Exercises for Lecture Course on Numerical Optimization (NUMOPT)
Albert-Ludwigs-Universität Freiburg – Winter Term 2020-2021

Exercise 1: General Information, Introduction to CasADi, Convex Optimization

Prof. Dr. Moritz Diehl, Dimitris Kouzoupis and Florian Messerer

Part I: Introduction to CasADi

The aim of this part is to learn how to use MATLAB and how to formulate and solve an optimization problem using CasADi, namely the minimization of the potential energy of a chain of masses connected by springs.

Prepare your laptop

1. **MATLAB:** The exercises of this course are exclusively done in MATLAB. Instructions on how to get a free student license from the online software shop can be found here:

   `https://www.rz.uni-freiburg.de/services-en/beschaffung-em/software-en/matlab-license?set_language=en`

   If you are unfamiliar with MATLAB, here are some useful tutorials:

   - `http://www.maths.dundee.ac.uk/ftp/na-reports/MatlabNotes.pdf`
   - `http://www.math.mtu.edu/~msgocken/intro/intro.html`

2. **CasADi:** For this and future exercises we need to install CasADi. CasADi is an open-source tool for nonlinear optimization and algorithmic differentiation. Further information can be found at:

   `https://web.casadi.org`

   We will use CasADi's Opti stack because it provides a syntax close to paper notation. For the documentation see `https://web.casadi.org/docs/#document-opti`

   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 IPOPT, qpOASES, WORHP, KNITRO, ... (some of which are already included).

   To download and install CasADi follow the instructions below:

   (a) Download the current version for MATLAB from `https://web.casadi.org/get/` and unzip.

   (b) Move the folder called 'casadi-windows-matlabR2016a-v3.5.5' (or similar) to the default MATLAB directory or any directory of your choice.

   (c) Start MATLAB and go to the directory that you chose in Step (b).

   (d) Add the path of CasADi to the MATLAB path, by running:

   ```
   >> addpath('casadi-windows-matlabR2016a-v3.5.5')
   ```

   in the command line of MATLAB (adapt the folder name if necessary).

   (e) Test CasADi by running:

```
>> import casadi.*
>> x = MX.sym('x');
>> disp(jacobian(sin(x),x))
```

Your output should be `cos(x)`.

(f) To save the path beyond your current session of MATLAB, run

```
>> savepath
```

Exercise Tasks

3. **A tutorial example**: Let's first look at the following unconstrained optimization problem

$$\min_{x \in \mathbb{R}} \quad x^2 - 2x \tag{1}$$

(a) Derive first the optimal value for x on paper. Then, download the code provided for exercise 1 from the course homepage and run `ex1_toy_example.m` in MATLAB to solve the same problem with CasADi. Is the result the same?

$x^* =$ 1

(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?

$x^* =$ 1.5

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

$$\begin{aligned} \min_{x,y \in \mathbb{R}} \quad & x^2 - 2x + y^2 + y \\ \text{s.t.} \quad & x \geq 1.5, \\ & x + y \geq 0. \end{aligned} \tag{2}$$

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

$x =$ 1.5 $\quad$ $y =$ -0.5

4. **Equilibrium position of a hanging chain:** We want to model a chain of springs attached to two supports and hanging freely in between. The chain consists of N masses connected by $N-1$ massless springs. Each mass m_i has position (y_i, z_i), $i = 1, \dots, N$. We would like to find the equilibrium position which minimizes the potential energy of the full system. The potential energy associated with each spring is given by

$$V_{\text{el}}(y_i, y_{i+1}, z_i, z_{i+1}) = \frac{1}{2} D \left((y_i - y_{i+1})^2 + (z_i - z_{i+1})^2 \right),$$

for $i = 1, \dots, N-1$, and with spring constant $D \in \mathbb{R}^+$. The gravitational potential energy of each mass is

$$V_{\text{g}}(z_i) = m\, g\, z_i,$$

for $i = 1, \dots, N$, where g is the earth gravity constant, and $m = m_1 = \dots m_N$. The total potential energy is thus given by

$$V_{\text{chain}}(y, z) = \frac{1}{2} D \sum_{i=1}^{N-1} \left((y_i - y_{i+1})^2 + (z_i - z_{i+1})^2 \right) + mg \sum_{i=1}^{N} z_i,$$

where $y = (y_1, \dots, y_N)$ and $z = (z_1, \dots, z_N)$. The minimizing chain position is therefore the solution of the optimization problem

$$\begin{aligned} \min_{y, z \in \mathbb{R}^N} \quad & V_{\text{chain}}(y, z) \\ \text{s.t.} \quad & y_1 = \bar{y}_1, \\ & y_N = \bar{y}_N, \\ & z_1 = \bar{z}_1, \\ & z_N = \bar{z}_N. \end{aligned} \tag{3}$$

where $(\bar{y}_1, \bar{z}_1)$ and $(\bar{y}_N, \bar{z}_N)$ are the fixed positions of the outer masses.

(a) What type of optimization problem is this?

The objective function is quadratic and the constraints are affine $\Rightarrow$ the problem is a quadratic program (QP)

(b) Formulate the problem using $N = 40$, $m = 4/N$ kg, $D = \frac{70}{40} N$ N/m, $g = 9.81$ m/s^2 with the outer masses fixed to $(\bar{y}_1, \bar{z}_1) = (-2, 1)$ and $(\bar{y}_N, \bar{z}_N) = (2, 1)$. You can start from the template code `ex1_hanging_chain.m`. Solve the problem using CasADi with IPOPT as solver and interpret the results.

(c) Introduce ground constraints: $z_i \geq 0.5$ and $z_i - 0.1 y_i \geq 0.5$ for $i = 2, \dots, N-1$. What type of optimization problem is the resulting problem? Solve the problem and plot the result. Compare the result with the previous one.

We have added only affine constraints, so the problem is still a QP.

Part II: Convex Optimization

In this part we learn how to recognize convex sets and functions. Moreover we revisit the hanging chain problem from the previous part and add different types of constraints.

Exercise Tasks

5. **Convex sets and functions:** Determine whether the following sets and functions are convex or not.

   (a) A wedge, i.e., a set of the form (with $a_1, a_2 \in \mathbb{R}^n$, $b_1, b_2 \in \mathbb{R}$):

   $\{x \in \mathbb{R}^n | a_1^\top x \leq b_1, a_2^\top x \leq b_2\}$

   $a_i^\top x \leq b_i$ each defines a halfspace, which is convex. The set is a conjunction of two convex sets (the halfspaces), therefore convex.

   Alternative: show set definition also holds for $z = (1-t)x + ty$ with x, y elements of the set.

   (b) The set of points closer to a given point than a given set (with $x_0 \in \mathbb{R}^n$, $\mathcal{S} \subseteq \mathbb{R}^n$):

   $\{x \in \mathbb{R}^n \mid \|x - x_0\|_2 \leq \|x - y\|_2 \; \forall y \in \mathcal{S}\}$

   The set can be equivalently written as the intersection

   $$\Omega = \cap_{y \in \mathcal{S}} \{x \in \mathbb{R}^n \mid \|x - x_0\|_2 \leq \|x - y\|_2\} = \cap_{y \in \mathcal{S}} \Omega_y.$$

   Now each Ω_y defines a halfspace, which is convex. Then Ω is an intersection of convex sets and therefore convex itself.

   To see each Ω_y defines a halfspace:

   $\|x - x_0\|_2^2 \leq \|x - y\|_2^2 \Leftrightarrow (x - x_0)^\top (x - x_0) \leq (x - y)^\top (x - y)$

   $\Leftrightarrow x^\top x - 2x^\top x_0 + x_0^\top x_0 \leq x^\top x - 2x^\top y + y^\top y$

   $\Leftrightarrow \underbrace{(y - x_0)^\top}_{a_y^\top} x \leq \underbrace{\frac{1}{2}\left(\|y\|_2^2 - \|x_0\|_2^2\right)}_{b_y} \Leftrightarrow a_y^\top x \leq b_y$

   (c) The set of points closer to one set than another ($\mathcal{S}, \mathcal{T} \subseteq \mathbb{R}^n$):

   $C := \{x \in \mathbb{R}^n | \operatorname{dist}(x, \mathcal{S}) \leq \operatorname{dist}(x, \mathcal{T})\}$ with $\operatorname{dist}(x, \mathcal{S}) := \inf_{z \in \mathcal{S}} \|x - z\|_2$

   Not convex.

   Counter example: $n = 1$, $\mathcal{S} = \{-1, 1\}$, $\mathcal{T} = \{0\} \Rightarrow \mathcal{C} = \{x \in \mathbb{R} \mid x \leq -\frac{1}{2}$ or $x \geq \frac{1}{2}\}$

   (d) The function $f(x_1, x_2) = 1/(x_1 x_2)$ on $\mathbb{R}^2_{++}$.

   Convex if Hessian is positive semidefinite on the domain ($\mathbb{R}^2_{++}$)

   $\nabla^2 f(x_1, x_2) = \frac{1}{x_1 x_2} \begin{bmatrix} \frac{2}{x_1^2} & \frac{1}{x_1 x_2} \\ \frac{1}{x_1 x_2} & \frac{2}{x_2^2} \end{bmatrix}$

   A symmetric matrix A is positive definite iff its **leading** principal minors are > 0, i.e., the determinants of all upper-left quadratic submatrices. Check leading principal minors:

   $\det\left(\nabla^2 f(x_1, x_2)\right) = \frac{1}{x_1 x_2}\left(\frac{4}{x_1^2 x_2^2} - \frac{1}{x_1^2 x_2^2}\right) = \frac{3}{x_1^3 x_2^3} > 0 \; \forall x_1, x_2 \in \mathbb{R}^2_{++}$

   $\det\left(\frac{1}{x_1 x_2} \frac{2}{x_1^2}\right) = \frac{2}{x_1^3 x_2} > 0 \; \forall x_1, x_2 \in \mathbb{R}^2_{++}$

   $\Rightarrow$ Hessian is positive definite.

   Alternative: show $z^\top \nabla^2 f(x_1, x_2) z \geq 0 \; \forall z \in \mathbb{R}^2$, or compute eigenvalues.

   (e) The function $f(x_1, x_2) = x_1/x_2$ on $\mathbb{R}^2_{++}$.

   A is positive semidefinite $\Leftrightarrow$ **all** its principal minors are ≥ 0.

   $\nabla^2 f(x_1, x_2) = \begin{bmatrix} 0 & -\frac{1}{x_2^2} \\ -\frac{1}{x_2^2} & \frac{2x_1}{x_2^3} \end{bmatrix}$

   $\det\left(\nabla^2 f\right) = 0 - \frac{1}{x_2^4} < 0 \; \forall x_2 \in \mathbb{R}^2_{++} \Rightarrow$ Hessian is not PSD.

   Alternative: show $z^\top \nabla^2 f z \geq 0 \; \forall z \in \mathbb{R}^2$ does not hold, or compute eigenvalues.

6. **Minimum of coercive functions:** Prove that the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x)$$

with $f : \mathbb{R}^n \to \mathbb{R}$ a continuous, coercive function, has a global minimum point.

Hint: Use the Weierstrass Theorem and the following definition.

Definition (Coercive functions). A continuous function $f(x)$ that is defined on $\mathbb{R}^n$ is coercive if

$$\lim_{\|x\| \to \infty} f(x) = +\infty$$

or equivalently, if $\forall\, M\, \exists\, R : \|x\| > R \Rightarrow f(x) > M$.

Choose $M = f(0)$. Then from coerciveness we know that $\exists r \geq 0 : \; \|x\| > r \Rightarrow f(x) > f(0)$. Now define set $\Omega := \{x : \|x\| \leq r\}$, which is compact. Then we know from the Weierstrass Theorem that $\exists x^* : f(x^*) \leq f(x)\; \forall x \in \Omega$, i.e., x^* is a minimizer of f on Ω.
More specifically $f(x^*) \leq f(0)$ also holds , since $0 \in \Omega\; \forall r \geq 0$. We know that $f(0) < f(x) \forall x :$ $\|x\| > r$, so also $f(x^*) < f(x)\; \forall x \in \mathbb{R}^n \setminus \Omega$.
Therefore $f(x^*)$ is a global minimum of f on $\mathbb{R}^n$, and x^* a global minimizer.

7. **Hanging chain, revisited:** Recall the hanging chain problem from the previous part.

   (a) What would happen if you add, instead of the linear ground constraints, the nonlinear ground constraints $z_i \geq -0.2 + 0.1y_i^2$, for $i = 2, \ldots, N-1$ to your problem? Do not use MATLAB yet! What type of optimization problem is the resulting problem? Is it convex?

   The objective is still quadratic, but now there are quadratic constraints as well. Therefore it is no longer a QP. Instead it is a Quadratically Constrained Quadratic Program (QCQP).
   For each $i = 2, \ldots, N-1$, the constraint describes a parabola opened in positive z-direction, where all points above the parabola are part of the set. This is a convex set, therefore the NLP is still convex.
   Alternative: bring into standard NLP form $h(x) \geq 0$. This is a concave function, therefore the NLP is still convex (see lecture notes theorem 3.4).

   (b) What would happen if you add instead the nonlinear ground constraints $z_i \geq -y_i^2$, $i = 2, \ldots, N-1$? Do you expect this optimization problem to be convex?

   The additional constraints are still quadratic, therefore it is still a QCQP.
   The constraint describes a parabola opened in negative z-direction, where all points above the parabola are part of the set. This is not a convex set, therefore the NLP is not convex.

   (c) Now solve both variations using CasADi and plot the results (both chain and constraints). If any of these two optimization problems is non-convex, does it have multiple local minima? If yes, can you confirm that numerically by initializing the solver differently? Note that in CasADi you can provide an initial value `x0` for variable `x` via

   ```
   opti.set_initial(x,x0)
   ```

   As demonstrated in code, for the non-convex constraint from (b) different initializations find different solutions (local optima).