

An active-set algorithm for convex QPs

Mark S. Gockenbach

1 Introduction

I will now discuss an active-set strategy for solving the quadratic program

$$\min \quad \frac{1}{2}p \cdot Hp + g \cdot p \quad (1)$$

$$s.t. \quad Ap = b \quad (2)$$

$$Cp \geq d \quad (3)$$

where $H \in \mathbb{R}^{n \times n}$, $g \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ ($m < n$), $C \in \mathbb{R}^{p \times n}$, and $d \in \mathbb{R}^p$. I will assume that H is positive definite and that A has rank m . The first condition implies that (1–2) is a convex program and the second means that the rows of A are linearly independent.

2 Outline of the algorithm

The algorithm that I describe solves a sequence of equality-constrained QPs, each determined by the current estimate $p^{(j)}$ of the solution p^* and the current *working set* of constraints. The working set of constraints will be derived from the equality constraints and a subset of the inequality constraints (a guess at the inequality constraints active at the solution). To describe these constraints easily, I adopt the following notation: The rows of A will be denoted by $a^{(1)}, a^{(2)}, \dots, a^{(m)}$ and the rows of C by $a^{(m+1)}, a^{(m+2)}, \dots, a^{(m+p)}$. (The vector $a^{(i)}$ is the gradient of the corresponding linear constraint.) I will also set $b_{m+i} = d_i$, $i = 1, 2, \dots, p$.

Suppose, after I compute $p^{(j)}$ (an estimate of p^*), I suspect that the active constraints at p^* are

$$a^{(i_r)} \cdot p \geq b_{i_r}, \quad r = 1, 2, \dots, t,$$

where $m + 1 \leq i_1 < i_2 < \dots < i_t \leq m + p$. I set

$$W_j = \{1, 2, \dots, m, i_1, i_2, \dots, i_t\}.$$

(I will not put i_t in W_j unless $a^{(i_t)} \cdot p^{(j)} = b_{i_t}$; however, some inequality constraints may be active at $p^{(j)}$ and yet their indices not go in W_j . I explain this further below.) I then define $p^{(j+1)}$ to be the solution of the equality-constrained QP

$$\min \quad \frac{1}{2}p \cdot Hp + g \cdot p \quad (4)$$

$$s.t. \quad a^{(i)} \cdot p = b_i, \quad i \in W_j. \quad (5)$$

It turns out to be more convenient to express this QP in terms of the step $s^{(j)}$ from $p^{(j)}$ to $p^{(j+1)}$, rather than in terms of $p^{(j+1)}$ itself. It is easy to see that, with $s = p - p^{(j)}$, (4–5) is equivalent to

$$\min \quad \frac{1}{2}s \cdot Hs + g^{(j)} \cdot s \quad (6)$$

$$s.t. \quad a^{(i)} \cdot s = 0, \quad i \in W_j, \quad (7)$$

where $g^{(j)} = Hp^{(j)} + g$. (To derive this, I just substitute $p = p^{(j)} + s$ into (4–5) and use the fact that $a^{(i)} \cdot p^{(j)} = b_i$ for all $i \in W_j$.)

It would seem natural to define the working set to include all inequality constraints active at $p^{(j)}$. However, I want both the solution and Lagrange multiplier of the equality-constrained QP (6–7) to be unique, which requires that the set $\{a^{(i)} : i \in W_j\}$ be linearly independent. This might make it impossible to include all active constraints in the current working set. Therefore, in describing the following algorithm, I assume that the working set W_j satisfies the following properties:

1. Every original equality constraint is included in the working set: $\{1, 2, \dots, m\} \subset W_j$.
2. The set $\{a^{(i)} : i \in W_j\}$ is linearly independent.
3. If $m + i \in W_j$, then the i th inequality constraint is active at $p^{(j)}$: $a^{(m+i)} \cdot p^{(j)} = b_{m+i}$.

The algorithm requires an initial feasible point $p^{(0)}$:

$$\begin{aligned} Ap^{(0)} &= b, \\ Cp^{(0)} &= d. \end{aligned}$$

Also required is an initial working set W_0 satisfying the above requirements. In rough outline, the algorithm is simple:

Choose $p^{(0)}$ and W_0 .

For $j = 0, 1, 2, \dots$

Solve the equality-constrained QP (6–7) to get $s^{(j)}$.

Set $p^{(j+1)} = p^{(j)} + \alpha_j s^{(j)}$ for some $\alpha_j \in [0, 1]$.

Check to see if $p^{(j+1)}$ is the optimal solution to (1–2).

If not, update the working set W_j to get W_{j+1} .

The quadratic program (6–7) is a convex equality-constrained QP, whose solution can be found by solving the first-order optimality conditions, which take the form

$$\begin{aligned} Hp + q - \sum_{i \in W_j} \omega_i a^{(i)} &= 0, \\ a^{(i)} \cdot p - b_i &= 0, \quad i \in W_j. \end{aligned}$$

These equations define a *linear* system for p and ω_i , $i \in W_j$. Under the assumptions given on H and W_j , this system is guaranteed to have a unique solution. When the problem is large, solving this system can be expensive, and it is important to do it efficiently. However, for the purposes of describing this algorithm, I will just assume that the system is solved using some standard software for linear systems.

3 Updating the working set

The heart of the algorithm consists of techniques for recognizing an optimal solution, and for updating the working set. I will show below that $p^{(j)}$ is the solution to the original QP if $s^{(j)} = 0$ and the Lagrange multipliers ω_i , $i \in W_j$, are all nonnegative. Therefore, if $p^{(j)}$ is not a solution to the original QP, then either $s^{(j)} \neq 0$ or at least one ω_i is negative. I now consider these two possibilities in turn.

3.1 A nonzero step

The step $s^{(j)}$ is computed with the intention of setting $p^{(j+1)} = p^{(j)} + s^{(j)}$. However, the vector $p^{(j)} + s^{(j)}$ may be infeasible. Since, by construction, all of the constraints corresponding to the working set are satisfied by $p^{(j)} + s^{(j)}$, this can only happen if an inequality constraint not belonging to the working set is violated. Since $p^{(j)}$ is feasible,

$$a^{(i)} \cdot p^{(j)} \geq b_i$$

must hold for every $i \notin W_j$. Since

$$a^{(i)} \cdot (p^{(j)} + s^{(j)}) = a^{(i)} \cdot p^{(j)} + a^{(i)} \cdot s^{(j)},$$

if

$$a^{(i)} \cdot (p^{(j)} + s^{(j)}) < b_i,$$

then

$$a^{(i)} \cdot s^{(j)} < 0 \tag{8}$$

must hold. If (8) holds for some $i \notin W_j$, then $p^{(j+1)}$ is set to $p^{(j)} + \alpha_j s^{(j)}$, where $\alpha_j \geq 0$ is chosen so that $p^{(j+1)}$ is feasible. A little algebra shows that, if $a^{(i)} \cdot s^{(j)} < 0$ for $i \notin W_j$, then the largest value of α_j such that constraint i is satisfied by $p^{(j)} + \alpha s^{(j)}$ is

$$\alpha = \frac{b_i - a^{(i)} \cdot p^{(j)}}{a^{(i)} \cdot s^{(j)}}.$$

Since (8) may hold for more than one $i \notin W_j$, the appropriate choice of α_j is

$$\alpha_j = \min \left\{ 1, \min \left\{ \frac{b_i - a^{(i)} \cdot p^{(j)}}{a^{(i)} \cdot s^{(j)}} : i \notin W_j, a^{(i)} \cdot s^{(j)} < 0 \right\} \right\}. \tag{9}$$

If the i th constraint satisfies

$$\alpha_j = \frac{b_i - a^{(i)} \cdot p^{(j)}}{a^{(i)} \cdot s^{(j)}}$$

(even if α_j happens to equal 1), such a constraint is called a *blocking* constraint. As long as α_j is strictly positive, then each blocking constraint has the property that it is inactive at $p^{(j)}$ but active at $p^{(j+1)}$. Therefore, one of the corresponding indices i , say i_j , will be added to W_j to obtain W_{j+1} . For any such i_j ,

$$\begin{aligned} a^{(i)} \cdot s^{(j)} &= 0, \quad i \in W_j, \\ a^{(i_j)} \cdot s^{(j)} &< 0. \end{aligned}$$

From this it is obvious that $a^{(i_j)}$ is independent of

$$\{a^{(i)} : i \in W_j\},$$

and hence that W_{j+1} indexes a linearly independent set.

Another point to note is that when $s^{(j)} \neq 0$ and $\alpha_j > 0$, then $p^{(j+1)}$ yields a strictly smaller value for the objective function of the original QP than does $p^{(j)}$. (If it did not, then $s = 0$ would be a solution to (6–7).) This will be important later in proving that the algorithm produces a solution to the original QP in a finite number of steps.

If α_j is zero, then each blocking constraint is active at $p^{(j)}$ but is not in the working set. One of the blocking constraints is added to the working set to produce W_{j+1} and $p^{(j+1)}$ is set to $p^{(j)}$. (Thus, in this case, the updating is formally the same: $p^{(j+1)}$ is set to $p^{(j)} + \alpha_j s^{(j)}$ and a blocking constraint is added to the working set. However, the meaning of the step is different.)

3.2 A negative multiplier

I now suppose that the solution to

$$\begin{aligned} \min \quad & \frac{1}{2}s \cdot Hs + g^{(j)} \cdot s \\ \text{s.t.} \quad & a^{(i)} \cdot s = 0, \quad i \in W_j \end{aligned}$$

is $s^{(j)} = 0$, and at least one of the Lagrange multipliers ω_{i_r} , $i_r \in W_j$, $i_r > m$, is negative. This suggests that the objective function can be decreased by following a feasible direction away from the hyperplane defined by $a^{(i)} \cdot p = b_i$. Therefore, the working set W_{j+1} is defined to be $W_j \setminus \{i_r\}$ and $p^{(j+1)}$ is set to $p^{(j)}$. To show that this update is valid, I will show that the step $s^{(j+1)}$ produced by solving

$$\min \quad \frac{1}{2}s \cdot Hs + g^{(j+1)} \cdot s \tag{10}$$

$$\text{s.t.} \quad a^{(i)} \cdot s = 0, \quad i \in W_{j+1} \tag{11}$$

has the following properties:

1. It is a descent direction for the original quadratic objective function $q(p) = (1/2)p \cdot Hp + g \cdot p$ at $p^{(j)} = p^{(j+1)}$.
2. It points away from the hyperplane defined by $a^{(i_r)} \cdot p = b_{i_r}$, that is, $a^{(i_r)} \cdot s^{(j+1)} > 0$.

To prove this, I begin with the optimality conditions for (6–7). Since $s^{(j)} = 0$ is the solution, these imply form

$$g^{(j)} - \sum_{i \in W_j} \omega_i a^{(i)} = 0. \tag{12}$$

On the other hand, with $p^{(j+1)} = p^{(j)}$, it follows that $g^{(j+1)} = g^{(j)}$ and therefore the optimality conditions for (10–11) are

$$Hs^{(j+1)} + g^{(j)} - \sum_{\substack{i \in W_j \\ i \neq i_r}} \bar{\omega}_i a^{(i)} = 0, \tag{13}$$

$$a^{(i)} \cdot s^{(j+1)} = 0, \quad i \in W_j, i \neq i_r. \tag{14}$$

I will first show that $s^{(j+1)} \neq 0$. If not, then (12) and (13) yield

$$\begin{aligned} g^{(j)} &= \sum_{i \in W_j} \omega_i a^{(i)}, \\ g^{(j)} &= \sum_{\substack{i \in W_j \\ i \neq i_r}} \bar{\omega}_i a^{(i)}. \end{aligned}$$

Since $\{a^{(i)} : i \in W_j\}$ is linearly independent, $g^{(j)}$ can be expressed as a linear combination of $a^{(i)}$, $i \in W_j$, in only one way. Therefore,

$$\begin{aligned} \omega_i &= \bar{\omega}_i, \quad i \in W_j, \quad i \neq i_r, \\ \omega_{i_r} &= 0. \end{aligned}$$

But this contradicts the assumption that ω_{i_r} is negative. Thus $s^{(j+1)} = 0$ cannot hold.

To prove that $s^{(j+1)}$ is a descent direction for q , I must show that $g^{(j)} \cdot s^{(j+1)} < 0$, since $g^{(j)} = \nabla q(p^{(j)})$. But this follows from (13) and (14):

$$\begin{aligned} g^{(j)} \cdot s^{(j+1)} &= \sum_{\substack{i \in W_j \\ i \neq i_r}} \bar{\omega}_i a^{(i)} \cdot s^{(j+1)} - s^{(j+1)} \cdot H s^{(j+1)} \\ &= -s^{(j+1)} \cdot H s^{(j+1)} \quad (\text{since } a^{(i)} \cdot s^{(j+1)} = 0 \text{ for all } i \in W_j, i \neq i_r) \\ &< 0 \quad (\text{since } H \text{ is positive definite}) \end{aligned}$$

Finally, I show that $a^{(i_r)} \cdot s^{(j+1)}$ is positive. From (12),

$$\begin{aligned} a^{(i_r)} \cdot s^{(j+1)} &= \frac{1}{\omega_{i_r}} \left(g^{(j)} \cdot s^{(j+1)} - \sum_{\substack{i \in W_j \\ i \neq i_r}} \bar{\omega}_i a^{(i)} \cdot s^{(j+1)} \right) \\ &= \frac{1}{\omega_{i_r}} g^{(j)} \cdot s^{(j+1)} \\ &> 0 \quad (\text{since both } \omega_{i_r} \text{ and } g^{(j)} \cdot s^{(j+1)} \text{ are negative}). \end{aligned}$$

I have now proved the properties that I claimed for $s^{(j+1)}$. As before, the next step will be

$$p^{(j+2)} = p^{(j+1)} + \alpha_{j+1} s^{(j+1)},$$

where α_{j+1} is chosen by a consideration of the blocking constraints, if any. As before, $\alpha_{j+1} = 0$ is possible, in which case a blocking constraint will be added to the working set. However, the fact that $a^{(i_r)} \cdot s^{(j+1)} > 0$ means that constraint i_r is not a blocking constraint, and therefore cannot immediately re-enter the working set. If α_{j+1} is strictly positive, then, since $s^{(j+1)}$ is a descent direction for q , the objective function q is reduced by this step.

4 Recognizing an optimal solution

Earlier I stated that if $s^{(j)} = 0$ and the corresponding Lagrange multipliers ω_i , $i \in W_j$, are all nonnegative, then in fact $p^{(j)}$ is the optimal solution for the original QP (1–2). I will now prove this claim. The original QP can be written in the form

$$\min \quad \frac{1}{2} p \cdot H p + g \cdot p \quad (15)$$

$$s.t. \quad a^{(i)} \cdot p = b_i, \quad i = 1, 2, \dots, m, \quad (16)$$

$$a^{(i)} \cdot p \geq b_i, \quad i = m+1, m+2, \dots, m+p \quad (17)$$

The optimality conditions, which are sufficient since this is a convex program, are

$$H p + g - \sum_{i=1}^{m+p} \omega_i a^{(i)} = 0, \quad (18)$$

$$a^{(i)} \cdot p = b_i, \quad i = 1, 2, \dots, m, \quad (19)$$

$$a^{(i)} \cdot p \geq b_i, \quad i = m+1, m+2, \dots, m+p, \quad (20)$$

$$\omega_i \geq 0, \quad i = m+1, m+2, \dots, m+p, \quad (21)$$

$$\omega_i (a^{(i)} \cdot p - b_i) = 0, \quad i = m+1, m+2, \dots, m+p. \quad (22)$$

If $s^{(j)} = 0$ is the solution of

$$\begin{aligned} \min \quad & \frac{1}{2}s \cdot Hs + g^{(j)} \cdot s \\ \text{s.t.} \quad & a^{(i)} \cdot s = 0, \quad i \in W_j, \end{aligned}$$

then there exist multipliers ω_i , $i \in W_j$, such that

$$Hp^{(j)} + g - \sum_{i \in W_j} \omega_i a^{(i)} = 0.$$

Defining $\omega_i = 0$ for $i \notin W_j$, it follows that (18) and (21) are satisfied by $p^{(j)}$ and w_i , $i = 1, 2, \dots, m + p$. Moreover, the algorithm described above guarantees that $p^{(j)}$ is feasible, so (19) and (20) are satisfied as well. It remains only to show that the complementarity condition (22) is satisfied. However, this follows immediately from the facts that each inequality constraint corresponding to $i \in W_j$ is active at $p^{(j)}$ and $\omega_i = 0$ for all $i \notin W_j$. This completes the proof.

5 Example

I will now solve the following QP in two variables:

$$\begin{aligned} \min \quad & (p_1 - 3)^2 + 2(p_2 - 2)^2 \\ \text{s.t.} \quad & p_1 \geq 0 \\ & p_2 \geq 0 \\ & p_1 - p_2 \geq -1 \\ & -2p_1 - p_2 \geq -3. \end{aligned}$$

Using the notation I described above,

$$\begin{aligned} H &= \begin{bmatrix} 2 & 0 \\ 0 & 4 \end{bmatrix}, \quad g = \begin{bmatrix} -6 \\ -8 \end{bmatrix}, \\ a^{(1)} &= (1, 0), \quad a^{(2)} = (0, 1), \quad a^{(3)} = (1, -1), \quad a^{(4)} = (-2, -1), \\ b_1 &= 0, \quad b_2 = 0, \quad b_3 = -1, \quad b_4 = -3. \end{aligned}$$

I now describe the steps of the algorithm, applied to this example. I write out the first couple of steps in detail and just summarize the rest.

Step 0: I choose $p^{(0)} = (0, 0)$, $W_0 = \{1, 2\}$ (so the initial working set consists of the constraints $p_1 = 0$, $p_2 = 0$).

Step 1: The gradient $g^{(0)}$ is given by $g^{(0)} = Hp^{(0)} + g = g = (-6, -8)$. I must solve

$$\begin{aligned} \min \quad & \frac{1}{2}s \cdot Hs + g^{(0)} \cdot s \\ \text{s.t.} \quad & a^{(1)} \cdot s = 0 \\ & a^{(2)} \cdot s = 0. \end{aligned}$$

Since the constraints reduce to $s_1 = 0$, $s_2 = 0$, the solution is $s^{(0)} = (0, 0)$. The Lagrange multiplier equation

$$Hs + g^{(0)} = \omega_1 a^{(1)} + \omega_2 a^{(2)}$$

then yields $\omega_1 = -6$, $\omega_2 = -8$. Since $s^{(0)} = 0$ but ω_1 and ω_2 are both negative, this is the case described in Section 3.2, and one of the constraints must be removed from the working set. I will remove constraint 2 and set $W_1 = \{1\}$. The next iterate is $p^{(1)} = p^{(0)} + s^{(0)} = 0$.

Step 2: The gradient is $g^{(1)} = Hp^{(1)} + g = g = (-6, -8)$. I must solve

$$\begin{aligned} \min \quad & \frac{1}{2}s \cdot Hs + g^{(0)} \cdot s \\ \text{s.t.} \quad & a^{(1)} \cdot s = 0, \end{aligned}$$

for which the optimality conditions are

$$\begin{aligned} Hs + g^{(1)} &= \omega_1 a^{(1)}, \\ a^{(1)} \cdot s &= 0. \end{aligned}$$

These equations are

$$\begin{aligned} 2s_1 - 6 &= \omega_1, \\ 4s_2 - 8 &= 0, \\ s_1 &= 0, \end{aligned}$$

and the solution is easily seen to be $s_1 = 0$, $s_2 = 2$, $\omega_1 = -6$. Therefore $s^{(1)} = (0, 2)$ and the case described in Section 3.1 applies. I must determine α_1 so that $p^{(2)} = p^{(1)} + \alpha_1 s^{(1)}$. To do this, I check for blocking constraints:

$$\begin{aligned} a^{(2)} \cdot s^{(1)} &= 2 > 0, \\ a^{(3)} \cdot s^{(1)} &= -2 < 0, \\ a^{(4)} \cdot s^{(1)} &= -2 < 0. \end{aligned}$$

Constraints 3 and 4 are potential blocking constraints, and the next step is to compute α_1 :

$$\begin{aligned} \frac{b_3 - a^{(3)} \cdot p^{(1)}}{a^{(3)} \cdot s^{(1)}} &= \frac{1}{2}, \\ \frac{b_4 - a^{(4)} \cdot p^{(1)}}{a^{(4)} \cdot s^{(1)}} &= \frac{3}{2}, \\ \alpha_1 = \min \left\{ 1, \frac{1}{2}, \frac{3}{2} \right\} &= \frac{1}{2}. \end{aligned}$$

Therefore, constraint 3 is the blocking constraint and

$$p^{(2)} = p^{(1)} + \alpha_1 s^{(1)} = (0, 1), \quad W_2 = \{1, 3\}.$$

Step 3: I will just summarize the computations for the remaining steps. Solving the equality-constrained QP yields

$$s^{(2)} = (0, 0), \quad \omega_1 = -10, \quad \omega_2 = 4.$$

This is the case described in Section 3.2; constraint 1 must be dropped from the working set:

$$p^{(3)} = (0, 1), \quad W_2 = \{3\}.$$

Step 4: The equality-constrained QP yields

$$s^{(3)} = \left(\frac{5}{3}, \frac{5}{3} \right), \quad \omega_3 = -\frac{8}{3}.$$

The only potential blocking constraint is constraint 4, and

$$\frac{b_4 - a^{(4)} \cdot p^{(3)}}{a^{(4)} \cdot s^{(3)}} = \frac{2}{5} < 1.$$

Therefore, $\alpha_3 = 2/5$, constraint 4 is the blocking constraints, and

$$p^{(4)} = p^{(3)} + \alpha_3 s^{(3)} = \left(\frac{2}{3}, \frac{5}{3} \right), \quad W_4 = \{3, 4\}.$$

Step 5: The equality-constrained QP yields

$$s^{(4)} = (0, 0), \omega_3 = -\frac{2}{3}, \omega_4 = 2.$$

Constraint 3 must be removed from the working set, so

$$p^{(5)} = p^{(4)} = \left(\frac{2}{3}, \frac{5}{3}\right), W_5 = \{4\}.$$

Step 6: The equality-constrained QP yields

$$s^{(5)} = \left(\frac{1}{9}, -\frac{2}{9}\right), \omega_4 = \frac{20}{9}.$$

Only constraint 2 is a potential blocking constraint; however,

$$\frac{b_2 - a^{(2)} \cdot p^{(5)}}{a^{(2)} \cdot s^{(5)}} = \frac{15}{2} > 1,$$

so $\alpha_5 = 1$. There is no blocking constraint and hence no change in the working set:

$$p^{(6)} = p^{(5)} + s^{(5)} = \left(\frac{7}{9}, \frac{13}{9}\right), W_6 = \{4\}.$$

Step 7: The algorithm terminates, since solving the equality-constrained QP yields

$$s^{(6)} = (0, 0), \omega_4 = \frac{20}{9}.$$

The solution is

$$p^* = \left(\frac{7}{9}, \frac{13}{9}\right)$$

and the vector of Lagrange multipliers is

$$\omega = \left(0, 0, 0, \frac{20}{9}\right).$$

Having gone through the above steps to find the solution of this simple QP, the reader might logically wonder if there is not a simpler way. In particular, why not use the optimality conditions to formulate and solve a system of equations?

For the current example, the optimality conditions are

$$\begin{aligned} 2p_1 - 6 &= \theta_1 + \theta_3 - 2\theta_4, \\ 4p_2 - 8 &= \theta_2 - \theta_3 - \theta_4, \\ \theta_1 p_1 &= 0, \\ \theta_2 p_2 &= 0, \\ \theta_3(p_1 - p_2 + 1) &= 0, \\ \theta_4(2p_1 + p_2 - 3) &= 0, \\ \theta_i &\geq 0, \quad i = 1, 2, 3, 4, \\ p_1, p_2 &\geq 0, \\ p_1 - p_2 &\geq -1, \\ -2p_1 - p_2 &\geq -3. \end{aligned}$$

The first six conditions define six nonlinear equations in six unknowns. Solving them algebraically results in 11 solutions, 10 of which fail to satisfy one or more inequality constraints or one or more nonnegativity constraints on the multipliers. The 11th solution is the one found above.

This proliferation of (potential) solutions to the optimality conditions is to be expected when inequality constraints are present. Indeed, with p inequality constraints, the complementarity condition suggests that there may be up to 2^p solutions. For example, in solving the above system, one must consider the cases $\theta_1 = 0$ and $p_1 = 0$, and for each of these cases, the cases $\theta_2 = 0$ and $p_2 = 0$, and so forth. Of course, not all of these possibilities leads to a solution, but they must all be considered. Therefore, when p is large, this is not a practical approach. It is not immediately obvious that the algorithm described above *is* practical, but it turns out to be efficient, provided care is taken in its implementation. I discuss the efficient implementation briefly below.

6 Finite termination of the algorithm

I will now discuss the question of whether the above algorithm is guaranteed to converge in a finite number of iterations. Those familiar with the concept of cycling in linear programming will not be surprised by the conclusion: The algorithm is guaranteed to converge if it does not cycle. Cycling is usually considered a remote possibility and ignored in practice. Therefore, the algorithm (as implemented) is not guaranteed to terminate, but it virtually always does.

Cycling is the phenomenon in which, for some $j \geq 0$ and $k > 0$,

$$p^{(j)} = p^{(j+1)} = \dots = p^{(j+k)}$$

and $W_{j+k} = W_j$. These conditions imply that, over a course of k iterations, the estimate $p^{(j)}$ of the solution never changes, while constraints are added to and removed from the working set in such a way that $W_{j+k} = W_j$. If this occurs, then it is possible that the same sequence of iterations will be repeated over and over again, in which case the algorithm will never terminate.

There are two conditions under which $p^{(j+1)} = p^{(j)}$ can occur. First of all, if $s^{(j)} = 0$, then necessarily $p^{(j+1)} = p^{(j)}$. Second, it can happen that $s^{(j)} \neq 0$ holds, but $\alpha_j = 0$. This second case only occurs when one of the constraints not currently in the working set is both active and blocking.

To illustrate the second possibility, I add the (redundant) constraint $-p_2 \geq -5/3$ to the QP treated in the previous section. The new constraint will be constraint 5, with $a^{(5)} = (0, -1)$ and $b_5 = -5/3$. The first three steps of the algorithm proceed as before, but in Step 4 the new constraint is also a blocking constraint. Therefore, either constraint 4 or 5 can be added to the working set. If I choose to add constraint 5, then

$$p^{(4)} = \left(\frac{2}{3}, \frac{5}{3} \right), \quad W_4 = \{3, 5\}.$$

The next step then results in

$$s^{(4)} = (0, 0), \quad \omega_3 = -\frac{14}{3}, \quad \omega_5 = 6,$$

and constraint 3 must be dropped from the working set, giving

$$p^{(5)} = \left(\frac{2}{3}, \frac{5}{3} \right), \quad W_5 = \{5\}.$$

The new step 6 produces

$$s^{(5)} = \left(\frac{7}{3}, 0 \right), \quad \omega_5 = \frac{4}{3}.$$

The only potentially blocking constraint is constraint 4, with

$$a^{(4)} \cdot s^{(5)} = -\frac{14}{3}$$

and

$$\frac{b_4 - a^{(4)} \cdot p^{(5)}}{a^{(4)} \cdot s^{(5)}} = \frac{-3 - (-3)}{-14/3} = 0.$$

This is exactly the situation described above: constraint 4 does not belong to the current working set, but it is both active and a blocking constraint, forcing $\alpha_5 = 0$ to hold. It follows that

$$p^{(6)} = \left(\frac{2}{3}, \frac{5}{3} \right), \quad W_5 = \{4, 5\}.$$

In this example, the redundant constraint does not cause any particular difficulty; the next step causes constraint 5 to be dropped from the working set, and then the solution is found as before in the following step. However, in a more complicated example it is conceivable that a cycle could appear. I will now show that if the above phenomenon does not occur, that is, if α_j is strictly positive whenever $s^{(j)}$ is nonzero, then the algorithm terminates after a finite number of steps.¹

I first make three preliminary observations:

1. There is a finite number of possible working sets (this number is bounded by the number of subsets of $\{m+1, m+2, \dots, m+p\}$, which is 2^p).
2. Each working set W_j corresponds to a unique approximation $p^{(j+1)}$, determined by solving the QP (4-5) and hence to a unique value of the objective function q .
3. In moving from $p^{(j)}$ to $p^{(j+1)}$ by the above algorithm, the value of q cannot increase; that is,

$$q(p^{(j+1)}) \leq q(p^{(j)})$$

holds for all j .

The above observations imply that if q is reduced in moving from $p^{(j)}$ to $p^{(j+1)}$, then the working set W_j will never be visited again. Since there is a finite number of possible working sets, it suffices to show that each working set encountered is eventually changed in a step that reduces q .

If $s^{(j)} = 0$, then $p^{(j)}$ is the global minimizer of the quadratic objective function q on the feasible set defined by the current working set. This can be verified directly from the optimality conditions. The vector $p^{(j)}$ is also the solution of the original QP if and only if the multipliers ω_i , $i \in W_j$, $i > m$, are all nonnegative. On the other hand, if at least one of the multipliers is negative, then removing the corresponding constraint from the active set produces a nonzero $s^{(j+1)}$ (I proved this on page 4). Under the assumption that α_{j+1} cannot be zero, this leads to a decrease in the objective function q in moving from $p^{(j+1)}$ to $p^{(j+2)}$. Therefore, W_j can never be encountered again.

The final step in the proof is to show that $s^{(j)} = 0$ occurs at least every $n - m$ steps in the algorithm. (Then, by the argument in the previous paragraph, a working set is discarded, never to be encountered again, at least every $n - m$ iterations.) To prove this, I first note that if $p^{(j)} \neq 0$, then $\alpha_j > 0$ by assumption. If $\alpha_j = 1$ and no blocking constraint is encountered, then $p^{(j+1)} = p^{(j)} + s^{(j)}$ is the minimizer of q over the feasible set defined by the working set W_j , so $s^{(j+1)} = 0$ will hold. On the other hand, when $\alpha_j \leq 1$ and there is a blocking constraint, then W_{j+1} is formed by adding a new constraint to W_j . This can happen at most $n - m$ times before the working set contains n linearly independent constraints, which then determine a unique point. Clearly, on the next iteration, the step will be zero. This completes the proof that $s^{(j)} = 0$ occurs at least every $n - m$ iterations, and also the proof that the algorithm terminates after a finite number of steps (under the given assumption).

¹This proof is based on Nocedal and Wright [1], pages 466-467.

7 A note about efficient implementation

Each step of the above algorithm requires the solution of an equality constrained QP with n variables and t constraints, $m \leq t \leq n$. The solution to each QP is obtained by solving the $(n+t) \times (n+t)$ extended system obtained from the optimality conditions. The efficiency of the algorithm depends on the expense of solving these linear systems.

Fortunately, it is not necessary to solve each extended system “from scratch.” Instead, after factoring the first extended system, the factorization for each subsequent system can be obtained by updating the previous one. This is possible because, from one step to the next, the system changed only by the addition or deletion of one row and column. Details are found in Section 16.4 of Nocedal and Wright [1] (and the references given therein).

References

- [1] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer-Verlag, New York, 1999.