Programming Exercise 2

Applied Numerical Optimazation

Wintersemester 2012/13

Global optimization algorithm

Problem description. Consider the optimization problem given by

$$\min_{x\in\mathbb{R}^n} x^T H x + c^T x \tag{1a}$$

$$\text{s.t. } x^T Q_i x + a_i x = b_i \; \forall\, i \in \{1, ..., m\} \tag{1b}$$

$$x^L \leq x \leq x^U \tag{1c}$$

with $n, m \in \mathbb{N}$, $m < n$, a symmetric, positive semidefinite matrix $H \in \mathbb{R}^{n\times n}$, symmetric indefinite matrices $Q_i \in \mathbb{R}^{n\times n} \forall i \in \{1, ..., m\}$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ with row vectors a_i, and $b \in \mathbb{R}^m$. The above problem is a nonconvex optimization problem with the possibility of suboptimal local minima.

Example. Consider the problem

$$\min_{x\in\mathbb{R}^3_+} x_1 + x_2 + x_3^2 \tag{2a}$$

$$\text{s.t. } x_1 x_2 + x_3 = 8 \tag{2b}$$

$$x_2 x_3 = 15 \tag{2c}$$

The terms x_1x_2 and x_2x_3, which are responsible for the non-convexity of the model, are called *bilinear* terms. In this example, the matrices and vectors in the general problem definition would be

$$c = [1, 1, 0]^T, \qquad b = [8, 15]^T, \qquad A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$
$$H = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad Q_1 = \begin{bmatrix} 0 & 0.5 & 0 \\ 0.5 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad Q_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0.5 \\ 0 & 0.5 & 0 \end{bmatrix} \tag{3}$$

Your task is to implement an algorithm for underestimating the optimal objective value of the optimization problem.

The Algorithm. In order to get a lower bound on the objective value of the nonconvex optimization problem, we need to construct a *convex relaxation* and solve it to global optimality.

One widely used relaxation of the *bilinear* terms $x_i x_j$ are the so-called *McCormick Envelopes* [1]. The relaxation works as follows: each term $x_i x_j$ is replaced by an auxiliary variable w_{ij}. Then, the following constraints for w_{ij} are defined:

$$w_{ij} \geq x_i^L x_j + x_i x_j^L - x_i^L x_j^L \tag{4a}$$
$$w_{ij} \geq x_i^U x_j + x_i x_j^U - x_i^U x_j^U \tag{4b}$$
$$w_{ij} \leq x_i^U x_j + x_i x_j^L - x_i^U x_j^L \tag{4c}$$
$$w_{ij} \leq x_i^L x_j + x_i x_j^U - x_i^L x_j^U \tag{4d}$$

These constraints are added to the original optimization problem, and the bilinear terms are replaced with the variables w_{ij}. The resulting convex QP is solved using standard methods, yielding a lower bound on the objective value of (1).

Your Task is to implement the following function in Matlab

```
function [f_lb, f_ub] = convex_bound(n, m, c, H, Q, A, lb, ub)
```

where $Q \in \mathbb{R}^{m \cdot n \times n}$ are the constraint matrices $Q_i \in \mathbb{R}^{n \times n}$ vertically concatenated, such that Q_i is the i-th submatrix of Q. This function must return a lower bound `f_lb` and an upper bound `f_ub` on the optimal value of the optimization problem (1).

In your function, follow the steps below:

1. Check the inputs for correct lengths. Q_i should be *symmetric*.
2. Analyze the matrix Q for how many auxiliary variables w need to be generated. It is extremely important that no auxiliary variable is generated twice (why?). Note that H is guaranteed to be positive semidefinite.
3. Generate an additional matrix B that implements the inequality constraints (4).
4. Compose the convex QP and solve it using Matlab's `quadprog` function.
5. Apply the local solver `fmincon` to the **original** problem to compute an **upper bound** on the problem.

[1] Garth P. McCormick. Computability of global solutions to factorable nonconvex programs. *Mathematical Programming*, 1976.