INTERIOR-POINT METHODS FOR NONCONVEX NONLINEAR PROGRAMMING: COMPLEMENTARITY CONSTRAINTS

HANDE Y. BENSON, DAVID F. SHANNO, AND ROBERT J. VANDERBEI

Operations Research and Financial Engineering
Princeton University
ORFE-02-02

August 6, 2002

ABSTRACT. In this paper, we present the formulation and solution of optimization problems with complementarity constraints using an interior-point method for nonconvex nonlinear programming. We identify possible difficulties that could arise, such as unbounded faces of dual variables, linear dependence of constraint gradients and initialization issues. We suggest remedies. We include encouraging numerical results on the MacMPEC test suite of problems.

1. INTRODUCTION

We consider optimization problems with complementarity constraints:

$$
\begin{aligned}
&\text{minimize} &&f(x)\\
&\text{subject to} &&0 \leq g(x) \perp x \geq 0,
\end{aligned}
\tag{1}
$$

where $x \in \mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}$, and $g : \mathbb{R}^n \to \mathbb{R}^n$. The notation

$$
0 \leq g(x) \perp x \geq 0
\tag{2}
$$

is shorthand for

$$g(x) \geq 0, \quad x \geq 0, \text{ and } g_j(x)x_j = 0 \text{ for all } j.$$

Such constraints are called *complementarity constraints*, because the pairs $(g_i(x), x_i)$, $i = 1, \ldots, n$ are *complementary*. We restrict our attention in this paper to the problem formulation given by (1), but, in general, other variables and linear or nonlinear constraints can appear in the optimization model as well. There can also be nonzero bounds on the complementary pairs. The complementary pairs can also both be functions. In fact, many of the problems in the test set have the form:

$$(3) \qquad \begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & l_1 \leq g(x) \leq u_1 \perp l_2 \leq h(x) \leq u_2, \end{aligned}$$

where $x \in \mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$, $h : \mathbb{R}^n \to \mathbb{R}^m$, and $l_1, u_1, l_2, u_2 \in \mathbb{R}^m$ are the lower and upper bounds on the expressions. For any single complementary pair, exactly two of these bounds are finite. Hence, there are several possible forms for a complementarity constraint. One illustrative example is *mixed complementarity*, in which there are both lower and upper bounds on one of the expressions, say $g(x)$:

$$(4) \qquad \begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & l_1 \leq g(x) \leq u_1 \perp h(x). \end{aligned}$$

The complementarity constraint means that

$$(5) \qquad \begin{aligned} & l \leq g(x) \leq u \\ & \text{if } g_i(x) = l_i \text{ then } h_i(x) \geq 0, \quad i = 1, \ldots, m \\ & \text{if } l_i < g_i(x) < u_i \text{ then } h_i(x) = 0, \quad i = 1, \ldots, m \\ & \text{if } g_i(x) = u_i \text{ then } h_i(x) \leq 0, \quad i = 1, \ldots, m. \end{aligned}$$

The arguments that will be presented in this paper for the form (1) readily translate to this general formulation.

Constraints of the form (2) are also called *equilibrium constraints* as they often arise in problems relating to economic and physical equilibria. Therefore, a problem of the form (1) is called a *mathematical program with equilibrium constraints* (MPEC). An excellent survey of application areas where MPECs arise can be found in [4].

Recently, Fletcher, et.al. [5] proposed solving MPECs using sequential quadratic programming methods for nonlinear programming. In this paper, we present an interior-point algorithm for solving these problems. In Section 2, we describe the algorithm for solving general-purpose nonlinear programming problems (NLPs). In Section 3, we present ways to reformulate the complementarity constraints to fit them

into the NLP paradigm. We also analyze the possible trouble spots for an interior-point method and propose ways to overcome them. In Section 4, we present numerical results on the MacMPEC test suite [7].

2. LOQO: An Infeasible Interior-Point Method

We begin with an overview of the LOQO algorithm. A more detailed explanation can be found in [8]. The basic problem we consider is

$$
\begin{aligned}
& \text{minimize } f(x) \\
& \text{subject to } h_i(x) \geq 0, \qquad i = 1, \ldots, m,
\end{aligned}
\tag{6}
$$

where $f(x)$ and the $h_i(x)$'s are assumed to be twice continuously differentiable and x is an n-vector of otherwise unconstrained (i.e., free) variables.

We add nonnegative slacks, w, to the inequality constraint in (6) and incorporate these slacks in a logarithmic barrier term in the objective function:

$$
\begin{aligned}
& \text{minimize } f(x) - \mu \sum_{i=1}^{m} \log(w_i) \\
& \text{subject to } h(x) - w = 0.
\end{aligned}
\tag{7}
$$

Denoting the Lagrange multipliers for the system (7) by y, the first order conditions for the problem are

$$
\begin{aligned}
\nabla f(x) - \nabla h(x)y &= 0, \\
-\mu e + WYe &= 0, \\
h(x) - w &= 0,
\end{aligned}
\tag{8}
$$

where e is the vector of all ones, $\nabla h(x)$ is the transpose of the Jacobian matrix of the vector $h(x)$, and W and Y are the diagonal matrices with the w_i's and y_i's on the diagonal, respectively.

Newton's method is employed to iterate to a triple (x, w, y) satisfying (8). Letting

$$
\begin{aligned}
H(x, y) &= \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 h_i(x), \\
A(x) &= \nabla h(x)^T, \\
\sigma &= \nabla f - A(x)^T y, \\
\gamma &= \mu W^{-1} e - y, \text{ and} \\
\rho &= w - h(x),
\end{aligned}
$$

the directions given by Newton's method are found by solving the *reduced KKT system*

$$(9) \qquad \begin{bmatrix} -H(x,y) & A(x)^T \\ A(x) & WY^{-1} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \sigma \\ \rho + WY^{-1}\gamma \end{bmatrix},$$

and

$$(10) \qquad \Delta w = WY^{-1}(\gamma - \Delta y).$$

LOQO then proceeds to a new estimate of the optimum by

$$(11) \qquad \begin{aligned} x^{(k+1)} &= x^{(k)} + \alpha^{(k)}\Delta x^{(k)}, \\ w^{(k+1)} &= w^{(k)} + \alpha^{(k)}\Delta w^{(k)}, \\ y^{(k+1)} &= y^{(k)} + \alpha^{(k)}\Delta y^{(k)}, \end{aligned}$$

where $\alpha^{(k)}$ is chosen to ensure that $w^{(k+1)} > 0$, $y^{(k+1)} > 0$, and either the barrier function or the infeasibility is reduced.

LOQO concludes that it has reached an optimal solution when the *primal infeasibility*, ρ, and the *dual infeasibility*, σ, are both less than 10^{-6}, and the primal and the dual objective functions agree to at least 8 significant digits. The primal objective function is given by the value of $f(x)$ at the current iterate. The dual objective function is given by

$$f(x) - y^T h(x) + (A(x)^T y - \nabla f(x))^T x.$$

3. COMPLEMENTARITY CONSTRAINTS

In this section, we focus on how to express complementarity constraints in an optimization problem and how to solve this problem using the interior-point algorithm described in the previous section.

3.1. **Modeling with Complementarity Constraints.** Consider a complementarity constraint in the form:

$$(12) \qquad 0 \le g(x) \perp h(x) \ge 0.$$

Such a constraint can be reformulated in three ways:

- The expressions themselves can be taken as the complementary pairs:

$$(13) \qquad \begin{aligned} & g(x) \ge 0 \\ & h(x) \ge 0 \\ & g_i(x)h_i(x) = 0, \quad i = 1,\ldots,m. \end{aligned}$$

- One of the two constraints, say $g(x)$ can be slacked:

(14)
$$\begin{aligned}
& g(x) - w = 0 \\
& w \geq 0 \\
& h(x) \geq 0 \\
& w_i h_i(x) = 0, \quad i = 1, \ldots, m.
\end{aligned}$$

- Both of the expressions in the constraint can be slacked:

(15)
$$\begin{aligned}
& g(x) - w = 0 \\
& h(x) - s = 0 \\
& w \geq 0 \\
& s \geq 0 \\
& w_i s_i = 0, \quad i = 1, \ldots, m.
\end{aligned}$$

These three formulations have the same optimal solution, but they have different algorithmic performance. The choice of which one to use can be dictated by the user's preference and the modeling language.

In the presence of both lower and upper bounds on one of the expressions, we use a pair of inequalities in the reformulation. The constraints given by (5) can be expressed alternatively as

(16)
$$\begin{aligned}
& l \leq g(x) \leq u \\
& (g_i(x) - l_i)h_i(x) \leq 0, \quad i = 1, \ldots, m \\
& (g_i(x) - u_i)h_i(x) \leq 0, \quad i = 1, \ldots, m.
\end{aligned}$$

Slack variables can also be used in this situation in any of the three forms outlined above.

3.2. **Modeling Languages.** Both GAMS [2] and AMPL [6] provide constructs for expressing complementarity constraints. Codes using Newton's method require second partial derivatives. Hence, the models used in our study are formulated in AMPL [6], which is currently the only modeling language providing second derivative information.

The complementarity operator, $\perp$, is implemented as the `complements` operator. Hence, (3) is expressed as:

(17)
$$\begin{aligned}
& \text{minimize} \quad f(x) \\
& \text{subject to} \quad l_1 \leq g(x) \leq u_1 \text{ complements } l_2 \leq h(x) \leq u_2,
\end{aligned}$$

where exactly two of the bounds are finite. However, in conveying such a problem to a solver, AMPL introduces a slack variable converting the complementarity constraint to

(18)
$$\begin{aligned}
& g(x) - w = 0, \\
& l_1 \leq w \leq u_1 \text{ complements } l_2 \leq h(x) \leq u_2.
\end{aligned}$$

This conversion does not take place if the complementarity relationship is between a variable and an expression, provided there is a unique

complementary variable for each constraint. More details on AMPL's handling of complementarity constraints is given in [3].

3.3. **Reformulation of Complementarity Constraints.** We return now to the formulation given by (1). In order to solve a problem of the form (1) using the algorithm described in the previous section, the complementarity constraint needs to be reformulated with simple inequalities. For ease of implementation, we take the approach of using one slack variable and express (2) equivalently as

$$(19) \quad \begin{aligned} g(x) - w &= 0 \\ w &\geq 0 \\ x &\geq 0 \\ -Wx &\geq 0, \end{aligned}$$

where W is the diagonal matrix whose entries are the elements of the vector $w \in \mathbb{R}^n$. We call (19) the *pairwise* formulation, as the last set of constraints state that the product of the individual pairs of complementary variables is less than or equal to 0.

Another way to express the complementarity constraints is by the *summation* formulation:

$$(20) \quad \begin{aligned} g(x) - w &= 0 \\ w &\geq 0 \\ x &\geq 0 \\ -w^T x &\geq 0, \end{aligned}$$

Note that due to the nonnegativity constraints, the solution to the summation formulation will satisfy the complementarity constraints but the two formulations lead to significantly different algorithmic performance. In fact, we have found that while the summation formulation improves performance on a few problems, in general, the pairwise formulation is faster and more stable. It is the pairwise formulation that we implement in the interior-point code.

Note that the constraints $-Wx \geq 0$ in the pairwise formulation and $-w^T x \geq 0$ in the summation formulation can also be expressed as equalities. However, in our numerical testing, we found that the equality formulation gave worse performance.

In the next section, we highlight some of the difficulties that arise when complementarity constraints are conveyed as (19) to an interior-point method. We propose remedies.

3.4. **Possible Difficulties.** The optimization problem with constraints of the form (19) is

$$
\begin{array}{rrcl}
\text{minimize} & f(x) & & \\
\text{subject to} & g(x) - w & = & 0 \\
& w & \geq & 0 \\
& x & \geq & 0 \\
& -Wx & \geq & 0.
\end{array}
$$

(21)

The first-order optimality conditions are:

$$
\begin{array}{rcl}
\nabla f(x) - \nabla g(x)v - z + W\lambda & = & 0 \\
v - y + X\lambda & = & 0 \\
g(x) - w & = & 0 \\
w - s & = & 0 \\
x - t & = & 0 \\
Wx + u & = & 0 \\
Sy - \mu e & = & 0 \\
Tz - \mu e & = & 0 \\
U\lambda - \mu e & = & 0
\end{array}
$$

(22)

where s, t, u are the slack variables and v, y, z, λ are the dual variables of appropriate dimension, and X is the diagonal matrix whose entries are the elements of the vector x.

Consider an optimal solution x^*. We partition the set of indices $\{1, \ldots, n\}$ into two sets, $\mathcal{I}$ and $\mathcal{J}$, such that

$$
x_{\mathcal{I}}^* > 0, \quad x_{\mathcal{J}}^* = 0.
$$

By complementarity, we have that $w_{\mathcal{I}}^* = 0$. If $w_{\mathcal{J}}^* > 0$, the solution is *strictly complementary*. Using the second and third equations in (22), we can eliminate v and w from the first equation. In block matrix form, the result is

$$
\begin{bmatrix} \nabla_{\mathcal{I}} f(x^*) \\ \nabla_{\mathcal{J}} f(x^*) \end{bmatrix} - \begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{I}} g_{\mathcal{J}}(x^*) \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{J}} g_{\mathcal{J}}(x^*) \end{bmatrix} \begin{bmatrix} y_{\mathcal{I}}^* \\ y_{\mathcal{J}}^* \end{bmatrix} - \begin{bmatrix} z_{\mathcal{I}}^* \\ z_{\mathcal{J}}^* \end{bmatrix} +
$$

(23)

$$
\begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) X_{\mathcal{I}}^* & 0 \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) X_{\mathcal{I}}^* & G_{\mathcal{J}}(x^*) \end{bmatrix} \begin{bmatrix} \lambda_{\mathcal{I}}^* \\ \lambda_{\mathcal{J}}^* \end{bmatrix} = 0,
$$

where $G(x)$ is the diagonal matrix with $G_{jj}(x) = g_j(x)$.

Suppose that the constraint gradients at x^* are linearly independent; that is

(24)

$$
\begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{I}} g_{\mathcal{J}}(x^*) \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{J}} g_{\mathcal{J}}(x^*) \end{bmatrix}
$$

is nonsingular. If, in addition, the solution $(x^*, g(x^*))$ is strictly complementary, then the matrix

$$(25) \qquad \begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) X_{\mathcal{I}}^* & 0 \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) X_{\mathcal{I}}^* & G_{\mathcal{J}}(x^*) \end{bmatrix}$$

is also nonsingular, as $x_{\mathcal{I}}^* > 0$ and $G_{\mathcal{J}}(x^*) > 0$. In this case, the nonzero Lagrange multipliers $y_{\mathcal{I}}$, $z_{\mathcal{J}}$, and λ lie on an unbounded n dimensional manifold (Fletcher, et.al. [5]). This can be seen by noting that the equation (23) can be broken up into the pair of equations:

$$\begin{bmatrix} \nabla_{\mathcal{I}} f(x^*) \\ \nabla_{\mathcal{J}} f(x^*) \end{bmatrix} - \begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{I}} g_{\mathcal{J}}(x^*) \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{J}} g_{\mathcal{J}}(x^*) \end{bmatrix} \begin{bmatrix} y_{\mathcal{I}}^* \\ y_{\mathcal{J}}^* \end{bmatrix} - \begin{bmatrix} z_{\mathcal{I}}^* \\ z_{\mathcal{J}}^* \end{bmatrix} \quad = \quad c$$

$$\begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) X_{\mathcal{I}}^* & 0 \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) X_{\mathcal{I}}^* & G_{\mathcal{J}}(x^*) \end{bmatrix} \begin{bmatrix} \lambda_{\mathcal{I}}^* \\ \lambda_{\mathcal{J}}^* \end{bmatrix} \quad = \quad -c,$$

for any vector c.

The fact that the optimal multipliers comprise an unbounded manifold could imply that a primal-dual interior-point code could have difficulty with problems with complementarity constraints. This supposed difficulty stems from the fact that interior-point methods tend to converge to the center of the feasible set, and, as shown in the previous section, primal-dual methods solve for both primal and dual variables simultaneously. However, as the results of the next section demonstrate, this is not a problem. While we have no explanation for the good behavior of the algorithm in computing dual variables, it is not the first time this phenomenon has been noted. The technique of splitting free variables into positive and negative components has long been used, with similar good computational performance.

However, a real problem occurs when $w_j = 0$ for some $j \in \{1, \dots, n\}$. When the optimal solution is not strictly complementary, we have that

$$(26) \qquad x_j^* = 0, \ g_j(x^*) = 0, \text{ and } w_j^* = 0.$$

Since the complementarity constraint is always active at optimality, we also have that

$$u_j^* = 0.$$

The effect of these three variables vanishing at optimality can be seen by examining the first-order conditions that involve the dual variables, λ:

$$(27) \qquad \begin{aligned} \nabla f(x) - \nabla g(x) v - z + W\lambda &= 0 \\ v - y + X\lambda &= 0 \\ U\lambda - \mu e &= 0 \end{aligned}$$

When $x_j^* = w_j^* = u_j^* = 0$, λ_j has a multiplier of 0 wherever it appears in the first-order conditions, and, therefore, can take on any value. In this case, λ_j goes to infinity while trying to find the center of the unbounded optimal face. This creates numerical difficulties for the solver, often resulting in less than satisfactory accuracy levels, such as a dual infeasibility of 10^{-4} or only 5 digits of agreement between the primal and the dual objective function values.

To overcome this difficulty, we monitor the values of complementary pairs $(x^{(k)}, w^{(k)})$ at each iteration k, along with the associated dual variables $\lambda^{(k)}$. In nonstrictly complementary problems, as optimality is approached, we can detect that at least one of the complementary pairs will have both components go to 0 and the dual variable going to infinity. The exact criteria that is implemented in the code is to label the jth pair *nonstrictly complementary* if

$$x_j^{(k)} \le 10^{-3}, w_j^{(k)} \le 10^{-3}, \lambda_j^{(k)} \ge 10^4.$$

Generally the complementary variables have much smaller values. When we detect this situation, we fix

$$x_j = 0, w_j = 0, \lambda_j = 0$$

for all subsequent iterations. We also fix the slack variables associated with these variables and their complementarity constraint to 0; that is,

$$t_j = 0, s_j = 0, u_j = 0.$$

Then, we eliminate from the reduced KKT system (9) the columns corresponding to the variables x_j, w_j, λ_j, and the rows corresponding to the constraints

$$
\begin{array}{rcl}
x_j - t_j & = & 0 \\
w_j - s_j & = & 0 \\
w_j x_j + u_j & = & 0.
\end{array}
$$

We assume that nonstrict complementarity arises near the local optimum, so we set the barrier parameter, μ, to 0 as well. When nonstrict complementarity is detected and the associated variables and the barrier parameter are fixed to 0, the code generally takes several more iterations to readjust the values of the other variables in the problem and reach an optimal solution.

Another important issue in solving MPECs using interior-point methods is the initialization of the variables involved in the complementarity constraint. For the system of the form

$$
\begin{array}{rcl}
x & \ge & 0 \\
w & \ge & 0 \\
-Wx & \ge & 0,
\end{array}
$$

sometimes (and quite often for the problems in our test set), the problem is formulated without providing initial values for the variables x. There is also the question of providing initial values for the newly-introduced variables w. In the case of a general NLP, we assume that both sets of variables start at 0. For an MPEC, however, such a point presents a problem when solving with an interior-point method. Not only are the variables initialized at their bounds, but we also have nonstrict complementarity at the initial solution.

Since the variables w are introduced into the problem by the reformulation in the interior-point method, we need to initialize those variables internally. They serve as a type of slack variable in our problem since we set

$$g(x) - w = 0.$$

Therefore, we can initialize them as we do with the other slack variables. The value for the initialization is determined by the parameter `bndpush` in LOQO, which is set to 1 by default. However, the user can change this value, or "tune" the code, by specifying, for example,

```
option loqo_options "bndpush=0.1"
```

in the AMPL model for the problem.

As for the variables x, the user may not have provided an initial solution, or an uninformed user may have provided an initial solution of 0. In such cases, we set those components of x equal to the value of `bndpush` as well.

As we discussed at the beginning of this section, there are several ways to formulate a complementarity constraint. It is sometimes the case that the user has chosen to express the AMPL model using one or more sets of additional variables or that AMPL has introduced these additional variables due to its format of communicating with the solvers. For example, a constraint of the form

$$(28) \qquad\qquad 0 \le h(x) \perp x \le 0$$

can be expressed by the user as

$$(29) \qquad \begin{aligned} h(x) - r \;&=\; 0 \\ 0 \le r \perp x \;&\le\; 0. \end{aligned}$$

In this case, the information that is passed from AMPL to the solver is that an "expression" r is complementary to the variable x. The solver, however, does not know that this expression consists only of the slack

variable r and proceeds to transform (29) into

$$
\begin{array}{rcl}
h(x) - r &=& 0 \\
r - w &=& 0 \\
w &\geq& 0 \\
x &\geq& 0 \\
-Wx &\geq& 0.
\end{array}
$$

This introduces another set of slacks into the problem, which may cause LOQO to take longer and even affect the accuracy level that can be achieved. Also, the appropriate initialization of all of these variables can be a problem as discussed above. In the next section, we will discuss several models where the additional variables r have an adverse affect on the performance of the solver.

Before moving on to the numerical results, we discuss briefly the implications of some of these issues for the summation formulation. In this case, the problem we are considering is

(30)
$$
\begin{array}{rrcl}
\text{minimize} & f(x) \\
\text{subject to} & g(x) - w &=& 0 \\
& w &\geq& 0 \\
& x &\geq& 0 \\
& -w^T x &\geq& 0.
\end{array}
$$

The first-order conditions for (30) are:

(31)
$$
\begin{array}{rcl}
\nabla f(x) - \nabla g(x)v - z + w\lambda &=& 0 \\
v - y + x\lambda &=& 0 \\
g(x) - w &=& 0 \\
w - s &=& 0 \\
x - t &=& 0 \\
wx + u &=& 0 \\
Sy - \mu e &=& 0 \\
Tz - \mu e &=& 0 \\
u\lambda - \mu &=& 0
\end{array}
$$

where u, the slack variable of the complementarity constraints, and λ, the dual variable of the complementarity constraint, are now scalars.

Using the second and third equations in (31), we can eliminate v and w from the first equation. In block matrix form, the system at

optimality is

$$
\begin{bmatrix} \nabla_{\mathcal{I}} f(x^*) \\ \nabla_{\mathcal{J}} f(x^*) \end{bmatrix} - \begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{I}} g_{\mathcal{J}}(x^*) \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) & \nabla_{\mathcal{J}} g_{\mathcal{J}}(x^*) \end{bmatrix} \begin{bmatrix} y_{\mathcal{I}}^* \\ y_{\mathcal{J}}^* \end{bmatrix} - \begin{bmatrix} z_{\mathcal{I}}^* \\ z_{\mathcal{J}}^* \end{bmatrix} +
$$

(32)

$$
\lambda^* \begin{bmatrix} \nabla_{\mathcal{I}} g_{\mathcal{I}}(x^*) & 0 \\ \nabla_{\mathcal{J}} g_{\mathcal{I}}(x^*) & I_{\mathcal{J}} \end{bmatrix} \begin{bmatrix} x_{\mathcal{I}}^* \\ w_{\mathcal{J}}^* \end{bmatrix} = 0,
$$

All of the possible difficulties outlined above are applicable to (32) as well. The only difference is that nonstrict complementarity may not have an adverse effect on the system when there is only one such pair. However, when there are multiple cases of nonstrict complementarity in the problem, we observe that numerical difficulties arise due to λ going to infinity.

4. Numerical Results

In this section, we present the numerical results of solving MPECs using LOQO(Version 6.04), which incorporates complementarity constraints in pairwise form with the remedies discussed in the previous section. The AMPL test problems come from the MacMPEC suite compiled by Leyffer [7]. The solver was compiled with the AMPL solver interface Version 20010823.

Numerical results on the full test set are given in the Appendix. The formulation used in these results is the pairwise formulation with inequality constraints as given in (21). Results for the summation formulation and for using equality constraints can be obtained from tables on the authors' website [1]. We provide here a discussion of some of the interesting cases, highlighting the issues presented in the previous section.

In the tables, "†" indicates when the problem has a nonstrictly complementary solution and the associated dual variables are going to infinity. As mentioned in the previous section, we detect such cases and eliminate all affiliated variables and constraints from the problem. Doing so, LOQO solves the following problems: *ex9.2.2*, *qpec2*, *ralph1*, *scholtes4*, *tap-09*, and *tap-15*, which were unsolvable before.

In the tables, problems marked with "*" are instances when LOQO was *tuned*, or needed one or more of its settings changed from the defaults. In these cases, the modified setting was the initial value of the slack variables in the problem. The default value is 1, but in most cases where tuning was required, the slacks were initialized at 0.1. For two problems, *bar_truss_3* and *pack-rig1p-32*, the initial value was 0.01. The problem of proper slack variable initialization is an important one

in solving MPECs, and one that requires further study for general NLPs as well.

As discussed in the previous section, user-defined slack variables may prevent attaining the desired level of accuracy or complicate the issue of finding a correct initialization. In the problem *bem-milanc30-s*, the model is expressed in the form

$$(33) \quad \begin{aligned} \text{minimize} \quad & f(x, y) \\ \text{subject to} \quad & g(x) - s = 0 \\ & 0 \leq s \perp y \geq 0. \end{aligned}$$

With this formulation, LOQO is unable to attain more that 6 digits of agreement between the primal and the dual solutions. If the variable s is removed resulting in the problem

$$(34) \quad \begin{aligned} \text{minimize} \quad & f(x, y) \\ \text{subject to} \quad & 0 \leq g(x) \perp y \geq 0, \end{aligned}$$

LOQO finds a solution quite easily in 27 iterations.

In the case of the packing problems, there are two expressions that are complementary to each other; that is, constraints of the form (12). As discussed in the previous section, there are three ways to approach such problems, each giving a very different numerical performance. We illustrate the results with the example *pack-comp1-32*.

- Without slack variables: For each complementarity pair, the information provided from AMPL to the solver has one variable and one expression, as given in (18). Therefore, using the `complements` operator to express *pack-comp1-32* without slack variables as in (13) will only result in AMPL's converting the problem to the form (14). In order to observe the behavior of the algorithm in the absence of slack variables, we have coded a new problem in the standard NLP form of (6) with constraints of the form

$$(35) \quad \begin{aligned} & g(x) \geq 0 \\ & h(x) \geq 0 \\ & -g_i(x)h_i(x) \geq 0, \quad i = 1, \dots, m \end{aligned}$$

  written explicitly in the model. LOQO can solve this problem easily in 72 iterations and 62.21 seconds.
- With one slack variable: The version of the problem that appears in the test set is of the form (14), with an additional variable for $g(x)$. LOQO fails on this problem when running with default settings, but when the slacks are initialized at 0.1, it solves this problem in 66 iterations and 46.01 seconds. If, instead, the additional variable is for $h(x)$, LOQO again requires

that the slacks be initialized at 0.1 and solves the problem in 67 iterations and 44.50 seconds.

- With two slack variables: We have also made a version of this problem in the form (15). LOQO cannot find a solution to this problem on the defaults. When the slack variables are initialized to 0.1, it cannot get more than 7 digits of agreement between the primal and the dual objective functions. When the initialization is changed to 0.01, LOQO reaches an optimal solution in 59 iterations and 46.52 seconds. However, we should note that LOQO struggles with achieving the default accuracy levels.

It is important to note that the only true failures of LOQO on this test set are not due to complications arising from the complementarity constraints. There are only a few such problems: *ralphmod*, *siouxfls*, *siouxfls1*, *water-net* and *water-FL*. In the first problem, there is an unbounded face of feasible solutions. The rest of these problems are rank-deficient, which can create problems for an interior-point code.

In the numerical results, a comparison with the optimal objective function values reported by the authors of the suite is also provided. In several cases, LOQO finds a better optimal solution than the one previously reported. Moreover, in the case of *design-cent-3*, which is reported to be an infeasible problem, LOQO finds an optimal solution where the primal infeasibility level is less than 10^{-8}.

References

[1] H. Y. Benson. Full numerical results on the MacMPEC test suite of problems. http://www.princeton.edu/~hyurttan/mpecs.html.

[2] A. Brooke, D. Kendrick, and A. Meeraus. *GAMS: A User's Guide*. Scientific Press, 1988.

[3] M. C. Ferris, R. Fourer, and D. M. Gay. Expressing complementarity problems in an algebraic modeling language and communicating them to solvers. *SIAM Journal on Optimization*, 9:991–1009, 1999.

[4] M. C. Ferris and J. S. Pang. Engineering and economic applications of complementarity problems. *SIAM Review*, 39(4):669–713, 1997.

[5] R. Fletcher, S. Leyffer, D. Ralph, and S. Scholtes. Local convergence of sqp methods for mathematical programs with equilibrium constraints. Technical Report NA-209, University of Dundee, May 2002.

[6] R. Fourer, D.M. Gay, and B.W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Scientific Press, 1993.

[7] S. Leyffer. MacMPEC AMPL collection of mathematical programs with equilibrium constraints. http://www.maths.dundee.ac.uk/~sleyffer/MacMPEC/index.html.

[8] R.J. Vanderbei and D.F. Shanno. An interior-point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.

HANDE Y. BENSON, PRINCETON UNIVERSITY, PRINCETON, NJ

DAVID F. SHANNO, RUTGERS UNIVERSITY, NEW BRUNSWICK, NJ

ROBERT J. VANDERBEI, PRINCETON UNIVERSITY, PRINCETON, NJ

Problem	n	m	iter	time	f^*	f_r^*
bar-truss-3 *	41	40	30	0.06	1.0167E+04	1.0167E+04
bard1	8	7	14	0.01	1.7000E+01	1.7000E+01
bard2	15	12	34	0.03	6.5980E+03	-6.5980E+03
bard3	7	6	20	0.03	-1.2679E+01	-1.2679E+01
bard1m	9	7	15	0.02	1.7000E+01	1.7000E+01
bard2m	15	12	20	0.02	-6.5980E+03	-6.5980E+03
bard3m	9	8	23	0.02	-1.2679E+01	-1.2679E+01
bem-milanc30-s	4900	4897	27	11.03	1.0297E+03	8.3283E+01
bilevel1	16	15	30	0.04	5.0000E+00	-6.0000E+01
bilevel2	24	21	28	0.05	-6.6000E+03	-6.6000E+03
bilevel3	14	13	21	0.03	-1.2679E+01	-1.2679E+01
bilin	14	13	37	0.05	4.4000E+00	5.6000E+00
dempe	4	3	17	0.02	3.1250E+01	3.1250E+01
design-cent-1	15	14	16	0.02	1.8606E+00	1.8607E+00
design-cent-2	16	18	16	0.02	3.4838E+00	3.4838E+00
design-cent-3	18	14	24	0.07	3.7234E+00	(I)
design-cent-4	30	28	13	0.02	3.0792E+00	3.0792E+00
desilva	8	6	26	0.03	-1.0000E+00	-1.0000E+00
df1	3	4	28	0.02	1.3245E-10	0.0000E+00
ex9.1.1	18	17	13	0.02	-1.3000E+01	-6.0000E+00
ex9.1.2	10	10	11	0.01	-6.2500E+00	-6.2500E+00
ex9.1.3	29	27	32	0.05	-2.3000E+01	-6.0000E+00
ex9.1.4	10	9	15	0.02	-3.7000E+01	-3.7000E+01
ex9.1.5	18	17	13	0.02	-1.0000E+00	-1.0000E+00
ex9.1.6	20	19	23	0.04	-4.9000E+01	-1.5000E+01
ex9.1.7	23	21	50	0.09	-6.0000E+00	-6.0000E+00
ex9.1.8	14	12	13	0.02	-3.2500E+00	-3.2500E+00
ex9.1.9	17	16	31	0.05	3.1111E+00	3.1111E+00
ex9.1.10	14	12	13	0.01	-3.2500E+00	-3.2500E+00
ex9.2.1	14	13	13	0.02	1.7000E+01	2.5000E+01
ex9.2.2 †	9	9	36	0.03	1.0000E+02	1.0000E+02
ex9.2.3	18	17	17	0.02	5.0000E+00	-5.5000E+01

TABLE 1. Numerical performance of LOQO on the macMPEC set of complementarity constrained problems. n is the number of variables, m is the number of constraints, $iter$ is the number of iterations, $time$ is the runtime in CPU seconds, f^* is the optimal objective function value reported by LOQO, and f_r^* is the optimal objective function value reported by the authors of macMPEC. * indicates that the LOQO was tuned and † indicates that the problem was nonstrictly complementary. (I) indicates that there is no known feasible solution to the problem.

Problem	n	m	iter	time	f^*	f_r^*
ex9.2.4	10	9	11	0.02	5.0000E-01	5.0000E-01
ex9.2.5	11	10	13	0.01	5.0000E+00	9.0000E+00
ex9.2.6	22	18	27	0.04	-1.0000E+00	-1.0000E+00
ex9.2.7	14	13	13	0.02	1.7000E+01	2.5000E+01
ex9.2.8	8	7	11	0.01	1.5000E+00	1.5000E+00
ex9.2.9	12	11	12	0.01	2.0000E+00	2.0000E+00
gnash10	21	20	24	0.04	-2.3082E+02	-2.3082E+02
gnash11	21	20	24	0.03	-1.2991E+02	-1.2991E+02
gnash12	21	20	23	0.03	-3.6933E+01	-3.6933E+01
gnash13	21	20	23	0.04	-7.0618E+00	-7.0618E+00
gnash14	21	20	29	0.04	-1.7905E-01	-1.7905E-01
gnash15	21	20	53	0.08	-3.5470E+02	-3.5470E+02
gnash16	21	20	44	0.05	-2.4144E+02	-2.4144E+02
gnash17	21	20	34	0.05	-9.0749E+01	-9.0749E+01
gnash18	21	20	32	0.05	-2.5698E+01	-2.5698E+01
gnash19	21	20	28	0.04	-6.1167E+00	-6.1167E+00
gauvin	5	4	18	0.01	2.0000E+01	2.0000E+01
hakonsen	13	12	22	0.03	2.4367E+01	2.4367E+01
hs044-i	30	24	30	0.05	1.7090E+01	1.5618E+01
incid-set1-8	166	168	23	0.63	6.7671E-09	3.8160E-17
incid-set1-16	710	716	66	11.01	1.1698E-09	1.2060E-16
incid-set1-32 *	2950	2964	410	465.41	2.8223E-07	2.7990E-07
incid-set1c-8	166	175	23	0.52	6.3889E-10	3.8160E-17
incid-set1c-16	710	731	75	11.19	1.0035E-09	1.2060E-16
incid-set1c-32 *	2950	2995	164	185.92	9.8167E-06	9.8170E-06
incid-set2-8	166	168	31	0.83	4.5179E-03	4.5180E-03
incid-set2-16	710	716	59	9.31	3.2169E-03	3.0360E-03
incid-set2-32	2950	2964	151	165.56	1.7681E-03	1.7380E-03
incid-set2c-8	166	175	28	0.61	5.4713E-03	5.4710E-03
incid-set2c-16	710	731	59	9.14	3.5995E-03	3.6020E-03
incid-set2c-32	2950	2995	456	485.29	2.4358E-03	2.4330E-03
jr1	3	2	9	0.01	5.0000E-01	5.0000E-01

TABLE 2. Numerical performance of LOQO on the macMPEC set of complementarity constrained problems. n is the number of variables, m is the number of constraints, $iter$ is the number of iterations, $time$ is the runtime in CPU seconds, f^* is the optimal objective function value reported by LOQO, and f_r^* is the optimal objective function value reported by the authors of macMPEC. * indicates that the LOQO was tuned and † indicates that the problem was nonstrictly complementary.

Problem	n	m	iter	time	f^*	f_r^*
jr2	3	2	11	0.01	5.0000E-01	5.0000E-01
kth1	3	2	9	0.01	1.3830E-08	0.0000E+00
kth2	3	2	9	0.01	7.2660E-10	0.0000E+00
kth3	3	2	11	0.01	5.0000E-01	5.0000E-01
liswet1-050	202	153	20	0.12	1.3994E-02	1.3990E-02
liswet1-100	402	303	23	0.25	1.3734E-02	1.3730E-02
liswet1-200	802	603	33	0.72	1.7009E-02	1.7010E-02
nash1	8	6	14	0.01	9.9454E-18	7.8886E-30
outrata31	9	8	11	0.01	2.8827E+00	3.2077E+00
outrata32	9	8	12	0.01	3.1755E+00	3.4494E+00
outrata33	9	8	16	0.02	2.8881E+00	4.6043E+00
outrata34	9	8	15	0.02	5.7892E+00	6.5927E+00
pack-comp1-8	156	170	40	0.71	6.0000E-01	6.0000E-01
pack-comp1-16 *	692	736	31	2.78	6.1695E-01	6.1695E-01
pack-comp1-32 *	2916	3068	66	44.59	6.5298E-01	6.5298E-01
pack-comp1c-8	156	177	33	0.55	6.0000E-01	6.0000E-01
pack-comp1c-16 *	692	751	32	2.58	6.2304E-01	6.2304E-01
pack-comp1c-32 *	2916	3099	39	19.06	6.6144E-01	6.6144E-01
pack-comp2-8	156	170	39	0.8	6.7312E-01	6.7312E-01
pack-comp2-16 *	692	736	28	2.65	7.2714E-01	7.2714E-01
pack-comp2-32 *	2916	3068	54	33.55	7.8260E-01	7.8260E-01
pack-comp2c-8	156	177	19	0.3	6.7346E-01	6.7346E-01
pack-comp2c-16 *	692	751	27	2.03	7.2747E-01	7.2747E-01
pack-comp2c-32 *	2916	3099	40	19.81	7.8294E-01	7.8294E-01
pack-rig1-8	119	118	25	0.3	7.8793E-01	7.8793E-01
pack-rig1-16	538	537	70	5.17	8.2601E-01	8.2601E-01
pack-rig1-32 *	2330	2329	76	37.91	8.5089E-01	8.5090E-01
pack-rig1c-8	119	125	24	0.31	7.8830E-01	7.8830E-01
pack-rig1c-16	538	552	38	2.54	8.2650E-01	8.2650E-01
pack-rig1c-32 *	2330	2360	48	18.65	8.5164E-01	8.5164E-01
pack-rig1p-8	152	151	23	0.3	7.8793E-01	7.8793E-01
pack-rig1p-16	648	647	64	5.46	8.2601E-01	8.2601E-01

TABLE 3. Numerical performance of LOQO on the macMPEC set of complementarity constrained problems. n is the number of variables, m is the number of constraints, *iter* is the number of iterations, *time* is the runtime in CPU seconds, f^* is the optimal objective function value reported by LOQO, and f_r^* is the optimal objective function value reported by the authors of macMPEC. * indicates that the LOQO was tuned and † indicates that the problem was nonstrictly complementary.

Problem	n	m	iter	time	f^*	f_r^*
pack-rig1p-32 *	2716	2715	58	32.28	8.5089E-01	8.5090E-01
pack-rig2-8	115	114	24	0.29	7.8040E-01	7.8040E-01
pack-rig2-16 †	524	523			(IL)	(I)
pack-rig2-32	2241	2240			(IL)	(I)
pack-rig2c-8	115	121	27	0.32	7.9931E-01	7.9931E-01
pack-rig2c-16	524	538			(IL)	(I)
pack-rig2c-32	2241	2271			(IL)	(I)
pack-rig2p-8	148	147	24	0.33	7.8040E-01	7.8040E-01
pack-rig2p-16	630	629	57	4.91	1.0851E+00	1.08514 (?)
pack-rig2p-32 †	2622	2621			(IL)	1.13589 (?)
portfl-i-1	99	37	40	0.16	1.5031E-05	1.5020E-05
portfl-i-2	99	37	39	0.16	1.4581E-05	1.4570E-05
portfl-i-3	99	37	46	0.18	6.2673E-06	6.2650E-06
portfl-i-4	99	37	37	0.15	2.1804E-06	2.1770E-06
portfl-i-6	99	37	33	0.13	2.3760E-06	2.3610E-06
qpec-100-1	205	202	18	1.09	9.9003E-02	9.9003E-02
qpec-100-2	210	202	20	1.36	-6.4452E+00	-6.2605E+00
qpec-100-3	210	204	20	1.27	-5.4817E+00	-5.4829E+00
qpec-100-4	220	204	20	1.46	-3.9821E+00	-3.6007E+00
qpec-200-1	410	404	24	8.31	-1.9348E+00	-1.9348E+00
qpec-200-2	420	404	25	11.58	-2.3987E+01	-2.4030E+01
qpec-200-3	420	408	22	8.44	-1.9534E+00	-1.9534E+00
qpec-200-4	440	408	24	12.65	-6.2305E+00	-6.1932E+00
qpec1	50	40	11	0.02	8.0000E+01	8.0000E+01
qpec2 †	50	40	32	0.06	4.5000E+01	4.5000E+01
ralph1 †	3	2	106	0.08	8.9483E-81	0.0000E+00
ralph2	3	2	30	0.03	2.4214E-11	0.0000E+00
ralphmod	204	200			(IL)	-6.8303E+02
scholtes1	4	2	10	0.01	2.0000E+00	2.0000E+00
scholtes2	4	2	11	0.02	1.5000E+01	1.5000E+01
scholtes3	3	2	9	0.01	5.0000E-01	5.0000E-01
scholtes4 †	4	4	51	0.04	-1.8672E-09	-3.0734E-07

TABLE 4. Numerical performance of LOQO on the macMPEC set of complementarity constrained problems. n is the number of variables, m is the number of constraints, *iter* is the number of iterations, *time* is the runtime in CPU seconds, f^* is the optimal objective function value reported by LOQO, and f_r^* is the optimal objective function value reported by the authors of macMPEC. * indicates that the LOQO was tuned and † indicates that the problem was nonstrictly complementary. (I) indicates that there is no known feasible solution to the problem, (IL) indicates that LOQO reached its iteration limit, and (?) indicates that the reported solution may not be correct.

Problem	n	m	iter	time	f^*	f_r^*
scholtes5	5	4	10	0.01	1.0000E+00	1.0000E+00
siouxfls	4151	4124			(IL)	-2.0826E+02
siouxfls1	4151	4124			(IL)	9.7939E+02
sl1	11	8	32	0.03	1.0000E-04	1.0000E-04
stackelberg1	4	3	17	0.01	-3.2667E+03	-3.2667E+03
tap-09 †	118	100	154	1.27	2.0917E+02	1.0914E+02
tap-15 †	277	250	209	5.73	2.4049E+02	1.8430E+02
water-net	80	64			(IL)	9.2917E+02
water-FL	257	204			(IL)	3.4119E+03

TABLE 5. Numerical performance of LOQO on the macMPEC set of complementarity constrained problems. n is the number of variables, m is the number of constraints, $iter$ is the number of iterations, $time$ is the runtime in CPU seconds, f^* is the optimal objective function value reported by LOQO, and f_r^* is the optimal objective function value reported by the authors of macMPEC. * indicates that the LOQO was tuned and † indicates that the problem was nonstrictly complementary. (IL) indicates that LOQO reached its iteration limit.