasADi's Opti stack. In the course, you have seen how to solve an optimal control problem by discretizing first the time-depending trajectories using a Runge Kutta integration algorithm and then solving the resulting equality-constrained optimization problem with a self-implemented Newton method. Now, instead of Newton method, use CasADi's Opti stack with IPOPT as an NLP solver to solve the optimization problem. You can start with the template `ex2_ocp_example.py`.

3. **Convexity** Which of the following sets is NOT convex ( $c \in \mathbb{R}^n, b \in \mathbb{R}^m, A \in \mathbb{R}^{m \times n}$ )?

(a) $\square \{x \in \mathbb{R}^n \mid \ Ax + b\ _2^2 \leq 3\}$	(b) $\square \{x \in \mathbb{R}^n \mid \log(c^\top x) \geq 3\}$
(c) $\square \{x \in \mathbb{R}^n \mid \exp(c^\top x) \leq 3\}$	(d) $\square \{x \in \mathbb{R}^n \mid \ Ax + b\ _2^2 \geq 3\}$

4. **Mixed-integer reformulation and Lagrangian** Consider the following *mixed-integer quadratic program* (MIQP):

$$\begin{aligned} \min_{x \in \{0,1\}^n} \quad & x^T Q x + q^T x \\ \text{s.t.} \quad & Ax \geq b, \end{aligned}$$

where  $Q \in \mathbb{R}^{n \times n}$ ,  $q \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m \times n}$  and  $b \in \mathbb{R}^m$ , and where the optimization variables  $x_i$  are restricted to take values in  $\{0, 1\}$ . Solving mixed-integer problems is in general a challenging task, thus it is common practice to look for continuous reformulations.

- Write down a continuous reformulation (hint: use the following relation  $x \in \{0, 1\} \leftrightarrow x(1 - x) = 0$ ). Is this reformulation convex?
- Formulate the Lagrangian for the continuous reformulation.