SUBSPACE ACCELERATED MATRIX SPLITTING ALGORITHMS FOR BOUND-CONSTRAINED QUADRATIC PROGRAMMING AND LINEAR COMPLEMENTARITY PROBLEMS *

DANIEL P. ROBINSON†, LIMING FENG‡, JORGE M. NOCEDAL§, AND JONG-SHI PANG¶

Abstract. This paper studies the solution of two problems—bound-constrained quadratic programs and linear complementarity problems—by two-phase methods that consist of an active set prediction phase and a subspace phase. The algorithms enjoy favorable convergence properties under weaker assumptions than those assumed for other methods in the literature. The active set prediction phase employs matrix splitting iterations that are tailored to the structure of the (nonconvex) bound-constrained problems and the (asymmetric) linear complementarity problems studied in this paper. Numerical results on a variety of test problems illustrate the performance of the methods.

Key words. quadratic programming, linear complementarity, iterative methods, Jacobi iteration, Gauss-Seidel iteration, splitting methods, two-phase methods, American options pricing

AMS subject classifications. 49M05, 49M15, 65K05, 65K10, 65K15, 91G60

1. Introduction.

This paper considers two-phase splitting methods for solving a linear complementarity problem (LCP) and a bound-constrained quadratic program (BQP). The LCP is to find a vector x satisfying

$$\text{LCP}(q,M) \qquad x \geq 0, \;\; Mx + q \geq 0, \;\; \text{and} \;\; (Mx+q) \circ x = 0,$$

where $M \in \mathbb{R}^{n \times n}$ and $q \in \mathbb{R}^n$ are given, and we use the notation $[a \circ b]_i \triangleq a_i b_i$. (See Cottle et al. [4] for an extensive study of LCPs.) The bound-constrained quadratic program is given by

$$\text{BQP}(q,M) \qquad \underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; f(x) \triangleq \tfrac{1}{2}\langle x, Mx\rangle + \langle q, x\rangle \quad \text{subject to} \;\; x \geq 0$$

where the matrix M is now assumed to be symmetric and $\langle \cdot, \cdot \rangle$ represents the standard inner-product on $\mathbb{R}^n$; we note that our algorithms may be extended in obvious ways to handle the more general bounds $x_\ell \leq x \leq x_u$. If M is symmetric and positive definite, problems LCP(q,M) and BQP(q,M) are equivalent; if M is merely symmetric, then problem LCP(q,M) is equivalent to finding a first-order solution to problem BQP(q,M); if M is asymmetric, there is no convenient relationship between these two problems. It is precisely for these reasons that we present two algorithms; the first designed to solve BQP(q,M) and the second to solve asymmetric LCP(q,M).

The methods we propose consist of two phases. In the first phase, we use iterations based on matrix splittings of M to predict those variables that will be equal to zero at a solution. The manner in which we use these iterates to guarantee convergence is dictated by the inherent nature of problems BQP(q,M) and LCP(q,M). In the second phase, we use the prediction afforded by the first phase to formulate a subproblem whose solution accelerates convergence.

The work by Moré and Toraldo [12] demonstrates that two-phase approaches for solving problem BQP(q,M) are often very effective. Their algorithm uses a projected gradient search in the first phase to predict the optimal active set, and then a conjugate gradient based subspace phase to accelerate

*Research supported by the National Science Foundation grants CMMI–0927367, CMMI–1029846, and CMMI–1030540, and the ISEN Institute at Northwestern University.

†Industrial Engineering and Management Sciences, Northwestern University, Technological Institute, 2145 Sheridan Road, Evanston, IL 60208-3118, (daniel.p.robinson@gmail.com).

‡117 Transportation Building MC-238, 104 S. Mathews Avenue, Urbana IL 61801, (fenglm@illinois.edu).

§Industrial Engineering and Management Sciences, Northwestern University, Technological Institute, 2145 Sheridan Road, Evanston, IL 60208-3118, (nocedal@eecs.northwestern.edu).

¶117 Transportation Building MC-238, 104 S. Mathews Avenue, Urbana IL 61801, (jspang@illinois.edu).

convergence. Kočvara and Zowe [9] also use a two-phase approach for solving BQP(q,M), although they assume that M is positive definite. One of their primary contributions is the use of more sophisticated iterations to approximate the active set at the solution, which include variants of successive overrelaxation and the linear conjugate gradient method. We view our algorithm for solving BQP(q,M) as an improvement over both of these works. First, in contrast to [9], our algorithm is globally convergent—under certain assumptions—even when problem BQP(q,M) is nonconvex. Second, our work may be viewed as a generalization of [12] since a projected gradient iteration is a specific instance—in fact the most basic—of our general matrix splitting iteration. We will show that this added freedom allows for the use of more sophisticated matrix splittings and leads to improved optimal active set identification.

It has been demonstrated by Feng et al. [6] and Morales, Nocedal, and Smelyanskiy [11] that some linear complementarity problems arising in finance and computational mechanics can also be solved efficiently with two-phase methods. For pricing American options, the complementarity problem is asymmetric, while frictionless contact problems arising in mechanics give rise to symmetric positive-semidefinite systems. For both of these applications, the methods proposed in [6] have no convergence guarantees and thus it is natural to ask whether an efficient two-phase algorithm with convergence guarantees can be designed. Our motivation for this work goes, however, beyond the applications in finance and mechanics mentioned above. Our key contribution in this area is the development of a provably convergent two-phase general purpose method that is efficient at solving the problems considered in [6] as well as more general problems.

To validate the effectiveness of our algorithm for solving BQP(q,M), we present results on problems from the CUTEr [1] test set and on randomly generated strictly convex BQPs. For our LCP algorithm, we first consider the asymmetric problems arising in the pricing of American options. We show that our method performs *exactly* the same as the algorithm from [6] whose performance is excellent. We then proceed to show that our algorithm is effective on a class of randomly generated asymmetric LCP problems, and in the process showcase the effectiveness of the subspace phase.

Preliminaries. We use matrix splittings of the form

$$M = B + C. \tag{1.1}$$

Given such a splitting, we define the following *fixed-point iteration* algorithm, which iteratively solves the following sub-LCP: find x satisfying

$$\text{LCP}(q + Cx^k, B) \qquad x \geq 0, \;\; Bx + q + Cx^k \geq 0, \;\; \text{and} \;\; (Bx + q + Cx^k) \circ x = 0.$$

(Note that if x^k solves LCP($q + Cx^k$,B) then x^k also solves LCP(q,M).)

Algorithm 1.1. Fixed-point iteration.
Input: starting point $x^0 \in \mathbb{R}^n$ and number of iterations m.
for $k = 0, 1, \ldots, m-1$
  Compute a solution x^{k+1} to problem LCP($q + Cx^k$,B).
end
Output: x^m

In Algorithm 1.1 we assume that each LCP($q + Cx^k$,B) has at least one solution, but in general we do not presume that there is a unique solution. From a practical point of view, we assume that the splitting is chosen such that problem LCP($q + Cx^k$,B) is numerically inexpensive to solve. Throughout the paper, we use the notation

$$y = \text{FPI}(x, p, B, C) \tag{1.2}$$

to mean that y is the output from the **F**ixed-**P**oint **I**teration Algorithm 1.1 with starting point x, number of iterations p, and matrix splitting $M = B + C$. If we let L denote the strictly lower-triangular part of M, U the strictly upper-triangular part of M, and D the diagonal part of M so that $M = L + D + U$, then Table 1.1 contains popular matrix splittings and the resulting method.

TABLE 1.1

Common matrix splittings and the resulting algorithms. The relaxation parameter satisfies $0 < \omega < 2$.

Matrix B	Matrix C	Resulting Method
I	$M - I$	Projected gradient
D	$L + U$	Projected Jacobi
$D + L$	U	Projected Gauss-Seidel
$\frac{1}{\omega}D + L$	$(1 - \frac{1}{\omega})D + U$	Projected Successive Overrelaxation

Our method for solving BQP(q,M) relies on minimizing f along so-called projected paths. To make this precise, we define the following projected search algorithm.

Algorithm 1.2. Projected search.
Input: base point $0 \le x \in \mathbb{R}^n$ and search direction d.
Compute α^* as the smallest solution of

$$\underset{\alpha \ge 0}{\text{minimize}}\ f\big(\max(x + \alpha d, 0)\big) \qquad \text{(maximum is component-wise).}$$

Output: $y = \max(x + \alpha^* d, 0)$

We use the notation

$$y = \mathrm{PS}(x, d)$$

to mean that y is the output from a **P**rojected **S**earch as described in Algorithm 1.2 for a given base point x and direction d. Notice that Algorithm 1.2 may produce $\alpha^* = \infty$, in which case the output y satisfies

$$y_i = \begin{cases} 0 & \text{if } i \in \mathcal{N}(d) \\ x_i & \text{if } i \in \mathcal{Z}(d) \\ \infty & \text{if } i \in \mathcal{P}(d) \end{cases}$$

where

$$\mathcal{N}(d) = \{i : d_i < 0\}, \quad \mathcal{Z}(d) = \{i : d_i = 0\}, \quad \text{and} \quad \mathcal{P}(d) = \{i : d_i > 0\}.$$

Moreover, this implies that $\lim_{\alpha\to\infty} f\big(x(\alpha)\big) = -\infty$ along the "feasible ray"

$$x_i(\alpha) = \begin{cases} 0 & \text{if } i \in \mathcal{N}(d) \\ x_i & \text{if } i \in \mathcal{Z}(d) \\ x_i + \alpha d_i & \text{if } i \in \mathcal{P}(d) \end{cases} \tag{1.3}$$

for all $\alpha \ge 0$.

Notation. Given a vector v, a matrix V, and an indexing set S, the notation v_S and V_S will denote the rows of v and the rows and columns of V that correspond to the indices in S. We use $[v]^+ = \max(v, 0)$, where the maximum is understood to be component-wise, $V \succ 0$ to mean that V is a positive-definite (not necessarily symmetric) matrix, and $V \succeq 0$ to mean that V is a positive-semidefinite matrix. We denote the inner-product of two n-dimensional vectors x and y as $\langle x, y \rangle \triangleq \sum_{i=1}^{n} x_i y_i$. Finally, given three real numbers r_1, r_2, and r_3, let $\text{med}(r_1, r_2, r_3)$ denote the median of the numbers r_1, r_2, and r_3.

2. Bound-constrained quadratic programming. Consider first the bound-constrained problem BQP(q,M) introduced on page 1. In Section 2.1 we describe and state our algorithm, in Section 2.2 prove that it is globally convergent, and in Section 2.3 provide some numerical results.

2.1. The algorithm. In this section we describe each step of our method, which is given as Algorithm 2.1 and depicted in Figure 2.1. Let the current iterate be $x^k \geq 0$. If x^k is optimal, we exit in Step 1; otherwise, we proceed to Step 2 and compute a so-called Cauchy step

$$x^{k,c} = x^k + \alpha^{k,c} d^{k,c}, \tag{2.1}$$

where

$$\alpha^{k,c} = \operatorname*{argmin}_{\alpha \geq 0} \; f(x^k + \alpha d^{k,c}) \;\; \text{subject to} \;\; x^k + \alpha d^{k,c} \geq 0 \tag{2.2}$$

and

$$d^{k,c} \triangleq \text{FPI}(x^k, 1, B, C) - x^k \neq 0. \tag{2.3}$$

(The fact that $d^{k,c} \neq 0$ is a consequence of Lemma 2.1 and the fact that we did not terminate in Step 1.) We note that if $\alpha^{k,c} = \infty$, then we must have that $\lim_{\alpha \to \infty} f(x^k + \alpha d^{k,c}) = -\infty$. This follows from the observations that (i) if $\alpha^{k,c} = \infty$, then $d^{k,c} \geq 0$ since otherwise the constraint in (2.2) would be violated for all α sufficiently large; (ii) Lemma 2.1 ensures that $d^{k,c}$ is a descent direction for f since otherwise x^k would have been a first-order solution to BQP(q,M) and the algorithm would have terminated in Step 1; (iii) f is a quadratic function; and (iv) the solution $\alpha^{k,c}$ is chosen in Algorithm 1.2 to be the smallest nonnegative solution. Thus,

$$\alpha^{k,c} = \infty \quad \Longrightarrow \quad \lim_{\alpha \to \infty} f(x^k + \alpha d^{k,c}) = -\infty. \tag{2.4}$$

The results of Section 2.2 will show that the Cauchy point drives convergence of our method.

We compute $x^{k,f}$ in Step 3 by performing n_f additional fixed-point iterations with starting point $x^{k,c}$, i.e.,

$$x^{k,f} = \text{FPI}(x^{k,c}, n_f, B, C). \tag{2.5}$$

In Step 4 we estimate the variables that are active at a solution of BQP(q,M) by computing $x^{k,pf}$ as the first minimizer of f along the projected path starting from $x^{k,c}$ and leading towards $x^{k,f}$, i.e.,

$$x^{k,pf} = \text{PS}(x^{k,c}, x^{k,f} - x^{k,c}); \tag{2.6}$$

the estimate of the active set is given by

$$\mathcal{A}^k = \{i : x_i^{k,pf} = 0\}. \tag{2.7}$$

Note that if any component of $x^{k,pf}$ is infinite, then the argument leading to (1.3) shows that there exists a ray emanating from $x^{k,c}$ upon which f is unbounded below.

In Step 5, we compute $x^{k,s}$ by approximately solving

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ f(x) = \tfrac{1}{2}\langle x, Mx\rangle + \langle x, q\rangle \ \text{ subject to } x_{\mathcal{A}^k} = 0,\ \|x - x^{k,pf}\|_2 \le \Delta^k, \tag{2.8}$$

where $\Delta^k > 0$ is a trust-region radius used to ensure that problem (2.8) is well defined, and $x_{\mathcal{A}^k}$ are the components of x that correspond to the indexing set $\mathcal{A}^k$. We accept $x^{k,s}$ as an approximate solution to (2.8) provided it satisfies

$$f\big(x^{k,s}\big) \le f\big(x^{k,pf}\big),\ \ x_{\mathcal{A}^k} = 0,\ \text{ and } \|x^{k,s} - x^{k,pf}\|_2 \le \Delta^k. \tag{2.9}$$

In Step 6 we compute x^{k+1} by performing a projected search starting from the point $x^{k,pf}$ and heading towards $x^{k,s}$, i.e.,

$$x^{k+1} = \mathrm{PS}(x^{k,pf}, x^{k,s} - x^{k,pf}). \tag{2.10}$$

Similar to Step 4, if any component of x^{k+1} is infinite, then the argument leading to (1.3) shows that there exists a ray emanating from $x^{k,pf}$ upon which f is unbounded below.

Finally, we update the trust-region radius in Step 7 based on the distance from $x^{k,pf}$ to x^{k+1}. To be precise, we define

$$\Delta^{k+1} = \mathrm{med}\big(\eta_{\mathrm{c}}\Delta^k,\ \eta_{\mathrm{e}}\|x^{k+1} - x^{k,pf}\|_2,\ \Delta_{\max}\big), \tag{2.11}$$

which ensures that the new trust-region radius is bounded above by $\Delta_{\max}$ and adjusted according to our expected progress in the subspace phase as indicated by the size of $\|x^{k+1} - x^{k,pf}\|_2$. Note that if $x^{k+1} = x^{k,pf}$, then we decrease the trust-region radius by some contraction factor η_{c} with the hope that the subspace step makes progress during the next iteration.

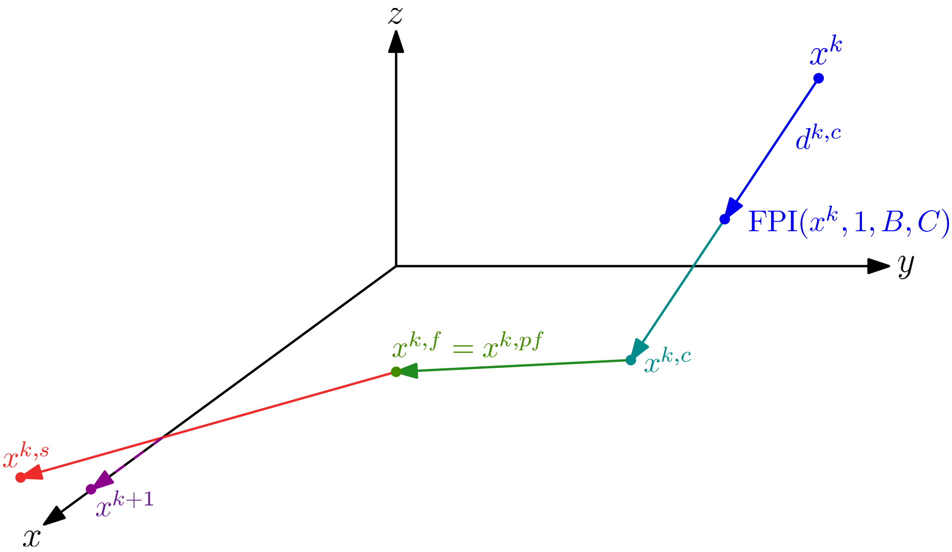
Algorithm 2.1. Algorithm for solving BQP(q,M).
Input: $x^0 \ge 0$, $n_{\max} \ge 0$, $0 < \eta_{\mathrm{c}} < 1 < \eta_{\mathrm{e}}$, $0 < \Delta_{\max}$, and $M = B + C$ with $B \succ 0$.
for $k = 0, 1, \ldots, n_{\max}$

1. **Check for optimality:** If x^k is a first-order solution to BQP(q,M), i.e., x^k solves LCP(q,M), then exit with first-order solution x^k.
2. **Cauchy step:** Compute FPI($x^k, 1, B, C$), define $d^{k,c}$ by (2.3), and solve (2.2) for $\alpha^{k,c}$. If $\alpha^{k,c} = \infty$ then exit; otherwise, define $x^{k,c}$ from (2.1).
3. **Additional fixed-point iterations:** Choose $n_f \ge 0$ and then compute $x^{k,f}$ by (2.5).
4. **Projected search on fixed-point iteration direction:** Compute $x^{k,pf}$ by (2.6). Exit if any component of $x^{k,pf}$ is infinite.
5. **Subspace phase:** Define $\mathcal{A}^k$ from (2.7), and compute $x^{k,s}$ by solving (2.8) approximately as specified by (2.9).
6. **Projected search on subspace direction:** Compute x^{k+1} from (2.10). Exit if any component of x^{k+1} is infinite.
7. **Update trust-region radius:** Define the new trust-region radius Δ^{k+1} by (2.11).

end

The reduced-space phase given by Steps 5 and 6 may be performed recursively. In this case, the active set $\mathcal{A}^k$ should be redefined each time and be based on the vector resulting from the projected search given by Step 6. For further details, see [6].

FIG. 2.1. *Steps computed in Algorithm* 2.1



2.2. Global convergence. We now proceed to prove that Algorithm 2.1 is convergent under certain assumptions on the splitting $M = B + C$. The careful reader may observe that intermediary results rely on solving LCP subproblems, which merely reinforces the close ties that exist between problems BQP(q,M) and LCP(q,M). We begin by stating [4, Lemma 5.5.1], which establishes a descent property for matrix splitting iterations.

LEMMA 2.1. *If $M = B + C$ is splitting of the symmetric matrix $M \in \mathbb{R}^{n\times n}$ such that $B \succeq 0$, then*

$$\langle d^{k,c}, Mx^k + q\rangle \le -\langle d^{k,c}, Bd^{k,c}\rangle \le 0, \tag{2.12}$$

where $d^{k,c}$ is defined by (2.3)*. Moreover, if B is either symmetric or positive definite, and $\langle d^{k,c}, Mx^k + q\rangle = 0$, then x^k solves* LCP(q,M).

The next lemma gives a bound on the decrease in f achieved by the Cauchy point $x^{k,c}$.

LEMMA 2.2. *If $M = B + C$ is a splitting of the symmetric matrix $M \in \mathbb{R}^{n\times n}$ such that $B \succ 0$, and x^k is not a solution to* LCP(q,M)*, then*

$$f(x^k) - f(x^{k,c}) \ge \tfrac{1}{2}|\langle d^{k,c}, Mx^k + q\rangle| \min\left(1, \frac{|\langle d^{k,c}, Mx^k + q\rangle|}{|\langle d^{k,c}, Md^{k,c}\rangle|}\right). \tag{2.13}$$

Proof. Since $B \succ 0$ and x^k is not a solution to LCP(q,M) by assumption, we conclude from Lemma 2.1 that

$$\langle d^{k,c}, \nabla f(x^k)\rangle = \langle d^{k,c}, Mx^k + q\rangle < 0 \tag{2.14}$$

so that $d^{k,c}$ is a descent direction for f at x^k. Moreover, the definition of f yields

$$f(x^k) - f(x^k + \alpha d^{k,c}) = -\alpha\langle d^{k,c}, Mx^k + q\rangle - \frac{\alpha^2}{2}\langle d^{k,c}, Md^{k,c}\rangle \tag{2.15}$$

for all α. We now consider different cases.
<u>Case 1</u>: $\langle d^{k,c}, Md^{k,c}\rangle \le 0$

It follows from $\langle d^{k,c}, Md^{k,c}\rangle \le 0$, (2.15), (2.14), and $x^k + d^{k,c} = \text{FPI}(x^k, 1, B, C) \ge 0$ that $\alpha^{k,c} \ge 1$ and

$$\begin{aligned} f(x^k) - f(x^{k,c}) &= f(x^k) - f(x^k + \alpha^{k,c} d^{k,c}) \\ &\ge \alpha^{k,c} |\langle d^{k,c}, Mx^k + q\rangle| \ge \tfrac{1}{2} |\langle d^{k,c}, Mx^k + q\rangle|. \end{aligned} \tag{2.16}$$

<u>Case 2</u>: $\langle d^{k,c}, Md^{k,c}\rangle > 0$
We define

$$\alpha_u = \operatorname*{argmin}_{\alpha \ge 0} f(x^k + \alpha d^{k,c}) = -\frac{\langle d^{k,c}, Mx^k + q\rangle}{\langle d^{k,c}, Md^{k,c}\rangle} > 0$$

and consider two subcases.
Subcase 1: $x^k + \alpha_u d^{k,c} \ge 0$
In this case it must follow that $\alpha^{k,c} \equiv \alpha_u$ and

$$\begin{aligned} f(x^k) - f(x^{k,c}) &= f(x^k) - f(x^k + \alpha^{k,c} d^{k,c}) \\ &= -\alpha_u \langle d^{k,c}, Mx^k + q\rangle - \frac{\alpha_u^2}{2} \langle d^{k,c}, Md^{k,c}\rangle \\ &= \frac{\alpha_u}{2} |\langle d^{k,c}, Mx^k + q\rangle|, \end{aligned} \tag{2.17}$$

where we have used (2.15), the definition of α_u, and algebraic simplification.
Subcase 2: $x^k + \alpha_u d^{k,c} \not\ge 0$
We conclude that

$$1 \le \alpha^{k,c} < \alpha_u = -\frac{\langle d^{k,c}, Mx^k + q\rangle}{\langle d^{k,c}, Md^{k,c}\rangle}, \tag{2.18}$$

which follows from the inequality $\langle d^{k,c}, Md^{k,c}\rangle > 0$, the fact that $x^k + \alpha_u d^{k,c}$ is assumed to have at least one negative component, the observation that $x^k + d^{k,c} \ge 0$, and the definition of α_u. It follows that

$$\begin{aligned} f(x^k) - f(x^{k,c}) &= f(x^k) - f(x^k + \alpha^{k,c} d^{k,c}) \\ &= -\alpha^{k,c} \langle d^{k,c}, Mx^k + q\rangle - \frac{(\alpha^{k,c})^2}{2} \langle d^{k,c}, Md^{k,c}\rangle \\ &\ge \frac{\alpha^{k,c}}{2} |\langle d^{k,c}, Mx^k + q\rangle| \ge \frac{1}{2} |\langle d^{k,c}, Mx^k + q\rangle|, \end{aligned} \tag{2.19}$$

where we have used (2.18) and algebraic simplification.

The desired result (2.13) now follows from (2.16), (2.17), and (2.19). ☐

We now give a bound on the guaranteed decrease in f given by the *full* step.

Corollary 2.3. *If $M = B + C$ is a splitting of the symmetric matrix $M \in \mathbb{R}^{n\times n}$ such that $B \succ 0$, and x^k is not a solution to* LCP(q,M), *then*

$$f(x^k) - f(x^{k+1}) \ge \tfrac{1}{2} |\langle d^{k,c}, Mx^k + q\rangle| \min\left(1, \frac{|\langle d^{k,c}, Mx^k + q\rangle|}{|\langle d^{k,c}, Md^{k,c}\rangle|}\right). \tag{2.20}$$

Proof. Observe that

$$f(x^k) > f(x^{k,c}) \ge f(x^{k,pf}) \ge f(x^{k+1}),$$

where we have used the assumption that x^k is not a solution to LCP(q,M), Lemma 2.1, Lemma 2.2, and the formulation of Steps 1, 3, and 5 of Algorithm 2.1 in succession. The desired result now follows from Lemma 2.2. □

The main convergence result for problem BQP(q,M) now follows.

THEOREM 2.4. *Let $M = B + C$ be a splitting of the symmetric matrix $M \in \mathbb{R}^{n\times n}$ such that $B \succ 0$, and assume that $n_{max} = \infty$ in Algorithm* 2.1. *Then either (i) the algorithm terminates finitely with a solution in Step* 1*; (ii) the algorithm terminates because an unbounded ray is discovered in either Step* 2*,* 4*, or* 6*; or (iii) an infinite sequence of iterates is computed such that every limit point of the iterates generated by Algorithm* 2.1 *is a solution to* LCP(q,M)*, i.e., is a first-order solution to* BQP(q,M)*.*

Proof. We assume that x^k is not a first-order solution to problem BQP(q,M) for all $k \ge 0$, since otherwise the algorithm would terminate finitely in Step 1, which corresponds to possibility (i) in the statement of the theorem. Moreover, we assume that Algorithm 2.1 does not terminate finitely by discovering a feasible ray upon which f is unbounded, which would correspond to possibility (ii) in the statement of the theorem. It follows that x^k is not a first-order solution for all k, an infinite number of iterates are computed, and each step of the algorithm is well defined. Let

$$\bar{x} = \lim_{k\in\mathcal{S}_1} x^k \tag{2.21}$$

be a limit point of the iterates generated by Algorithm 2.1 for some subsequence $\mathcal{S}_1$. We observe that f is monotonically decreasing over the sequence $\{x^k\}$ as a result of Corollary 2.3, and we claim that

$$\lim_{k\to\infty} f(x^k) = f(\bar{x}). \tag{2.22}$$

To see this, we first suppose that $f(x^k)$ is unbounded below. It follows that there exists an integer k_1 such that $f(x^k) \le f(\bar{x}) - 1$ for all $k \ge k_1$ since $\{f(x^k)\}$ is monotonically decreasing. This is clearly a contradiction since continuity of f and (2.21) imply that $\lim_{k\in\mathcal{S}_1} f(x^k) = f(\bar{x})$. Thus, $\{f(x^k)\}$ is bounded below, must converge, and clearly its limit is $f(\bar{x})$, which proves (2.22).

Next, we may use (2.22) to deduce from (2.20) that

$$\lim_{k\to\infty} \left\{ |\langle d^{k,c}, Mx^k + q\rangle| \min\left(1, \frac{|\langle d^{k,c}, Mx^k + q\rangle|}{|\langle d^{k,c}, Md^{k,c}\rangle|}\right) \right\} = 0 \tag{2.23}$$

and proceed by considering two cases.

<u>Case 1</u>: $\{|\langle d^{k,c}, Mx^k + q\rangle| / |\langle d^{k,c}, Md^{k,c}\rangle|\}_{k\in\mathcal{S}_1} \ge \varepsilon > 0$.

Under the assumptions of this case, (2.23) implies that $\lim_{k\in\mathcal{S}_1} |\langle d^{k,c}, Mx^k + q\rangle| = 0$, and hence we deduce from Lemma 2.1 and the fact that $B \succ 0$, that $\lim_{k\in\mathcal{S}_1} d^{k,c} = 0$. We may then use the definition of $d^{k,c}$, the triangle inequality, and (2.21) to show

$$\begin{aligned} 0 = \lim_{k\in\mathcal{S}_1} d^{k,c} &= \lim_{k\in\mathcal{S}_1} \Big(\mathrm{FPI}(x^k, 1, B, C) - x^k\Big) \\ &= \lim_{k\in\mathcal{S}_1} \Big(\mathrm{FPI}(x^k, 1, B, C) - \bar{x} + \bar{x} - x^k\Big) = \lim_{k\in\mathcal{S}_1} \mathrm{FPI}(x^k, 1, B, C) - \bar{x}. \end{aligned} \tag{2.24}$$

Now recall that the vector $y^k \triangleq \mathrm{FPI}(x^k, 1, B, C)$ satisfies

$$y^k \ge 0, \quad By^k + Cx^k + q \ge 0, \quad \text{and} \quad y^k \circ (By^k + Cx^k + q) = 0$$

by construction. Taking limits, using (2.21), recalling that $M = B + C$, and using $\lim_{k\in\mathcal{S}_1} y^k = \bar{x}$, which follows from (2.24), we have

$$\bar{x} \ge 0, \quad M\bar{x} + q \ge 0, \quad \text{and} \quad \bar{x} \circ (M\bar{x} + q) = 0$$

so that $\bar{x}$ is a first-order solution to BQP(q,M).
Case 2: there exists $\mathcal{S}_2 \subseteq \mathcal{S}_1$ such that $\lim_{k\in\mathcal{S}_2} |\langle d^{k,c}, Mx^k + q\rangle|/|\langle d^{k,c}, Md^{k,c}\rangle| = 0$.
We first claim that $\{d^{k,c}\}_{k\in\mathcal{S}_2}$ is bounded. To prove this, we suppose it is not true and define

$$b^k = \frac{d^{k,c}}{\|d^{k,c}\|_2}$$

for some subsequence $\mathcal{S}_3 \subseteq \mathcal{S}_2$ such that

$$\bar{b} = \lim_{k\in\mathcal{S}_3} b^k, \quad \|\bar{b}\|_2 = 1, \quad \text{and} \quad \lim_{k\in\mathcal{S}_3} \|d^{k,c}\|_2 = \infty. \tag{2.25}$$

Using the definition of b^k and Lemma 2.1, we conclude that

$$\frac{\langle b^k, Mx^k + q\rangle}{\|d^{k,c}\|_2} = \frac{\langle d^{k,c}, Mx^k + q\rangle}{\|d^{k,c}\|_2^2} \leq \frac{-\langle d^{k,c}, Bd^{k,c}\rangle}{\|d^{k,c}\|_2^2} = -\langle b^k, Bb^k\rangle. \tag{2.26}$$

We reach a contradiction by taking limits of (2.26) for $k \in \mathcal{S}_3$, since (2.25), (2.21), and the fact that B is a positive-definite matrix implies that the left-hand-side converges to zero and the right-hand-side is strictly negative. Thus, we conclude that $\{d^{k,c}\}_{k\in\mathcal{S}_2}$ must be bounded.

Next, the assumption of this case ensures that

$$\lim_{k\in\mathcal{S}_2} \langle d^{k,c}, Mx^k + q\rangle = 0,$$

while the boundedness of $\{d^{k,c}\}_{k\in\mathcal{S}_2}$ ensures the existence of $\mathcal{S}_4 \subseteq \mathcal{S}_2$ such that $\lim_{k\in\mathcal{S}_4} d^{k,c} = \bar{d}$. Combining these two facts with (2.21) yields

$$\lim_{k\in\mathcal{S}_4} \langle \bar{d}, M\bar{x} + q\rangle = 0. \tag{2.27}$$

Taking limits of (2.12) for $k \in \mathcal{S}_4$, and using (2.27) and the fact that $B \succ 0$, we again conclude that $\lim_{k\in\mathcal{S}_4} d^{k,c} = 0$. The argument in Case 1 may now be repeated to show that $\bar{x}$ is a first-order solution to BQP(q,M), which completes the proof. ☐

A couple of comments are in order.

- The theory just described carries over to the case that iteration dependent splittings $M = B^k + C^k$ are used provided the matrices $\{B^k\}_{k\geq 0}$ are uniformly positive definite.
- Limit points of the sequence $\{x^k\}_{k\geq 0}$ are guaranteed under the assumption that the level curves of f are bounded on the orthant $x \geq 0$.

2.3. Numerical tests. In this section we validate the effectiveness of our Matlab [10] implementation of Algorithm 2.1 by solving strictly convex problems in Section 2.3.1, convex problems in Section 2.3.2, and nonconvex problems in Section 2.3.3. For comparison, we have also written our own Matlab implementation of a two-phase projected gradient based method. The step length along the gradient direction at each step is chosen by the Borzalai-Borwein formula [5, 7] and the subspace step is computed as described below.

For simplicity and computational efficiency, we simplified the subspace step computation required by Step 5 of Algorithm 2.1; we set $x^{k,s}_{\mathcal{A}^k} = 0$ and solve for the remaining components by using the "backslash" operator in Matlab to solve the system

$$M_{\mathcal{I}} x^{k,s}_{\mathcal{I}} = -c_{\mathcal{I}}, \tag{2.28}$$

where $\mathcal{I}$ is the complement of the set $\mathcal{A}^k$ defined by (2.7). Note that if the matrix $M_{\mathcal{I}}$ is singular, Matlab may compute a least-length least-squares solution, and thus this scheme is always well defined.

Moreover, we no longer require a trust-region radius as given by (2.8) or parameters η_c and η_e, since we are implicitly assuming that $\Delta^k = \infty$. (This does not alter the convergence theory in any way since the trust-region radius is merely a means for ensuring that the subspace problem (2.8) is well defined.)

To allow for a fair comparison between the two algorithms we (i) allow a maximum of three recursive subspace steps to be computed per iteration for both algorithms as described just after the statement of Algorithm 2.1; (ii) choose $n_f = 0$ in Algorithm 2.1; and (iii) terminate execution in both algorithms when either a maximum iteration of $n_{\max} = 500$ was reached or when

$$\| \min(x^k, Mx^k + q)\|_\infty \leq 1.0e^{-6}, \tag{2.29}$$

which indicates that an approximation first-order solution to BQP(q,M) has been found.

2.3.1. Strictly convex problems. We first solved the strictly convex BQPs from the CUTEr test set; the results are recorded in Table 2.1. We note that since M is positive definite for this class of problems, we use projected successive overrelaxation as indicated in Table 1.1 for the matrix splitting of M. The columns have the following meaning: "Prob" represents the name of the problem solved, "n" the number of optimization variables, "stat" the outcome flag with zero always representing a successful solve, "res" the final residual as measured by the left side of (2.29), "iter" the number of iterations performed, "nss" the number of subspace steps computed, and "nsplit" the number of matrix splitting iterations performed. The initial point for all problems is the default value supplied by CUTEr.

From Table 2.1 we can see that both algorithms perform quite well on this class of problems. However, one may also observe that Algorithm 2.1 has a tendency to require fewer iterations and subspace steps. This is evidence that the more sophisticated projected successive overrelaxation fixed-point iteration is generally superior to a projected gradient iteration in terms of identifying an optimal active set—at least for strictly convex BQPs.

TABLE 2.1
Output from the projected gradient algorithm and Algorithm 2.1 on the strictly convex BQPs from the CUTEr test set.

		Projected Gradient				Algorithm 2.1				
Prob	n	stat	res	iter	nss	stat	res	iter	nss	nsplit
BIGGSB1	5000	1	$3.61e^{-03}$	500	500	0	$2.22e^{-16}$	215	216	215
BQP1VAR	1	0	$0.00e^{+00}$	1	1	0	$0.00e^{+00}$	1	1	1
BQPGABIM	50	0	$2.08e^{-17}$	2	3	0	$2.08e^{-17}$	1	1	1
BQPGASIM	50	0	$2.02e^{-17}$	2	3	0	$2.02e^{-17}$	1	1	1
OSLBQP	8	0	$0.00e^{+00}$	1	1	0	$0.00e^{+00}$	1	1	1
PENTDI	5000	0	$0.00e^{+00}$	1	1	0	$0.00e^{+00}$	1	1	1
SIM2BQP	2	0	$0.00e^{+00}$	1	1	0	$0.00e^{+00}$	1	1	1
SIMBQP	2	0	$0.00e^{+00}$	1	2	0	$0.00e^{+00}$	1	2	1

Next, we tested Algorithm 2.1 on a set of randomly generated strictly convex BQPs; the results are documented in Table 2.2. The random problems were formed by first computing M and q by sampling from the standard normal distribution. We then set $M \leftarrow \tau_s M^T M + \varepsilon$ and $q \leftarrow \tau_s q$, where the scale factor $\tau_s = 100$ and constant $\varepsilon = 1.0e^{-5}$ were used. In all cases an initial point of zero was used.

Table 2.2 tells a similar story; namely, both algorithms perform quite well, but the more sophisticated projected successive overrelaxation fixed-point iteration generally leads to fewer iterations and subspace steps as a result of its superior active set identification.

2.3.2. Convex problems. Next we solved the convex BQPs—eliminating the strictly convex problems—from the CUTEr test set; the results are recorded in Table 2.3. We again used the matrix splitting corresponding to successive overrelaxation and an initial point supplied by CUTEr.

Table 2.2

Output from the projected gradient algorithm and Algorithm 2.1 on ten randomly generated strictly convex BQPs.

	Projected Gradient				Algorithm 2.1				
n	stat	res	iter	nss	stat	res	iter	nss	nsplit
4000	0	$1.46e^{-08}$	14	38	0	$1.46e^{-08}$	4	10	4
4000	0	$1.64e^{-08}$	7	17	0	$1.64e^{-08}$	4	10	4
4000	0	$1.32e^{-08}$	6	15	0	$1.32e^{-08}$	4	9	4
4000	0	$1.44e^{-08}$	7	15	0	$1.44e^{-08}$	4	10	4
4000	0	$1.51e^{-08}$	7	17	0	$1.51e^{-08}$	4	10	4
4000	0	$1.42e^{-08}$	7	17	0	$1.42e^{-08}$	4	10	4
4000	0	$1.47e^{-08}$	7	17	0	$1.47e^{-08}$	5	11	5
4000	0	$1.66e^{-08}$	6	16	0	$1.66e^{-08}$	3	8	3
4000	0	$1.22e^{-08}$	31	85	0	$1.22e^{-08}$	4	10	4
4000	0	$1.39e^{-08}$	16	44	0	$1.39e^{-08}$	4	9	4

Table 2.3 indicates that the trend continues; namely, both algorithms perform quite well, but the more sophisticated projected successive overrelaxation fixed-point iteration generally leads to fewer iterations and subspace iterations as a result of its superior active set identification.

2.3.3. Nonconvex problems. Finally, we solved the nonconvex BQPs from the CUTEr test set and give the results in Table 2.4. We note that since M is now indefinite, we use the splitting $B = I$ and $C = M - I$ as indicated in Table 1.1. It would likely be advantageous to use a more sophisticated matrix splitting, but we leave this study to future research. The initial point for all problems is the default value supplied by CUTEr.

Table 2.4 indicates that Algorithm 2.1 is capable of handling nonconvex problems; this is an attribute not likely shared by the algorithm presented in [9].

3. Linear complementarity problems. We now consider an algorithm for solving problem LCP(q,M) on page 1 for a given square (not neccessarily symmetric) matrix $M \in \mathbb{R}^{n\times n}$ and vector $q \in \mathbb{R}^n$. Important to our discussion is the merit function

$$\phi(x) = \|\min(x, Mx + q)\|_2. \tag{3.1}$$

Note that $\phi(x) = 0$ if and only if x is a solution to LCP(q,M). One motivation for using (3.1) is that the linear structure of problem LCP(q,M) is maintained, which is in contrast to most commonly used nonlinear complementarity functions [3, 15, 2, 13].

Throughout this section we make the following assumption on the matrix splitting.

Assumption 3.1. *The matrix splitting $M = B + C$ is chosen such that the fixed-point iterations are contractions, i.e., there exists a constant $\rho_f \in (0, 1)$ such that for any x*

$$\|\text{FPI}(x, 2, B, C) - \text{FPI}(x, 1, B, C)\|_2 \le \rho_f \|\text{FPI}(x, 1, B, C) - x\|_2. \tag{3.2}$$

Note that Assumption 3.1 is satisfied, for example, when the matrix M is strictly diagonally dominant or symmetric and positive definite. See [4] for more contraction results related to matrix splittings. The algorithm proposed in Section 3.1 depends crucially on the availability of a matrix splitting for which condition (3.2) is satisfied.

3.1. The algorithm. In this section we describe each step of our method, which is given as Algorithm 3.1 and depicted in Figure 3.1. Let the current iterate be $x^k \ge 0$. In Step 1, we choose

TABLE 2.3
Output from the projected gradient algorithm and Algorithm 2.1 on the convex BQPs from the CUTEr test set.

		Projected Gradient				Algorithm 2.1				
Prob	n	stat	res	iter	nss	stat	res	iter	nss	nsplit
CHENHARK	5000	1	$2.40e^{-05}$	500	501	1	$2.38e^{-05}$	500	501	500
CVXBQP1	10000	0	$9.44e^{-16}$	2	7	0	$0.00e^{+00}$	1	1	1
HS3	2	0	$0.00e^{+00}$	2	3	0	$0.00e^{+00}$	1	1	1
HS3MOD	2	0	$0.00e^{+00}$	5	9	0	$0.00e^{+00}$	2	3	2
JNLBRNG1	10000	0	$4.74e^{-16}$	19	19	0	$4.74e^{-16}$	7	7	7
JNLBRNG2	10000	0	$2.24e^{-15}$	11	11	0	$2.23e^{-15}$	5	5	5
JNLBRNGA	10000	0	$4.43e^{-16}$	18	18	0	$4.43e^{-16}$	8	8	8
JNLBRNGB	10000	0	$3.57e^{-15}$	7	7	0	$3.56e^{-15}$	4	4	4
NOBNDTOR	5476	0	$1.07e^{-15}$	23	23	0	$1.07e^{-15}$	17	17	17
OBSTCLAE	10000	0	$1.60e^{-15}$	21	22	0	$1.60e^{-15}$	21	22	21
OBSTCLAL	10000	0	$1.60e^{-15}$	20	20	0	$1.60e^{-15}$	20	20	20
OBSTCLBL	10000	0	$1.27e^{-15}$	15	29	0	$1.26e^{-15}$	14	28	14
OBSTCLBM	10000	0	$1.27e^{-15}$	9	15	0	$1.26e^{-15}$	12	20	12
OBSTCLBU	10000	0	$1.27e^{-15}$	16	19	0	$1.26e^{-15}$	15	19	15
TORSION1	5476	0	$1.40e^{-15}$	23	23	0	$1.40e^{-15}$	19	19	19
TORSION2	5476	0	$1.40e^{-15}$	12	13	0	$1.40e^{-15}$	16	17	16
TORSION3	5476	0	$9.35e^{-16}$	11	11	0	$9.35e^{-16}$	9	9	9
TORSION4	5476	0	$9.35e^{-16}$	12	13	0	$9.35e^{-16}$	10	11	10
TORSION5	5476	0	$7.40e^{-16}$	6	6	0	$7.39e^{-16}$	6	6	6
TORSION6	5476	0	$7.40e^{-16}$	7	8	0	$7.39e^{-16}$	7	8	7
TORSIONA	5476	0	$9.84e^{-16}$	23	23	0	$9.83e^{-16}$	18	18	18
TORSIONB	5476	0	$9.84e^{-16}$	12	13	0	$9.83e^{-16}$	13	14	13
TORSIONC	5476	0	$1.23e^{-15}$	11	11	0	$1.22e^{-15}$	9	9	9
TORSIOND	5476	0	$1.23e^{-15}$	12	13	0	$1.22e^{-15}$	10	11	10
TORSIONE	5476	0	$6.29e^{-16}$	6	6	0	$6.28e^{-16}$	6	6	6
TORSIONF	5476	0	$6.29e^{-16}$	7	8	0	$6.28e^{-16}$	7	8	7

TABLE 2.4
Output from Algorithm 2.1 on the nonconvex BQPs from the CUTEr test set.

		Algorithm 2.1				
Prob	n	stat	res	iter	nss	nsplit
NCVXBQP1	10000	0	$0.00e^{+00}$	1	1	1
NCVXBQP2	10000	0	$2.77e^{-17}$	19	40	19
NCVXBQP3	10000	0	$1.45e^{-11}$	15	34	15
QUDLIN	5000	0	$0.00e^{+00}$	1	1	1

an integer $n_f \geq 2$ and compute $n_f + 1$ fixed-point iterations $\{x^{k,f,j}\}_{j=0}^{n_f+1}$ starting from x^k, i.e.,

$$x^{k,f,0} = x^k \quad \text{and} \quad x^{k,f,j+1} = \text{FPI}(x^{k,f,j}, 1, B, C) \quad \text{for } 0 \leq j \leq n_f. \tag{3.3}$$

We note that the choice $n_f \geq 2$ allows us to use a contraction argument to guarantee convergence of the method under certain assumptions. We denote the *actual* contraction resulting from the ith fixed-point iteration as

$$c^{k,f,i} \triangleq \frac{\|x^{k,f,i} - x^{k,f,i-1}\|_2}{\|x^{k,f,i-1} - x^{k,f,i-2}\|_2} \le \rho_f \quad \text{for all } k \ge 0 \text{ and } 2 \le i \le n_f + 1, \tag{3.4}$$

where the inequality follows from Assumption 3.1.

In Step 2 we predict the variables that are zero at a solution to LCP(q,M) by using the inactive and active index sets

$$\mathcal{I} = \{i : x^{k,f,n_f} > 0\} \ \text{ and } \ \mathcal{A} = \{i : x^{k,f,n_f} = 0\}, \tag{3.5}$$

and then compute a subspace step $x^{k,s}$ such that

$$x^{k,s}_{\mathcal{A}} = 0 \ \text{ and } \ x^{k,s}_{\mathcal{I}} = \max(x_{\mathcal{I}}, 0), \tag{3.6}$$

where $x_{\mathcal{I}}$ is an approximate solution of

$$\underset{x_{\mathcal{I}} \in \mathbb{R}^n}{\text{minimize}} \ \tfrac{1}{2}\|M_{\mathcal{I}\mathcal{I}} x_{\mathcal{I}} - q_{\mathcal{I}}\|_2 \quad \text{subject to } \|x_{\mathcal{I}} - x^{k,f,n_f}_{\mathcal{I}}\|_2 \le \Delta^k \tag{3.7}$$

for a given trust-region radius $\Delta^k > 0$. Generally, we expect the subspace step to improve the global and local converge of the iterates. To prove global convergence, however, we may accept any $x_{\mathcal{I}}$ satisfying

$$\|x_{\mathcal{I}} - x^{k,f,n_f}_{\mathcal{I}}\|_2 \le \Delta^k \tag{3.8}$$

as an approximate solution. Once the subspace step has been computed, we enter Step 3 and compute $n_s \ge 2$ (again, to allow for a contraction argument) additional fixed-point iterations $\{x^{k,s,j}\}_{j=0}^{n_s}$ starting from $x^{k,s}$, i.e.,

$$x^{k,s,0} = x^{k,s} \ \text{ and } \ x^{k,s,j+1} = \text{FPI}(x^{k,s,j}, 1, B, C) \ \text{ for } 0 \le j \le n_s - 1. \tag{3.9}$$

Although the iterates $\{x^{k,s,j}\}_{j=0}^{n_s}$ are the result of fixed-point iterations, we employ the superscript "s" to stress that these steps are still part of the subspace *phase*. Thus the subspace *phase* consists of the subspace *step* followed by n_s fixed-point iterations. Similar to before, we define the *actual* contraction resulting from the ith fixed-point iteration during the subspace phase as

$$c^{k,s,i} \triangleq \frac{\|x^{k,s,i} - x^{k,s,i-1}\|_2}{\|x^{k,s,i-1} - x^{k,s,i-2}\|_2} \le \rho_f \quad \text{for all } k \ge 0 \text{ and } 2 \le i \le n_s, \tag{3.10}$$

where the inequality follows from Assumption 3.1.

Finally, Step 4 is dedicated to determining step acceptance. We begin by choosing

$$\rho^k = \max\left(\rho_u, (1 + c^k_{\max})/2\right) \ \text{ for some constant } 1/2 < \rho_u < 1, \tag{3.11}$$

where

$$c^k_{\max} \triangleq \max\left(\max_{2 \le i \le n_f+1} \{c^{k,f,i}\}, \max_{2 \le i \le n_s} \{c^{k,s,i}\} \right) \le \rho_f \tag{3.12}$$

Thus $\rho^k < 1$ since $\rho_f < 1$, and ρ^k is a measure of the minimum contraction factor obtained during the previous fixed-point iterations. Next, we check if the following two contractions hold:

$$\|x^{k,s,1} - x^{k,f,n_f}\|_2 \le \rho^k \|x^{k,f,n_f} - x^{k,f,n_f-1}\|_2 \ \text{ and } \tag{3.13}$$

$$\|x^{k,s,2} - x^{k,s,1}\|_2 \le \rho^k \|x^{k,s,1} - x^{k,f,n_f}\|_2. \tag{3.14}$$

These conditions measure the effect—in terms of contraction—of the subspace *step* during the subspace *phase*. If both of these conditions are satisfied then we accept $x^{k+1} = x^{k,s,n_s}$, possibly increase Δ^k, and proceed to the next iteration. If either (3.13) or (3.14) fails, then we check whether

$$\phi(x^{k,s,n_s}) \le \tfrac{1}{2}\phi^k_{\max} \tag{3.15}$$

is satisfied, where $\phi^k_{\max}$ plays the role of a forcing sequence. The motivation for this condition is to accept "large" subspace steps that make substantial progress towards a solution of LCP(q,M) that would otherwise be rejected by conditions (3.13) and (3.14). If (3.15) is satisfied, we define $x^{k+1} = x^{k,s,n_s}$, decrease the value of $\phi^k_{\max}$ by setting $\phi^{k+1}_{\max} = \frac{1}{2}\phi^k_{\max}$, possibly increase the trust-region radius, and proceed to the next iteration. If (3.13) or (3.14) fails, and (3.15) does not hold, then we have no recourse but to decrease the trust-region radius Δ^k in (3.7) and compute a new—and smaller—subspace step. Based on this description, it will be convenient to define

$$\mathcal{C} = \{k : (3.13) \text{ and } (3.14) \text{ are satisfied at the } k\text{th iteration}\}, \tag{3.16}$$

$$\mathcal{M} = \{k : k \notin \mathcal{C} \text{ and } (3.15) \text{ is satisfied at the } k\text{th iteration}\}, \text{ and} \tag{3.17}$$

$$\mathcal{F} = \{k : k \notin (\mathcal{C} \cup \mathcal{M})\}. \tag{3.18}$$

We call iterates $k \in \mathcal{C}$ contraction iterates or $\mathcal{C}$-iterates, iterates $k \in \mathcal{M}$ merit function iterates or $\mathcal{M}$-iterates, and iterates $k \in \mathcal{F}$ failed iterates or $\mathcal{F}$-iterates.

Algorithm 3.1. Algorithm for solving LCP(q,M).
Input: $x^0 \ge 0$, $n_{\max} \ge 0$, $0 < \eta_c < 1 < \eta_e$, $0 < \Delta_R < \Delta_{\max}$, and $0 < \rho_u < 1$.
Choose splitting $M = B + C$ with $B \succ 0$, and set $\phi^0_{\max} = \max\big(\phi(x^0), 1.0e^5\big)$.
for $k = 0, 1, \ldots, n_{\max}$

1. **Fixed-point iterations:** Choose $n_f \ge 2$ and compute $\{x^{k,f,j}\}_{j=0}^{n_f+1}$ from (3.3).
2. **Subspace step:** Define $\mathcal{A}$ and $\mathcal{I}$ by (3.5), and compute $x^{k,s}$ to satisfy (3.6) where $x_{\mathcal{I}}$ is any approximate solution to (3.7) satisfying condition (3.8).
3. **Additional fixed-point iterations:** Choose $n_s \ge 2$ and compute $\{x^{k,s,j}\}_{j=0}^{n_s}$ from (3.9).
4. **Step acceptance:** Define ρ^k by (3.11) and
   if (3.13) and (3.14) are satisfied **then** [$\mathcal{C}$-iterate]
   set $x^{k+1} = x^{k,s,n_s}$, $\phi^{k+1}_{\max} = \phi^k_{\max}$, and $\Delta^{k+1} = \text{med}(\Delta_R, \eta_e\Delta^k, \Delta_{\max})$;
   else if (3.15) is satisfied **then** [$\mathcal{M}$-iterate]
   set $x^{k+1} = x^{k,s,n_s}$, $\phi^{k+1}_{\max} = \frac{1}{2}\phi^k_{\max}$, and $\Delta^{k+1} = \text{med}(\Delta_R, \eta_e\Delta^k, \Delta_{\max})$;
   else [$\mathcal{F}$-iterate]
   set $x^{k+1} = x^k$, $\phi^{k+1}_{\max} = \phi^k_{\max}$, and $\Delta^k = \eta_c\Delta^k$.
   end if
5. **Check for optimality:** If $\phi(x^{k+1}) = 0$, exit with solution x^{k+1}.

end

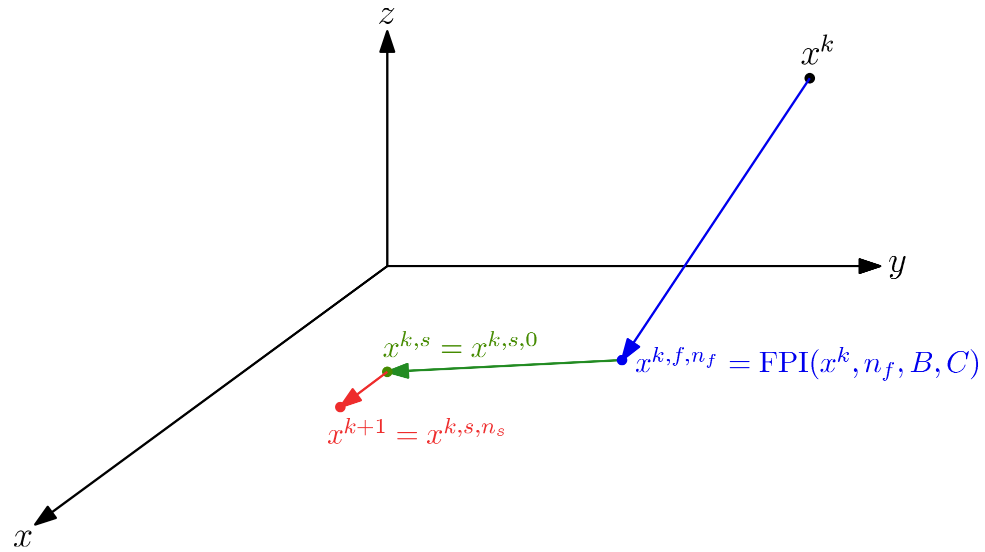
The reduced-space phase given by Step 2 may be performed recursively as discussed in [6]. In this case, the active/inactive sets $\mathcal{A}/\mathcal{I}$ defined by (3.5) should be redefined each time and be based on the vector $x^{k,s}$ resulting from Step 2. Also note that the trust-region problem (3.7) may be solved inexactly since the only requirement of the approximate solution is that it satisfies (3.8).

3.2. Global convergence. We begin our convergence analysis of Algorithm 3.1 by proving estimates for the contraction measure ρ^k.

LEMMA 3.1. *Let ρ^k be defined by* (3.11) *and define*

$$\rho = \max\big(\rho_u, (1 + \rho_f)/2\big), \tag{3.19}$$

FIG. 3.1. *Steps computed in Algorithm* 3.1



where ρ_f *is defined in Assumption* 3.1*. The following then hold for all* $k \geq 0$*:*

(a) $1 > \rho \geq \rho^k \geq \rho_u > 1/2$ *for all* $k \geq 0$*;*

(b) $\rho^k - c^k_{max} \geq \frac{1}{2}(1-\rho_f)$ *for all* $k \geq 0$*; and*

(c) $1/c^k_{max} - 1/\rho^k \geq (1-\rho_f)/(2\rho\rho_f)$*.*

Proof. It follows from the choice of ρ_u, the max function, (3.11), (3.12), and (3.19) that

$$1/2 < \rho_u \leq \max\left(\rho_u, (1+c^k_{\max})/2\right) = \rho^k \leq \max\left(\rho_u, (1+\rho_f)/2\right) = \rho < 1$$

for all $k \geq 0$. This proves part (a).

We now prove part (b). First, observe that the definition of ρ^k guarantees that $\rho^k \geq \left(1+c^k_{\max}\right)/2$ for all $k \geq 0$. Next subtract $c^k_{\max}$ from both sides of this inequality, and then use (3.12) to deduce

$$\rho^k - c^k_{\max} \geq \left(1+c^k_{\max}\right)/2 - c^k_{\max} \geq \tfrac{1}{2}(1-c^k_{\max}) \geq \tfrac{1}{2}(1-\rho_f),$$

which proves part (b).

To prove part (c), we use parts (b) and (a), (3.12), (3.4), and (3.10) to obtain

$$\frac{1}{c^k_{\max}} - \frac{1}{\rho^k} = \frac{\rho^k - c^k_{\max}}{\rho^k c^k_{\max}} \geq \frac{1-\rho_f}{2\rho^k c^k_{\max}} \geq \frac{1-\rho_f}{2\rho\rho_f},$$

which is the desired result. ◻

Our next aim is to show that if $n_{\max} = \infty$ in Algorithm 3.1, then the algorithm either terminates finitely with a solution, or generates infinitely many iterates belonging to the set $\mathcal{C} \cup \mathcal{M}$ of successful iterations. The following result proves that conditions (3.13) and (3.14) will both be satisfied when the trust-region radius Δ^k in problem (3.7) is sufficiently small.

LEMMA 3.2. *Let* $M = B + C$ *be a splitting of the matrix* $M \in \mathbb{R}^{n\times n}$ *such that* $B \succ 0$*. It follows that there exists a constant* $\kappa > 0$ *such that if the trust-region radius in problem* (3.7) *satisfies*

$$\Delta^k \leq \frac{(1-\rho_f)}{2} \min\left(\frac{1}{\rho\rho_f}\|x^{k,s,2} - x^{k,s,1}\|_2, \frac{1}{\kappa\|C\|_2}\|x^{k,f,n_f} - x^{k,f,n_f-1}\|_2\right), \tag{3.20}$$

then conditions (3.13) *and* (3.14) *are satisfied, where* ρ *is defined in* (3.19).

Proof. To simplify notation, we define

$$p_1 = x^{k,s,1} - x^{k,s}, \qquad p_2 = x^{k,s,2} - x^{k,s,1}, \tag{3.21}$$
$$\hat{p}_1 = x^{k,f,n_f} - x^{k,f,n_f-1}, \qquad \hat{p}_2 = x^{k,f,n_f+1} - x^{k,f,n_f}, \tag{3.22}$$
$$p_s = x^{k,s} - x^{k,f,n_f}, \qquad p_d = x^{k,s,1} - x^{k,f,n_f+1}. \tag{3.23}$$

(See Figure 3.2 for a cartoon depicting these steps, and recall that $x^{k,s} \equiv x^{k,s,0}$.) It then follows from (3.10) and (3.12) that

$$\|p_2\|_2 = c^{k,s,2}\|p_1\|_2 \le c^k_{\max}\|p_1\|_2. \tag{3.24}$$

Also, it follows from (3.5), (3.6), and the trust-region constraint in (3.7) that

$$\begin{aligned}\|p_s\|_2 &= \|x^{k,s} - x^{k,f,n_f}\|_2 = \|x^{k,s}_{\mathcal{I}} - x^{k,f,n_f}_{\mathcal{I}}\|_2 \\ &= \|\max(x_{\mathcal{I}}, 0) - x^{k,f,n_f}_{\mathcal{I}}\|_2 \le \|x_{\mathcal{I}} - x^{k,f,n_f}_{\mathcal{I}}\|_2 \le \Delta^k,\end{aligned} \tag{3.25}$$

where $x_{\mathcal{I}}$ is the solution to (3.7). We may further bound $\|p_s\|_2$ by using (3.25), (3.20), and part (c) of Lemma 3.1 to get

$$\|p_s\|_2 \le \Delta^k \le \frac{(1-\rho_f)}{2\rho\rho_f}\|p_2\|_2 \le \left(\frac{1}{c^k_{\max}} - \frac{1}{\rho^k}\right)\|p_2\|_2. \tag{3.26}$$

Multiplying (3.26) by ρ^k and then using (3.24) and the reverse triangle-inequality yields

$$\|p_2\|_2 \le \frac{\rho^k}{c^k_{\max}}\|p_2\|_2 - \rho^k\|p_s\|_2 \le \rho^k\|p_1\|_2 - \rho^k\|p_s\|_2 \le \rho^k\|p_1 + p_s\|_2.$$

This is precisely condition (3.14) since $p_s + p_1 = x^{k,s,1} - x^{k,f,n_f}$.

We now proceed to show that condition (3.13) is satisfied. First, it follows from (3.4) and (3.12) that

$$\|\hat{p}_2\|_2 = c^{k,f,n_f+1}\|\hat{p}_1\|_2 \le c^k_{\max}\|\hat{p}_1\|_2. \tag{3.27}$$

Second, if we define

$$q(x) = q + Cx, \tag{3.28}$$

then by construction we have that x^{k,f,n_f+1} is the *unique* solution to LCP($q(x^{k,f,n_f})$,B) and $x^{k,s,1}$ is the *unique* solution to LCP($q(x^{k,s})$,B). It then follows from the definition of p_d, [4, Theorem 7.2.1], (3.28), basic norm inequalities, and the definition of p_s, that there exists a number $\kappa > 0$ such that

$$\|p_d\|_2 = \|x^{k,s,1} - x^{k,f,n_f+1}\|_2 \le \kappa\|q(x^{k,f,n_f}) - q(x^{k,s})\|_2 = \kappa\|C\|_2\|p_s\|_2. \tag{3.29}$$

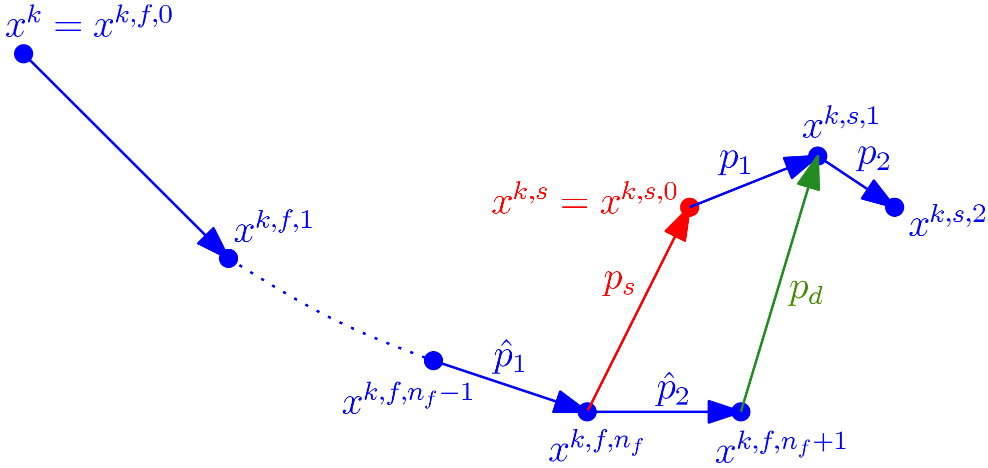
Next, using (3.25), (3.20), and part (b) of Lemma 3.1, we deduce that

$$\|p_s\|_2 \le \Delta^k \le \frac{(1-\rho_f)}{2\kappa\|C\|_2}\|\hat{p}_1\|_2 \le \frac{(\rho^k - c^k_{\max})}{\kappa\|C\|_2}\|\hat{p}_1\|_2. \tag{3.30}$$

If we multiply both sides of this inequality by $\kappa\|C\|_2$ and use (3.29) and (3.27), we find that

$$\rho^k\|\hat{p}_1\|_2 \ge \kappa\|C\|_2\|p_s\|_2 + c^k_{\max}\|\hat{p}_1\|_2 \ge \|p_d\|_2 + \|\hat{p}_2\|_2 \ge \|\hat{p}_2 + p_d\|_2.$$

FIG. 3.2. *Depiction of the steps p_1, p_2, $\hat{p}_1$, $\hat{p}_2$, p_s and p_d used in the proof of Lemma 3.2 and Theorem 3.4.*



Thus, condition (3.13) is satisfied since $\hat{p}_2 + p_d \equiv x^{k,s,1} - x^{k,f,n_f}$, which completes the proof. $\square$

We must interpret (3.20) correctly. In particular, note that the right-hand-side depends on Δ^k since $x^{k,s,1}$ and $x^{k,s,2}$ both depend on $x^{k,s,0} = x^{k,s}$, which in turn depends on Δ^k. Consequently, this lemma does *not* allow us to immediately deduce that both (3.13) and (3.14) will be satisfied for Δ^k sufficiently small since as we decrease Δ^k the right-hand side of (3.20) also changes.

We now use the previous lemma to prove that Algorithm 3.1 generates infinitely many iterates in the set $\mathcal{C} \cup \mathcal{M}$.

LEMMA 3.3. *Let $M = B + C$ be a splitting of the matrix $M \in \mathbb{R}^{n\times n}$ such that $B \succ 0$. Assume that Algorithm 3.1 does not terminate finitely, $n_{max} = \infty$, and $\{x^k\}_{k\geq 0}$ is the sequence of iterates. It follows that $|\mathcal{C} \cup \mathcal{M}| = \infty$.*

Proof. For a proof by contradiction, assume that there exists an integer $\hat{k}$ such that $k \in \mathcal{F}$ for all $k \geq \hat{k}$. It then follows from construction of the algorithm that

$$\lim_{k\to\infty} \Delta_k = 0, \tag{3.31}$$

and that

$$x^k = \widehat{x}, \;\; \phi^k_{\max} = \hat{\phi}_{\max}, \;\; \text{and} \;\; x^{k,f,i} = x^{\hat{k},f,i} \;\; \text{for all } k \geq \hat{k} \text{ and } i = 0 : n_f + 1. \tag{3.32}$$

We also note that no element of $\{x^{\hat{k},f,i}\}_{i=0}^{n_f}$ is a solution of LCP(q,M) for the following reason. If any element was a solution, then it would follow that $\|p_s(\hat{k})\|_2 = \|p_1(\hat{k})\|_2 = \|p_2(\hat{k})\|_2 = 0$, where we have made the dependence of p_s, p_1, and p_2 on the iteration number explicit (see Figure 3.2), which in turn implies that conditions (3.13) and (3.14) are satisfied. This is a contradiction since this implies that $\hat{k} \in \mathcal{C}$. In particular, we know that $x^{\hat{k},f,n_f-1}$ and $x^{\hat{k},f,n_f}$ are both not solutions and thus

$$\|\hat{p}_1(k)\|_2 = \|\hat{p}_1(\hat{k})\|_2 \neq 0 \;\; \text{and} \;\; \|\hat{p}_2(k)\|_2 = \|\hat{p}_2(\hat{k})\|_2 \neq 0 \;\; \text{(see Figure 3.2)} \tag{3.33}$$

for all $k \geq \hat{k}$. Moreover, Lemma 3.2, (3.31), (3.33), (3.32), and the fact that $k \in \mathcal{F}$ for all k sufficiently large, implies that

$$0 = \lim_{k\to\infty} \Delta^k \geq \frac{(1-\rho_f)}{2\rho\rho_f} \lim_{k\to\infty} \|p_2(k)\|_2 \implies \lim_{k\to\infty} p_2(k) = 0. \tag{3.34}$$

We may also conclude from (3.31), (3.7), (3.6), and (3.5) that

$$\lim_{k\to\infty}\left(x^{k,s}-x^{\hat{k},f,n_f}\right)=\lim_{k\to\infty}\left(x^{k,s}-x^{k,f,n_f}\right)=\lim_{k\to\infty}p_s(k)=0. \tag{3.35}$$

Combining (3.35) with an argument similar to that used to derive (3.29), we have

$$\lim_{k\to\infty}\|p_d(k)\|_2\le\kappa\|C\|_2\lim_{k\to\infty}\|p_s(k)\|_2=0 \tag{3.36}$$

for some constant $\kappa>0$.

We now show that for k sufficiently large condition (3.13) is satisfied. We have from (3.4) and (3.11) that

$$\|\hat{p}_2\|_2=c^{k,f,n_f+1}\|\hat{p}_1\|_2\le c^k_{\max}\|\hat{p}_1\|_2. \tag{3.37}$$

Combining this with part (b) of Lemma 3.1 and (3.36), we have

$$\|x^{k,s,1}-x^{k,f,n_f}\|_2\le\rho^k\|\hat{p}_1\|_2\ \text{ for } k\ge\bar{k} \text{ sufficiently large,}$$

which shows that condition (3.13) holds.

Finally, we reach a contradiction by showing that condition (3.14) is also satisfied for $k\ge\bar{k}$ sufficiently large. To this end, we use (3.36) and Figure 3.2 to deduce that

$$\lim_{k\to\infty}\|(x^{k,s,1}-x^{k,f,n_f})-\hat{p}_2\|_2=\lim_{k\to\infty}\|x^{k,s,1}-x^{k,f,n_f+1}\|_2=\lim_{k\to\infty}\|p_d(k)\|_2=0.$$

We may then combine this result with (3.33), (3.34), and part (a) of Lemma 3.1 to obtain

$$\|p_2(k)\|_2\le\frac{1}{4}\|\hat{p}_2\|_2\le\frac{\rho^k}{2}\|\hat{p}_2\|_2\le\rho^k\|x^{k,s,1}-x^{k,f,n_f}\|_2\ \text{ for } k\ge\bar{k} \text{ sufficiently large,}$$

which shows that condition (3.14) holds.

We have reached a contradiction since we have shown that both conditions (3.13) and (3.14) are satisfied for $k\ge\hat{k}$ sufficiently large and thus k will ultimately be an element of $\mathcal{C}$. We conclude that the set $\mathcal{C}\cup\mathcal{M}$ must be infinite. □

We now have our main convergence result for problem LCP(q,M).

THEOREM 3.4. *Let $M=B+C$ be a splitting of the matrix $M\in\mathbb{R}^{n\times n}$ such that $B\succ 0$, and $\{x^k\}_{k\ge0}$ the sequence of iterates generated by Algorithm* 3.1 *for the choice $n_{max}=\infty$. Then, either x^K is a solution to problem* LCP(q,M) *for some integer $K\ge0$ and the algorithm terminates finitely, or*

$$\liminf_{k\ge0}\phi(x^k)=0.$$

Moreover, if the sequence $\{x^k\}_{k\ge0}$ is bounded, there exists a limit point that is a solution to LCP(q,M).

Proof. If x^K is a solution to problem LCP(q,M) for some integer $K\ge0$, then Algorithm 3.1 exits in Step 5 with the solution x^K.

For the remainder of the proof we assume that a solution is not encountered. In this case, Algorithm 3.1 generates an infinite sequence $\{x^k\}_{k\ge0}$, and Lemma 3.3 ensures that $|\mathcal{C}\cup\mathcal{M}|=\infty$. We now consider two cases.

Case 1: $|\mathcal{M}|=\infty$

In this case, Step 4 of Algorithm 3.1 and (3.15) imply that

$$\phi(x^{k+1})=\phi(x^{k,s,n_s})\le\tfrac{1}{2}\phi^k_{\max}\ \text{ for } k\in\mathcal{M}. \tag{3.38}$$

Since $|\mathcal{M}| = \infty$, it also follows from Algorithm 3.1 that $\lim_{k\to\infty} \phi^k_{\max} = 0$ so that (3.38) implies

$$\lim_{k\in\mathcal{M}} \phi(x^{k+1}) = 0.$$

Moreover, if we assume that the sequence $\{x^k\}_{k\geq 0}$ is bounded, then so is $\{x^{k+1}\}_{k\in\mathcal{M}}$. Thus, there exists $\mathcal{S} \subseteq \mathcal{M}$ such that

$$\lim_{k\in\mathcal{S}} x^{k+1} = x^* \ \text{ and } \ \lim_{k\in\mathcal{S}} \phi(x^{k+1}) = 0,$$

and, therefore,

$$\phi(x^*) = \lim_{k\in\mathcal{S}} \phi(x^{k+1}) = 0,$$

where we have used the continuity of the function $\phi(x)$. This means that x^* is a limit point of the sequence of iterates and is a solution to problem LCP(q,M).

Case 2: $|\mathcal{M}| < \infty$

Since $|\mathcal{C} \cup \mathcal{M}| = \infty$, we know that there exists an integer $\hat{k} \geq 0$ such that

$$k \in (\mathcal{C} \cup \mathcal{F}) \text{ for all } k \geq \hat{k}, \text{ and } |\mathcal{C}| = \infty. \tag{3.39}$$

We now construct a sequence $\{z^j\}_{j\geq 0}$ from the iterates generated from Algorithm 3.1 by gathering the points $\{x^k,\ x^{k,f,1},\ x^{k,f,2},\ \dots x^{k,f,n_f},\ x^{k,s,1},\ x^{k,s,2}\}$ for all $k \in \mathcal{C}$ such that $k \geq \hat{k}$. Specifically, $z^0 = x^{\hat{k}}$, $z^1 = x^{\hat{k},f,1}$, $z^2 = x^{\hat{k},f,2}$, etc., so that $\{z^j\}_{j\geq 0}$ is an infinite sequence since $|\mathcal{C}| = \infty$. Using (3.4), (3.10), (3.13), (3.14), (3.19), and the condition $\rho^k \leq \rho < 1$ from part (a) of Lemma 3.1, we deduce that

$$\|z^{k+2} - z^{k+1}\|_2 \leq \rho \|z^{k+1} - z^k\|_2 \ \text{ for } k \geq 0.$$

Using a contraction argument similar to [14, Theorem 9.23], we can show that $\{z^j\}_{j\geq 0}$ is a Cauchy sequence. Since $\mathbb{R}^n$ with $\|\cdot\|_2$ is a complete metric space, we know that

$$\text{there exists a vector } x^* \text{ such that } \lim_{j\to\infty} z^j = x^*.$$

Next, we define $\{y^j\}_{j\geq 0}$ as the subsequence of $\{z^j\}_{j\geq 0}$ consisting of all the points $\{x^k, x^{k,f,1}\}$ for all $k \in \mathcal{C}$ such that $k \geq \hat{k}$. It follows from this construction that

$$\lim_{j\to\infty} y^j = x^* \ \text{ and } \ y^{j+1} = \text{FPI}(y^j, 1, B, C) \ \text{ for all even } j \geq 0.$$

We may now use the same argument as in the proof of Case 1 of Theorem 2.4 to prove that x^* is a solution to LCP(q,M). Moreover, we have

$$\lim_{k\in\mathcal{C}} x^k = x^*$$

since $\{x^k\}_{k\in\mathcal{C}}$ is a subsequence of $\{z^j\}_{j\geq 0}$. Finally, since $k \in (\mathcal{C} \cup \mathcal{F})$ for all $k \geq \hat{k}$, $x^{k+1} = x^k$ for all $k \in \mathcal{F}$, and there are only a finite number of $\mathcal{F}$-iterations between each pair of $\mathcal{C}$ iterations, we must also have

$$\lim_{k\to\infty} x^k = x^*$$

so that the entire sequence generated by Algorithm 3.1 converges to a solution x^* of problem LCP(q,M). Clearly, $\lim_{k\to\infty} \phi(x^k) = \phi(x^*) = 0$ since ϕ is continuous and x^* is a solution to LCP(q,M), which completes the proof for this case. □

A few comments are warranted.

- The theory just described carries over to the case that iteration dependent splittings $M = B^k + C^k$ are used provided the matrices $\{B^k\}_{k\geq 0}$ are uniformly positive definite.
- The convergence result holds even if $\phi^k_{\max}$ is decreased in Step 4 when $k \in \mathcal{C}$, i.e., the kth iterate is a contraction iterate. However, to promote the acceptance of rapidly convergent subspace steps, one should not decrease the quantity "too" quickly.
- Limit points of the sequence $\{x^k\}_{k\geq 0}$ are guaranteed under the assumption that the level curves of ϕ are bounded on the orthant $x \geq 0$.

3.3. Numerical tests. In this section we illustrate the effectiveness of our MATLAB implementation of Algorithm 3.1 by solving the Black-Scholes-Merton American options pricing problem studied in [6] and randomly generated strictly diagonally dominant *asymmetric* LCPs. For comparison, we have also written our own MATLAB implementation of the two-phase matrix-splitting based algorithm described in [6] henceforth referred to as Algorithm FLMN to reflect the last names of the authors. The subspace step is computed from (2.28).

For Algorithm 3.1 we used the following control parameters: number of initial fixed-point iterations $n_f = 1$, number of additional fixed-point iterations $n_s = 2$, trust-region contraction factor $\eta_c = 0.5$, trust-region expansion factor $\eta_e = 2$, maximum iterations allowed $n_{\max} = 500$, trust-region reset factor $\Delta_R = 1.0$, maximum trust-region radius $\Delta_{\max} = 1.0e^{+12}$, splitting contraction constant $\rho_u = 0.99$, and the successive overrelaxation matrix splitting of M as given in Table 1.1. For simplicity and computational efficiency, we computed an approximate solution of problem (3.7) by first solving system (2.28) for $x_{\mathcal{I}}$ using the MATLAB "backslash" operator, and then setting

$$x_{\mathcal{I}} \leftarrow x_{\mathcal{I}} + \min\left(1, \frac{\Delta^k}{\|x_{\mathcal{I}} - x_{\mathcal{I}}^{k,f,n_f}\|_2}\right)\left(x_{\mathcal{I}} - x_{\mathcal{I}}^{k,f,n_f}\right).$$

Note that $x_{\mathcal{I}}$ then satisfies (3.8) and, therefore, is acceptable as an approximate solution to (3.7).

For both Algorithm 3.1 and Algorithm FLMN, we allow a maximum of three recursive subspace steps to be computed per major iteration. Moreover, both algorithms are terminated when either the maximum number of iterations $n_{\max}$ is reached or

$$\phi(x^k) \leq 1.0e^{-5}$$

is satisfied, which indicates that an approximate solution to problem LCP(q,M) has been identified.

3.3.1. American options pricing. Consider pricing an American put option with strike price $K > 0$ and time to maturity $T > 0$. If the put option is exercised by the holder when the underlying asset price is S, then the holder receives a pay out of $\Psi(S) = \max(K - S, 0)$. If $V(t,S)$ denotes the value of the put option at time $t \in [0,T]$ when the asset price is S, then the Black-Scholes-Merton model assumes that V solves the variational inequality [8]

$$\begin{aligned} \frac{\partial V}{\partial t} + \tfrac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r-q)S\frac{\partial V}{\partial S} - rV \leq 0, \\ \Psi \leq V, \\ \left(\frac{\partial V}{\partial t} + \tfrac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r-q)S\frac{\partial V}{\partial S} - rV\right) \circ (V - \Psi) = 0, \end{aligned}$$

for all $t \in [0,T]$ and $S \in (0,\infty)$, where $\sigma \geq 0$ is the volatility of the asset, $r \geq 0$ is the risk-free interest rate, $q \geq 0$ is the dividend yield paid by the asset, and must satisfy the terminal condition

$$V(T,S) = \Psi(S) \text{ for } S \in (0,\infty).$$

As described in [6], we may solve this problem numerically by (i) performing a nonlinear change of variables; (ii) transforming the terminal value problem into an initial value problem; and (iii) discretize the problem by using the linear finite element method in space and a Crank-Nicolson scheme in time. The end result is that we need to solve the sequence of problems $\{\mathrm{LCP}(q_j,M_j)\}_{j=0}^N$, where N is the number of time subintervals of length $\Delta t = T/N$, $M_j = \mathbb{M} + (\Delta t/2)\mathbb{A}$, $q_j = -(\mathbb{M} - (\Delta t/2)\mathbb{A})x_{j-1} + \Delta t F$,

$$\mathbb{A} = \begin{pmatrix} a_0 & a_1 & & \\ a_{-1} & a_0 & \ddots & \\ & \ddots & \ddots & a_1 \\ & & a_{-1} & a_0 \end{pmatrix}, \quad \mathbb{M} = \frac{h}{6}\begin{pmatrix} 4 & 1 & & \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4 \end{pmatrix},$$

$a_0 = 2rh/3 + \sigma^2/h$, $a_{\pm 1} = \mp\mu/2 + rh/6 - \sigma^2/(2h)$, $\mu = r - q - \sigma^2/2$, and the "load" vector F may be computed by approximating the payoff function Ψ by its linear element interpolant and then evaluating a certain bilinear form. (See [6, Section 2] for more details.)

For all tests we chose $r = 0.5$, $q = 0$, $h = 0.0025$, $N = 40$, and $K = 100$. However, the remaining parameters σ, T, x_ℓ, and x_u were varied to obtain four test cases, where x_ℓ and x_u are the lower and upper bounds in space for the discretized problem, respectively.

Table 3.1 contains the results of our tests, which compare Algorithm 3.1 with Algorithm FLMN. The columns have the following meaning: "iter" is the total number of iterations, "nsplit" is the total number of splitting iterations, "nss" is the total number of subspace iterations, and "atm" is the at-the-money value of the put option, i.e., the spot price is the same as the strike price. These results are precisely what we hoped to obtain; the performance of Algorithm 3.1 is exactly the same as Algorithm FLMN, which was shown in [6] to solve these problems very efficiently. We find comfort, however, in knowing that Algorithm 3.1 is guaranteed to converge since Assumption 3.1 holds as a result of M_j being diagonally dominant; this is a well known result for the matrix splitting that corresponds to successive overrelaxation given in Table 1.1.

Table 3.1

Results from Algorithm FLMN and Algorithm 3.1 on four scenarios for pricing an American put option.

BSM Parameters				Algorithm FLMN				Algorithm 3.1			
σ	T	x_ℓ	x_u	iter	nsplit	nss	atm	iter	nsplit	nss	atm
0.2	0.5	-0.3	0.6	67	201	67	\$4.63	67	201	67	\$4.63
0.4	0.5	-0.5	1.0	112	325	112	\$10.13	112	325	112	\$10.13
0.2	5.0	-0.3	1.6	89	263	89	\$9.89	89	263	89	\$9.89
0.4	5.0	-0.8	3.2	92	252	92	\$24.44	92	252	92	\$24.44

3.3.2. Random asymmetric LCPs. Finally, we test the efficiency of Algorithm 3.1 and the effectiveness of the subspace phase on randomly generated LCPs, where M is constructed to be diagonally dominant. We accomplished this by first defining $M \in \mathbb{R}^{1000\times 1000}$ by sampling a standard normal distribution, and then scaling every element by $1.0e^3$. Finally, we redefined the ith diagonal element to be the larger of its current value and the absolute ith row sum for $1 \le i \le 1000$.

4. Conclusions. Kočvara and Zowe [9] have shown that a two-phase algorithm based on matrix splitting iterations may be effective for solving strictly convex BQPs. Similarly, the work by Moré and Toraldo [12] demonstrates that two-phase methods based on simple projected gradient iterations may be used to efficiently solve nonconvex BQPs. For strictly convex problems, one might suspect that the more sophisticated matrix splitting iterations, e.g. successive overrelaxtion, used in [9] might be superior to the simple gradient iterations utilized in [12]. A natural question is whether their exists a two-phase method with convergence guarantees for both convex and nonconvex problems that also utilizes sophisticated matrix splittings. We presented such an algorithm in Section 2.1, proceeded to prove convergence in Section 2.2, and exhibited its effectiveness on a variety of problems in Section 2.3.

TABLE 3.2
Output from Algorithm 3.1 with and without a subspace phase on ten randomly generated diagonally dominant LCPs.

	Algorithm 3.1 (no subspace)			Algorithm 3.1 (with subspace)			
n	res	iter	nsplit	res	iter	nsplit	nss
1000	$3.56e^{-08}$	5	7	$5.06e^{-09}$	1	3	1
1000	$2.68e^{-08}$	5	7	$8.44e^{-09}$	1	3	1
1000	$3.25e^{-08}$	5	7	$3.21e^{-09}$	1	3	1
1000	$2.97e^{-08}$	5	7	$7.49e^{-09}$	1	3	1
1000	$2.20e^{-08}$	5	7	$8.21e^{-09}$	1	3	1
1000	$2.20e^{-08}$	5	7	$4.40e^{-09}$	1	3	1
1000	$2.74e^{-08}$	5	7	$3.79e^{-09}$	1	3	1
1000	$4.06e^{-08}$	5	7	$5.31e^{-09}$	1	3	1
1000	$3.54e^{-08}$	5	7	$3.86e^{-09}$	1	3	1
1000	$3.99e^{-08}$	5	7	$6.45e^{-09}$	1	3	1

We believe the numerical results clearly indicate the improved active-set identification capabilities of sophisticated matrix splittings and generally results in fewer major iterations. To be fruitful in terms of computational time, however, the matrix splitting iteration must be inexpensive. This is true, for example, of many problems that arise in the numerical solution of partial differential equations since the problem matrices tend to be very sparse.

Feng et al. [6] and Morales, Nocedal, and Smelyanskiy [11] describe two-phase methods for solving LCPs that arise from pricing American options in finance and solving contact problems in mechanics. Their algorithms, however, do not enjoy global convergence guarantees, although they have proved to be very efficient in their tests thanks to a carefully designed subspace phase. A deserving question is whether one may formulate a *provably convergent* two-phase matrix splitting algorithm that performs equally well for pricing American options. This would supply additional comfort when pricing options, but also would likely prove useful for solving more general LCPs. Although BQPs and LCPs are closely related, the formulation of a convergent algorithm for LCPs based on matrix splittings is not straightforward since there is no natural idea of an "objective function". Therefore, although we solved BQPs by calculating descent directions from matrix splitting iterations, a similar concept does not lend itself to solving LCPs. Rather, we formulated a two-phase matrix splitting algorithm in Section 3.1 by combining a contraction argument with a "natural" merit function that is based directly on the structure of LCPs. In Section 3.2 we showed that the algorithm was globally convergent and in Section 3.3 supplied numerical tests. The results show that our provably convergent two-phase method is equally effective for pricing American options. We also highlighted the usefulness of the subspace phase by solving randomly generated LCPs whose defining matrix was asymmetric and diagonally dominant. The results clearly show that the subspace phase dramatically reduces the number of major iterations. We may conclude, therefore, that our two-phase method for solving LCPs is an attractive option when the matrix splitting iteration is inexpensive and the subspace problem is not prohibitively expensive to solve.

REFERENCES

[1] I. Bongartz, A. R. Conn, N. I. M. Gould, and P. L. Toint, *CUTE: Constrained and unconstrained testing environment*, Report 93/10, Département de Mathématique, Facultés Universitaires de Namur, 1993.

[2] B. Chen, X. Chen, and C. Kanzow, *A penalized Fischer-Burmeister NCP-function*, Math. Program., 88 (2000), pp. 211–216.

[3] J.-S. Chen, *On some NCP-functions based on the generalized Fischer-Burmeister function*, Asia-Pac. J. Oper. Res., 24 (2007), pp. 401–420.

[4] R. W. Cottle, J.-S. Pang, and R. E. Stone, *The linear complementarity problem*, Computer Science and Scientific

Computing, Academic Press Inc., Boston, MA, 1992.

[5] Y.-H. Dai and R. Fletcher, *Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming*, Numer. Math., 100 (2005), pp. 21–47.

[6] L. Feng, V. Linetsky, J. Luis, and M. J. Nocedal, *On the solution of complementarity problems arising in american options pricing*, 2009.

[7] R. Fletcher, *On the Barzilai-Borwein method*, in Optimization and control with applications, vol. 96 of Appl. Optim., Springer, New York, 2005, pp. 235–256.

[8] P. Jaillet, D. Lamberton, and B. Lapeyre, *Variational inequalities and the pricing of american options*, Acta Applicandae Mathematicae, 21 (1990), pp. 263–289.

[9] M. Kočvara and J. Zowe, *An iterative two-step algorithm for linear complementarity problems*, Numer. Math., 68 (1994), pp. 95–106.

[10] MathWorks Inc., *Matlab User's Guide*, Natick, Massachusetts, 1992.

[11] J. L. Morales, J. Nocedal, and M. Smelyanskiy, *An algorithm for the fast solution of symmetric linear complementarity problems*, Numer. Math., 111 (2008), pp. 251–266.

[12] J. J. Moré and G. Toraldo, *Algorithms for bound constrained quadratic programming problems*, Numer. Math., 55 (1989), pp. 377–400.

[13] T. S. Munson, F. Facchinei, M. C. Ferris, A. Fischer, and C. Kanzow, *The semismooth algorithm for large scale complementarity problems*, INFORMS J. Comput., 13 (2001), pp. 294–311.

[14] W. Rudin, *Principles of Mathematical Analysis*, McGraw-Hill, Inc., 1976.

[15] D. Sun and J. Sun, *Strong semismoothness of the Fischer-Burmeister SDC and SOC complementarity functions*, Math. Program., 103 (2005), pp. 575–581.