TEMPO Summer School on Numerical Optimal Control and Embedded Optimization
University of Freiburg, July 27 - August 7, 2015

Exercise 8: ECOS: Embedded Conic Solver

Alexander Domahidi Juan Jerez

http://syscop.de/teaching/numerical-optimal-control/

Preliminaries In this problem set, you will use the solver ECOS to solve some nonlinear (in terms of inequality constraints), but *convex* problems. ECOS is an open-source project, so you can view and download the source at `github.com/embotech/ecos`. There are interfaces to Matlab, Python and Julia, with their respective repositories. ECOS is supported by the modelling languages Yalmip, CVX, CVXPY and QCML. Contributors are always welcome - contact embotech for more information in case you want to get involved in the *development*.

1. Install ECOS: For *using* ECOS, obtain the precompiled Matlab MEX files as follows:
   (a) Register as a new user on `embotech.com/Register`, or login at `embotech.com/Login` if you are a registered user.
   (b) Navigate to the `embotech.com/ECOS/Download` and click "Download ECOS".
   (c) Save the zip file in a convenient location.
   (d) Go to that directory in Matlab and unpack the archive by double clicking or by typing in the command window: `unzip ecos-matlab-bin-allPlatforms-2.0.1.zip`
   (e) Add ECOS to the Matlab path, e.g. by right click on `ecos-matlab-bin-allPlatforms-2.0.1` → Add to Path → Selected Folders.
   (f) Test: in a different directory, type the commands `dims.l=1; x=ecos(1,sparse(-1),0,dims)` to solve the toy problem $\min_{x\geq 0} x$, which has the obvious solution $x = 0$. Verify that ECOS returns a number very close to zero. Read `help ecos`.

2. Problem types: for your reference, ECOS solves the problem

$$\begin{aligned} &\text{minimize } c^T x \\ &\text{subject to } Ax = b, \quad Gx \preceq_K h \end{aligned} \qquad \text{(ECOS-Primal)}$$

   The conic inequality reads as $s \in K, s \triangleq h - Gx$. The cone K is the cartesian product of 3 types of cones,

$$K \triangleq \mathbb{R}_+^n \quad \times \quad Q_1^{n_1} \times Q_1^{n_2} \times \cdots \times Q_M^{n_M} \quad \times \quad \mathcal{E}_1 \times \mathcal{E}_2 \times \dots \mathcal{E}_N \ , \qquad (1)$$

   where

   - $\mathbb{R}_+^n \triangleq \{x \in \mathbb{R}^n \mid x \geq 0\}$ (positive orthant) ,
   - $Q^n \triangleq \{(t, x) \in \mathbb{R}_+ \times \mathbb{R}^{n-1} \mid \|x\|_2 \leq t\}$ (second-order cone) ,
   - $\mathcal{E} \triangleq \text{cl}\{(x, y, z) \in \mathbb{R}^3 \mid e^{x/z} \leq y/z, z > 0\}$ (exponential cone) .

   To tell ECOS which cones are present in a problem, use the 4^{th} input argument `dims`, see Table 1.

Table 1: Correspondence of cones to the fields of the 4^{th} input argument of ECOS, the struct `dims`.

Cone	**Symbol**	**Matlab-code**
Positive orthant	$\mathbb{R}_Q^n$	`dims.l = n;`
Second-order cones	$Q_1^{n_1} \times Q_1^{n_2} \times \cdots \times Q_M^{n_M}$	`dims.q = [n1, n2, ..., nM];`
Exponential cones	$\mathcal{E}_1 \times \mathcal{E}_2 \times \dots \mathcal{E}_N$	`dims.e = N;`

Problem 1: ECOS as sparse QP solver The goal of this exercise is to use ECOS as a sparse QP solver to solve problems of the form

$$\begin{aligned} \text{minimize } & \frac{1}{2}x^T H x + f^T x \\ \text{subject to } & Dx = d \\ & l \leq Fx \leq u \end{aligned} \tag{QP}$$

with $H \in \mathbb{S}^n_+$ (H positive definite), $A \in \mathbb{R}^{p \times n}$ and $F \in \mathbb{R}^{m \times n}$. The task is to create a function with the prototype

```
function [x, solvetime] = myQPsolver(H,f,F,u,l,A,b)

% convert QP data into ECOS format
c = ...
G = ...
h = ...
dims = ...
A = ...
b = ...

% solve with ECOS
[xe, notused, info] = ecos(c,G,h,dims,A,b)

% return QP solution
x = ...
solvetime = info.timing.runtime;

end
```

1. Use an epigraph reformulation of the objective function to rewrite problem (QP) equivalently in (ECOS-Primal) form.

   Hint: Verify the following second-order cone representation of a quadratic function:

$$\left\{(t,x) \mid \frac{1}{2}x^T W^T W x + q^T x \leq t\right\} = \left\{(t,x) \mid \left\| \begin{matrix} Wx \\ \frac{t - q^T x - 1}{\sqrt{2}} \end{matrix} \right\|_2 \leq \frac{t - q^T x + 1}{\sqrt{2}}\right\}$$

2. Write the function `myQPsolver` that takes the problem data of (QP) as input arguments, converts them into the ECOS format, calls ECOS, and returns the solution. You can test your QP solver with the provided script `test_myQPsolver.m`.

3. Test your ECOS-based QP solver on problems with varying degree of sparsity by varying the number of variables `n`, the number of equality constraints `p` and the density factor `density_D` in `test_myQPsolver.m`. Compared to Matlab's `quadprog`, how long does it take to solve problems with 1000 variables, 50 sparse equality constraints with density factor 0.01?

   Solvetime QUADPROG: ☐ ECOS-QP: ☐

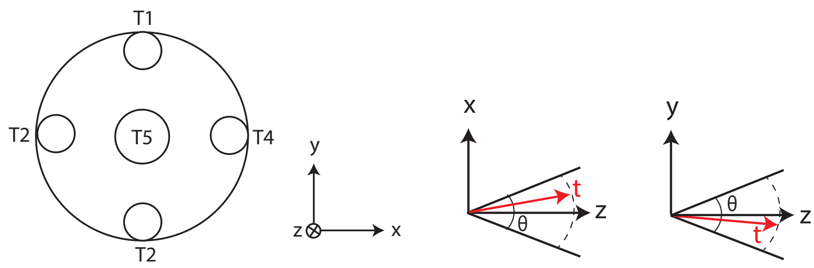

Figure 1: Left: Spacecraft with 5 thrusters for generating thrust in z direction. Right: feasible set of the thrust angles when rotating the thrusters around the y and x axis.

Table 2: Thruster specifications. Rotation angles around y and x axis are symmetric.

Thruster	**Min thrust [kN]**	**Max thrust [kN]**	**Min angle [deg]**	**Max angle [deg]**
1-4	0	100	-30	+30
5	0	400	-10	+10

Problem 2: Thruster allocation Spacecraft are often equipped with redundant thrusters. In this exercise, you will create an optimal thrust allocator (TA) using ECOS. The job of the TA is to compute a dispatch of thrust vectors for each thruster such that the overall generated thrust, which is a linear superposition of the individual thrusts is equal to a commanded thrust vector.

1. Consider the spacecraft in Figure 1 with 5 thrusters (the main thruster T5 and four auxiliary thrusters T1-4). Each of the thrusters can be tilted by electromagnetic actuators, and has a maximum amount of thrust it can produce, so thruster i can produce a three dimensional thrust vector $t_i = (t_{i,x}, t_{i,y}, t_{i,z})$. The scalar $\|t_i\|_2$ is the magnitude of the thrust produced by thruster T_i. Table 2 gives the specifications (constraints) for each thruster.

   Note: Neglect the moments generated by the thrusters throughout the exercise.

   (a) Formulate an optimization problem that dispatches the individual thrusters to track a given thrust reference: $(\sum_{i=1}^{5} t_i = \bar{t})$, while respecting the individual constraints in Table 2. Note that all thrusters can produce only positive thrust.

   Note: This is a feasibility problem.

   (b) Implement the optimization problem using the native ECOS interface. The optimal allocation for $\bar{t} = [10\ 20\ 700]^T$ should be similar to this:

Table 3: Optimal thrust allocation for $\bar{t} = [10\ 20\ 700]^T$.

t_1	t_2	t_3	t_4	t_5
1.9825	1.9825	1.9825	1.9825	2.0699
3.9145	3.9145	3.9145	3.9145	4.3421
77.0533	77.0533	77.0533	77.0533	391.7869

2. Due to a hardware failure, thruster T3 has been shut down and cannot produce any thrust. What is the maximum thrust in x direction that the spacecraft is capable of producing in this situation?

   Maximum thrust in x-direction after hardware failure of $T3$: $\boxed{}$

   Hint: Use the objective function to maximize for thrust in x-direction.

3. **Extra:** Due to nonlinear characteristics of the combustion process and the geometry of the nozzles, the amount of fuel m in kilograms required to produce a thrust of magnitude $\|t_i\|$ is given by

$$m_i(\|t\|) = \frac{c_i}{1.2\|t_i\|_{\max} - \|t_i\|} \quad . \tag{2}$$

where c_i is a geometry scaling factor for thruster T_i. We have $c_i = 1$ for $i = 1, \ldots, 4$ and $c_5 = 1.5$.

   (a) Plot the function $m_i(\|t_i\|)$. Is it convex?

   (b) Adjust the objective of your TA from task 1 to compute the minimum fuel thrust allocation.
   Hint: The set

   $$\text{closure}\{(t, s) \in \mathbb{R}^2 \mid ts \geq 1, s > 0\}$$

   can be represented by the second-order cone constraint

   $$\left\| \begin{matrix} \frac{t-s}{2} \\ 1 \end{matrix} \right\| \leq \frac{t+s}{2}$$

   How does the new objective impact the thrust dispatch?