Numerical Optimal Control, August 2014

# Exercise 1: Quadratic programming

Joel Andersson   Joris Gillis   Moritz Diehl   University of Freiburg – IMTEK

August 4th, 2014

## Equilibrium position for a hanging chain

We want to model a chain attached to two supports and hanging in between. Let us discretise it with $N$ mass points connected by $N-1$ springs. Each mass $i$ has position $(y_i, z_i),\ i = 1, \ldots, N$. The equilibrium point of the system minimises the potential energy. The potential energy of each spring is

$$V_{\mathrm{el}}^i = \frac{1}{2} D_i \left( (y_i - y_{i+1})^2 + (z_i - z_{i+1})^2 \right).$$

The gravitational potential energy of each mass is

$$V_{\mathrm{g}}^i = m_i \, g_0 \, z_i.$$

The total potential energy is thus given by:

$$V_{\mathrm{chain}}(y, z) = \frac{1}{2} \sum_{i=1}^{N-1} D_i \left( (y_i - y_{i+1})^2 + (z_i - z_{i+1})^2 \right) + g_0 \sum_{i=1}^{N} m_i \, z_i, \tag{1}$$

where $y = [y_1, \cdots, y_N]^T$ and $z = [z_1, \cdots, z_N]^T$. We wish to solve:

$$\underset{y,z}{\text{minimize}} \qquad V_{\mathrm{chain}}(y, z). \tag{2}$$

The problem we want to solve is relatively simple; this gives us the possibility to easily analyse the behaviour of the numerical methods we will use. The problem can be made a bit more involved by adding inequality constraints, modelling a plane that the chain might touch.

Formulate the problem in the following form, which is how quadratic programs (QPs) are represented in CasADi:

$$\begin{aligned} \underset{x}{\text{minimize}} \qquad & \frac{1}{2} x^{\mathrm{T}} H \, x + g^{\mathrm{T}} x \\ \text{subject to} \qquad & x_{\mathrm{lb}} \le x \le x_{\mathrm{ub}}, \\ & a_{\mathrm{lb}} \le A \, x \le a_{\mathrm{ub}}, \end{aligned}$$

where $x = [y_1, z_1, \ldots, y_N, z_N]^T$. In this representation, you get an *equality* constraint by having upper and lower bound equal, i.e. $a_{\mathrm{lb}}^{(k)} = a_{\mathrm{ub}}^{(k)}$ for some $k$.

Tasks:

1.1 Formulate the problem using $N = 4$, $m_i = 40/N$ kg, $D_i = 70N$ N/m, $g_0 = 9.81$ m/s$^2$ with the first and last mass point fixed to $(-2, 1)$ and $(2, 1)$, respectively. **Before starting to program**, write down the required matrices and vectors **on paper** (yes, **on paper**).

1. In a Python script, formulate the above matrices as `numpy` arrays. The following should be helpful:

```
from numpy import *
A = zeros((nA,nx))
g = zeros(nx)
ubx = inf * ones(nx) # Upper bound on x is infinity
A[0,2] = 1 # set the element at the first row and third column to 1
...
```

where `nx` and `nA` are the number of variables and linear constraints, respectively. Try to use Python `for` loops to construct these matrices using N as a parameter.

1.2 Unfortunately, the standard *numpy* or *scipy* packages do not ship with a QP solver (like `quadprog` in MATLAB). To save you the trouble from installing a proper package for convex programming (for example CVXOPT), we have provided you with a simple function[1] on the course website that allows you to solve a QP using `qpOASES` via CasADi. Its usage is:

```
x = qpsolve(H,g,lbx,ubx,A,lba,uba)
```

1.3 Visualize the solution by plotting $(y, z)$ using `matplotlib`. This should be helpful:

```
from matplotlib import pylab as plt
plt.plot(Y,Z,'o-')
plt.show()
```

**Hint**: This might be a good occasion to use a Python `slice`.

1.3 Introduce ground constraints: $z_i \geq 0.5$ and $z_i - 0.1\,y_i \geq 0.5$. Solve your QP again and plot the result. Compare the result with the previous one.

1.4 **Extra**: What would happen if you add instead of the piecewise linear ground constraints, the nonlinear ground constraints $z_i \geq y_i^2$ to your problem? The resulting problem is no longer a QP, but is it convex?

1.5 **Extra**: What would happen if you add instead the nonlinear ground constraints $z_i \geq -y_i^2$ to your problem? Is the problem convex?

[1] Also available via `https://gist.github.com/jaeandersson/d95cbbcdd00e056e8c0f`