QPCOMP: A Quadratic Programming Based Solver for Mixed Complementarity Problems*

Stephen C. Billups[†] and Michael C. Ferris [‡]

February 7, 1996

Abstract

QPCOMP is an extremely robust algorithm for solving mixed nonlinear complementarity problems that has fast local convergence behavior. Based in part on the NE/SQP method of Pang and Gabriel[14], this algorithm represents a significant advance in robustness at no cost in efficiency. In particular, the algorithm is shown to solve any solvable Lipschitz continuous, continuously differentiable, pseudo-monotone mixed nonlinear complementarity problem. QPCOMP also extends the NE/SQP method for the nonlinear complementarity problem to the more general mixed nonlinear complementarity problem. Computational results are provided, which demonstrate the effectiveness of the algorithm.

1 Introduction

This paper describes a new algorithm for solving the mixed nonlinear complementarity problem (MCP), which provides a significant improvement in robustness over previous superlinearly or quadratically convergent algorithms, while preserving these fast local convergence properties.

The MCP is defined in terms of a box $\mathbb{B} := \prod_{i=1}^{n}[l_i, u_i]$ and a function $f : \mathbb{B} \to \mathbb{R}^n$, where for each i, $-\infty \leq l_i < u_i \leq \infty$. The problem MCP$(f, \mathbb{B})$ is to find $x \in \mathbb{B}$ such that

$$(x-l)^\top f(x)_+ = (u-x)^\top f(x)_- = 0,$$

where $f(x)_+$ represents the projection of $f(x)$ onto the positive orthant, and $f(x)_- := f(x)_+ - f(x)$. Further, in the above definition, we agree that $\pm\infty \times 0 = 0$.

Note that by choosing $l = 0$ and $u = \infty$, the MCP reduces to the standard nonlinear complementarity problem (NCP), which is to find $x \geq 0$ such that

$$f(x) \geq 0 \qquad \text{and} \qquad x^\top f(x) = 0.$$

Complementarity problems (both MCP and NCP) arise in many applications [4, 7] and are the subject of much recent computational work. Indeed in recent years, a significant number of algorithms have been developed to solve complementarity problems. Most of these algorithms can be classified as *descent methods*; they work to minimize a nonnegative merit function, which is

*This material is based on research supported by National Science Foundation Grant CCR-9157632, Department of Energy Grant DE-FG03-94ER61915, and the Air Force Office of Scientific Research Grant F49620-94-1-0036.

[†]Department of Mathematics, University of Colorado, Denver, Colorado 80217

[‡]Computer Sciences Department, University of Wisconsin, Madison, Wisconsin 53706

chosen so that zeros of the merit function correspond to solutions of the complementarity problem. Among the algorithms included in this class are PATH [5, 15], MILES [17], SMOOTH [3], NE/SQP [14], and BDIFF [10]. Within this basic framework, there are substantial differences between the algorithms; the algorithms differ in the choice of merit function, the techniques used for determining search directions, and the globalization strategies used to guarantee descent of the merit function. However, because all of these algorithms work to minimize a merit function, their global convergence behavior is limited by the same fundamental difficulty: the merit function may have local minima that are not solutions of the complementarity problem. This difficulty manifests itself in different ways for different algorithms. In PATH and MILES, it arises as a rank-deficient basis or as a linear complementarity subproblem which is not solvable. In SMOOTH, it appears as a singular Jacobian matrix. In NE/SQP it arises as convergence to a point that fails some regularity condition.

Due to this difficulty, the best these algorithms can hope for, in terms of global convergence behavior, is to guarantee finding a solution only when the merit function has no strict local minimizers that are not global minimizers. In general, this means that the function f must be monotonic in order to guarantee convergence from arbitrary starting points.

This paper describes and implements an algorithm QPCOMP that does not suffer from the above difficulty, and hence is more robust than many other MCP algorithms. QPCOMP is based upon a strategy presented in Section 2 of this paper. This strategy provides a means of extending any algorithm which reliably solves strongly monotone MCPs so that it will solve a much broader class of problems. In particular, it will solve any problem which satisfies a pseudo-monotonicity condition at a solution. Applying this strategy to the NE/SQP algorithm[14], results in the QPCOMP algorithm.

NE/SQP is an algorithm for solving nonlinear complementarity problems that has a number of theoretical advantages. We present this algorithm in Section 3, along with extensions to the MCP framework that are necessary for its use in our context. When we tested this algorithm on our suite of test problems, we found that NE/SQP compares poorly to PATH, SMOOTH, and MILES in terms of robustness. In fact, we shall show in Section 3 that the algorithm cannot reliably solve even one dimensional monotone linear complementarity problems. However, NE/SQP works well on strongly monotone problems, which is all that is required for our strategy to work.

In Section 4, we present the QPCOMP algorithm. The main convergence result for this algorithm is given in Theorem 4.1, which shows global convergence under the assumption of pseudo-monotonicity at a solution, whenever f is a Lipschitz continuous, continuously differentiable function. The effectiveness of the algorithm is demonstrated convincingly by the test results given in Section 5. This is in spite of the poor performance of the NE/SQP algorithm on which QPCOMP is based.

Before we begin, a word about notation is in order. Iteration numbers appear as superscripts on vectors and matrices and as subscripts on scalars. Subscripts on a vector (or matrix) represent either subvectors (or submatrices) or components of the vector or matrix . For example, if M is an $n \times n$ matrix with elements $M_{jk}, j, k = 1, \ldots, n$, and J and K are index sets such that $J, K \subset \{1, \ldots, n\}$, then $M_{J,K}$ denotes the $|J| \times |K|$ submatrix of M consisting of the elements $M_{jk}, j \in J, k \in K$. Similarly, x_j represents the jth component of the vector x. The notation x_+ and x_- refers to the positive and negative components of the vector x. Specifically, x_+ is the vector whose ith component is given by $\max(x_i, 0)$, and $x_- := x_+ - x$. The directional derivative of a function $f : \mathbb{B} \to \mathbb{R}^n$ evaluated at the point x in the direction d is denoted by

$$f'(x; d) := \lim_{\lambda \downarrow 0} \frac{f(x + \lambda d) - f(x)}{\lambda},$$

provided the limit exists. Note that if x is a stationary point of f on $\mathbb{B}$, then $f'(x;d) = 0 \; \forall \, d$ such that $x+d \in \mathbb{B}$. The Euclidean and max norms are denoted by $\|\cdot\|$ and $\|\cdot\|_\infty$, respectively. Throughout the paper, we use the standard definitions of monotone and strongly monotone functions [13, Definition 5.4.2]. Similarly, in discussing convergence rates, we use the standard definitions of Q-superlinear and Q-quadratic convergence [13, Chapter 9]. Finally, we use the symbol $\mathbb{R}_+$ to represent the nonnegative real numbers.

2 The Basic Idea

As mentioned in the introduction, numerous algorithms exist which are extremely proficient at solving monotone or strongly monotone mixed complementarity problems. The challenge then is to develop an efficient algorithm that solves a broader class of problems. In this section we present a strategy for taking algorithms which work well on strongly monotone MCPs and extending them to solve MCPs for which a considerably weakened monotonicity condition is satisfied. To state this condition, we first need to define the concept of *pseudo-monotonicity*:

Definition 2.1 *Given a set $\mathbb{B} \subset \mathbb{R}^n$, the mapping $f : \mathbb{B} \to \mathbb{R}^n$ is said to be* pseudo-monotone *at a point $x^* \in \mathbb{B}$ if $\forall y \in \mathbb{B}$,*

$$f(x^*)^\top (y - x^*) \geq 0 \quad \textit{implies} \quad f(y)^\top (y - x^*) \geq 0. \tag{1}$$

f is said to be pseudo-monotone on $\mathbb{B}$ *if it is pseudo-monotone at every point in $\mathbb{B}$.*

It is known [9] that if a function $g : \mathbb{R}^n \to \mathbb{R}$ is pseudo-convex [11, Definition 9.3.1], then ∇g is a pseudo-monotone function. However, if g is only pseudo-convex at a point x^*, it does not necessarily follow that ∇g is pseudo-monotone at x^*.

Pseudo-monotonicity is a weaker condition than monotonicity. In particular, every monotone function is pseudo-monotone. But the converse is not true. For example, consider the function $f(x) := x/2 + \sin(x)$. This function is pseudo-monotone, but is not monotone. Note further that the natural merit function $\|f(x)\|^2/2$ has strict local minima that are not global minima. Thus, we see that the natural merit function of a pseudo-monotone function can have local minima that are not global minima.

In order to guarantee global convergence of our algorithm we shall require that the following assumption be satisfied:

Assumption 2.2 *MCP($f, \mathbb{B}$) has a solution x^* such that f is pseudo-monotone at x^*.*

If MCP($f, \mathbb{B}$) satisfies Assumption 2.2, we say that MCP($f, \mathbb{B}$) is *pseudo-monotone at a solution*. However, for simplicity, we will abuse terminology somewhat and say simply that MCP($f, \mathbb{B}$) is *pseudo-monotone*. This should not cause any confusion since all of our discussion will refer to problems which satisfy Assumption 2.2.

The strategy we present for pseudo-monotone MCPs is based upon extending a descent-based algorithm for strongly monotone MCPs. The idea behind a descent-based algorithm is to reformulate the MCP as a minimization problem involving a nonnegative merit function $\theta : \mathbb{B} \to \mathbb{R}_+$. The merit function is defined in such a way that $\theta(x) = 0$ if and only if x is a solution to MCP($f, \mathbb{B}$). If f is strongly monotone, it is easy to construct a merit function which has no local minima. It is then a simple task to find the global minimizer of θ, thereby giving a solution to the MCP. If however f is not monotone, then the merit function chosen will, in all likelihood, contain local minima for which $\theta \neq 0$. The algorithm may then terminate at such a local minimum, rather than at the solution.

To overcome this difficulty, we would like to find some way to "escape" from this local minimum. This can be accomplished by constructing an improved starting point $\tilde{x}$ where $\theta(\tilde{x})$ is smaller than the value of θ at the local minimum. Since the descent-based algorithm never allows the value of θ to increase, the algorithm can be restarted from $\tilde{x}$ with the guarantee that it will never return to the local minimum. Obviously, finding such an improved starting point is not a straightforward task. However, this can be achieved when the problem is pseudo-monotone. The remainder of this section describes how to construct this improved starting point.

We begin by defining a particular merit function for our algorithm: To do this, we first introduce the mapping $H : \mathbb{B} \to \mathbb{R}^n$ defined by

$$H_i(x) := \min(x_i - l_i, \max(x_i - u_i, f_i(x))). \tag{2}$$

It is easily shown that $H(x) = 0$ if and only if x solves MCP$(f, \mathbb{B})$. Using this function, we define the merit function

$$\theta(x) := \frac{1}{2} H(x)^\top H(x). \tag{3}$$

Clearly, x is a solution to MCP$(f, \mathbb{B})$ if and only if x is a minimizer of θ with $\theta(x) = 0$.

In Section 3 we will present a basic algorithm for solving strongly monotone MCPs, which is based on minimizing this particular choice of θ. However, for now, we simply assume that such an algorithm exists. Moreover we assume that the algorithm will fail in a *finite number* of iterations whenever it cannot solve the problem.

Now suppose the basic algorithm fails at a point x^0. Our strategy will be to solve a sequence of perturbed problems, generating a sequence of solutions $\{x^k\}$ that leads to an improved starting point $\tilde{x}$. The perturbed problems we solve are based on the following perturbation of f: given a *centering point* $\bar{x} \in \mathbb{B}$, and a number $\lambda > 0$, let

$$f^{\lambda,\bar{x}}(x) := f(x) + \lambda(x - \bar{x}).$$

If f is Lipschitz continuous, then for λ large enough, $f^{\lambda,\bar{x}}$ is strongly monotone. Thus, the basic algorithm will be able to solve the perturbed problem MCP$(f^{\lambda,\bar{x}})$.

With a sufficiently large λ we can then generate a sequence of iterates as follows: given a point x^0, then for $k = 0, \ldots$, choose x^{k+1} as the solution to MCP$(f^{\lambda,x^k}, \mathbb{B})$. Note that every subproblem in the sequence uses the same choice of λ, but a different choice of centering point. In particular the centering point for one subproblem is the solution of the previous subproblem. This is very reminiscent of the proximal point algorithm [16] and of Tikhonov regularization [18].

The following lemma gives sufficient conditions for a subsequence of these iterates to converge to a solution of MCP$(f, \mathbb{B})$.

Theorem 2.3 *Let $\lambda > 0$ and let $\{x^k\}, k = 0, 1, \ldots$ be a sequence of points in $\mathbb{B}$ such that for each k, x^{k+1} is a solution to MCP($f^{\lambda,x^k}, \mathbb{B}$). If MCP($f, \mathbb{B}$) satisfies Assumption 2.2, then*

1. *$\{x^k\}$ has a subsequence that converges to a solution $\bar{x}$ of MCP($f, \mathbb{B}$);*
2. *every accumulation point of $\{x^k\}$ is a solution of MCP($f, \mathbb{B}$);*
3. *if f is pseudo-monotone at any accumulation point $\bar{x}$ of $\{x^k\}$, then the iterates converge to $\bar{x}$.*

Proof Let x^* be the solution to $\mathrm{MCP}(f, \mathbb{B})$ given by Assumption 2.2. Since x^{k+1} is a solution to $\mathrm{MCP}(f^{\lambda,x^k}, \mathbb{B})$, then for each component i, exactly one of the following is true:

1. $x_i^{k+1} = l_i$ and $f_i(x^{k+1}) + \lambda(x_i^{k+1} - x_i^k) \geq 0$,
2. $l_i < x_i^{k+1} < u_i$ and $f_i(x^{k+1}) + \lambda(x_i^{k+1} - x_i^k) = 0$,
3. $x_i^{k+1} = u_i$ and $f_i(x^{k+1}) + \lambda(x_i^{k+1} - x_i^k) \leq 0$,

Let I_l, I_f and I_u be the sets of indices which satisfy the first, second, and third conditions respectively.

For $i \in I_l$, it follows that $0 \leq x_i^k - x_i^{k+1} \leq f_i(x^{k+1})/\lambda$. Also, $x_i^{k+1} - x_i^* = l_i - x_i^* \leq 0$, so

$$(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1}) \geq f_i(x^{k+1})(x_i^{k+1} - x_i^*)/\lambda. \tag{4}$$

By similar reasoning, this inequality holds for $i \in I_u$. Finally, for $i \in I_f$, $f_i^{\lambda,x^k}(x^{k+1}) = 0$, so $x_i^k - x_i^{k+1} = f_i(x^{k+1})/\lambda$, whereupon it follows that (4) is satisfied as an equality.

Thus in all cases, inequality (4) is satisfied, which gives us the following.

$$\begin{aligned}
(x_i^k - x_i^*)^2 &= (x_i^{k+1} - x_i^* + x_i^k - x_i^{k+1})^2 \\
&= (x_i^{k+1} - x_i^*)^2 + 2(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1}) + (x_i^k - x_i^{k+1})^2 \\
&\geq (x_i^{k+1} - x_i^*)^2 + 2f_i(x^{k+1})(x_i^{k+1} - x_i^*)/\lambda + (x_i^k - x_i^{k+1})^2 \qquad \text{by (4).}
\end{aligned}$$

Summing over all components,

$$\left\| x^k - x^* \right\|^2 \geq \left\| x^{k+1} - x^* \right\|^2 + 2f(x^{k+1})^\top \left(x^{k+1} - x^* \right)/\lambda + \left\| x^k - x^{k+1} \right\|^2.$$

Under Assumption 2.2, the inner product term above is nonnegative. Thus,

$$\left\| x^k - x^* \right\|^2 \geq \left\| x^{k+1} - x^* \right\|^2 + \left\| x^k - x^{k+1} \right\|^2,$$

so $\{\left\| x^k - x^* \right\|\}$ is a decreasing sequence, and $\left\| x^k - x^{k+1} \right\| \to 0$. It follows that $\{x^k\}$ has an accumulation point. Let $\bar{x}$ be any accumulation point. Then there is a subsequence $\{x^{k_j} : j = 0, 1, \ldots\}$ converging to $\bar{x}$. Since $\left\| x^k - x^{k+1} \right\| \to 0$, we also see that $x^{k_j+1} \to \bar{x}$. Finally, since x^{k_j+1} solves $\mathrm{MCP}(f^{\lambda,x^{k_j}}, \mathbb{B})$, we conclude by a straightforward continuity argument that $\bar{x}$ solves $\mathrm{MCP}(f^{\lambda,\bar{x}}, \mathbb{B})$, which implies that $\bar{x}$ solves $\mathrm{MCP}(f, \mathbb{B})$. This proves items 1 and 2.

To prove item 3, note that if f is pseudo-monotone at an accumulation point $\bar{x}$, then by item 2, $\bar{x}$ is a solution, so the above analysis can be repeated with x^* replaced by $\bar{x}$. We can then conclude that $\left\{ \left\| x^k - \bar{x} \right\| \right\}$ is a decreasing sequence. But since $\bar{x}$ is an accumulation point of $\{x^k\}$, it follows that $\left\| x^k - \bar{x} \right\| \to 0$. $\square$

Note that Theorem 2.3 did not make any assumptions on the choice of λ. Thus, even if λ is too small to ensure that $f^{\lambda,\bar{x}}$ is strongly monotone, the strategy will still work so long as each subproblem is solvable.

To illustrate the technique, it is useful to look at a simple example. Let $\mathbb{B} := \mathbb{R}_+$ and let $f : \mathbb{R}_+ \to \mathbb{R}$ be defined by

$$f(x) = (x - 1)^2 - 1.01.$$

Table 1: Iterates produced by solving sequence of perturbed problems, with ($\lambda = 1.1$)

k	x^k	$f(x^k)$	$\theta(x^k)$
0	0	-.01	.00005
1	.9110	-1.0021	.5021
2	1.5521	-.7052	.2487
3	1.8356	-.3118	.0486
4	1.9439	-.1191	.0071
5	1.9832	-.0433	.00094
6	1.9973	-.0155	.00012
7	2.0023	-.0055	.00002

This deceptively simple problem proved intractable for all of the descent-based methods we tested. In particular, we tried to solve this problem using PATH, MILES, NE/SQP, and SMOOTH. All four algorithms failed from a starting point of $x = 0$. But this should not be surprising since f is not monotone. However, f is pseudo-monotone on $\mathbb{B}$. Thus, it is easily solved by our technique. For example, using $\lambda = 1.1$ and a starting point $x^0 = 0$, the strategy generates the sequence of iterates shown in Table 1.

Note that at the 7th iteration, an improved starting point is found, (i.e, $\theta(x^7) < \theta(x^0)$). At this point, a basic algorithm (e.g., Newton's method) can be used to obtain the final solution.

In this section, we have introduced a basic strategy for taking descent-based algorithms that solve strongly monotone MCPs, and extending them to solve pseudo-monotone MCPs. This is, in fact, the main idea presented in this paper. However, to turn this strategy into a working algorithm, a number of details must be addressed:

1. We must ensure that the basic algorithm (for solving the strongly monotone MCPs) terminates in a finite number of iterations. This issue will be addressed in detail in Section 3.

2. Since we require finite termination of the basic algorithm, we must allow inexact solutions of the perturbed subproblems. We shall therefore need to incorporate control parameters into our strategy which govern the accuracy demanded by each subproblem. In the our actual implementation of the algorithm we demand very little accuracy for each subproblem. In fact, except in extreme circumstances, we allow only one step of the basic algorithm before updating the perturbed problem. To guarantee convergence of this approach requires more laborious analysis which we defer until Section 4.

3. Since we have no a priori information regarding the Lipschitz continuity of f, we shall have to incorporate some adaptive strategy for choosing λ in order to ensure that, eventually, the subproblems all become solvable.

The next two sections of the paper are devoted to addressing these details.

3 Subproblem Solution

In this section, we present an algorithm for solving strongly monotone MCPs, which is based on the NE/SQP algorithm [14]. NE/SQP was originally developed as a method for solving the nonlinear complementarity problem. When it was first introduced, NE/SQP offered a significant advance in the robustness of NCP solvers because the subproblems it needs to solve at each iteration are convex quadratic programs, which are always solvable. Today, its robustness has been greatly surpassed by PATH, MILES, and SMOOTH (see Section 5). However, NE/SQP is still a viable technique for solving strongly monotone MCPs. Moreover, NE/SQP has the very desirable feature of evaluating the function f only on its domain $\mathbb{B}$. This is in marked contrast to the SMOOTH algorithm which requires f to be defined on all of $\mathbb{R}^n$.

In this section, we first present the NE/SQP algorithm extended to the MCP framework. Since the development closely parallels that given in [14], we are deliberately terse in our presentation. Moreover, we omit the proofs to Proposition 3.1, Theorem 3.4 and Lemma 3.5. However, detailed proofs for these results are given in [1, Chapter 2]. Once the extended NE/SQP algorithm is presented we will then modify it to ensure finite termination. We note that Gabriel [8] also extended NE/SQP to address the upper bound nonlinear complementarity problem, a special case of MCP where $l = 0$ and $u > 0$ is finite.

3.1 Extension of NE/SQP to the MCP Framework

Recall that a vector x solves MCP$(f, \mathbb{B})$ if and only if $\theta(x) = 0$, where θ is defined by (2) and (3). The NE/SQP algorithm attempts to solve this problem by solving the minimization problem $\min_{x\in\mathbb{B}} \theta(x)$. We will use θ as a merit function for the MCP. To describe the algorithm in detail we need to partition the indices $\{1, \ldots, n\}$ into five sets as follows:

$$\begin{array}{rcl} I_l(x) & = & \{i : x_i - l_i < f_i(x)\} \\ I_{el}(x) & = & \{i : x_i - l_i = f_i(x)\} \\ I_f(x) & = & \{i : x_i - u_i < f_i(x) < x_i - l_i\} \end{array} \qquad \begin{array}{rcl} I_{eu}(x) & = & \{i : x_i - u_i = f_i(x)\} \\ I_u(x) & = & \{i : x_i - u_i > f_i(x)\}. \end{array}$$

It will at times be convenient to refer also to the index sets $J_l(x) := I_l(x) \bigcup I_{el}(x)$ and $J_u(x) := I_u(x) \bigcup I_{eu}(x)$. As in the original description of NE/SQP, the subscripts of these sets are chosen to reflect their meaning. For example, the subscripts l, f, and u correspond to the indices where $H_i(x) = (x_i - l_i), f_i(x)$, and $(x_i - u_i)$ respectively. The subscripts el and eu correspond to the indices where $f_i(x)$ is equal to l_i and u_i, respectively.

These index sets are used to define an iteration function $\phi : \mathbb{B} \times \mathbb{R}^n \to \mathbb{R}$ as follows: $\phi(x, d) := \sum_{i=1}^n \phi_i(x, d)$, where

$$\phi_i(x, d) := \begin{cases} \frac{1}{2}(x_i - l_i + d_i)^2 & i \in I_l(x) \bigcup I_{el}(x) \\ \frac{1}{2}(x_i - u_i + d_i)^2 & i \in I_u(x) \bigcup I_{eu}(x) \\ \frac{1}{2}(f_i(x) + \nabla f_i(x)^\top d)^2 & i \in I_f(x) \end{cases} \qquad i = 1, \ldots, n.$$

Given a point $x^k \in \mathbb{B}$, the algorithm chooses a descent direction d^k by solving the convex quadratic programming problem (QP$_k$) given by

$$\mathrm{QP_k} : \min_{\mathrm{x^k + d} \in \mathbb{B}} \phi(\mathrm{x^k}, \mathrm{d}).$$

We note that in the original NE/SQP algorithm, an additional constraint was added to this quadratic program, namely, $d_i = 0$ if $f_i(x^k) = 0$ and $x_i^k = l_i$ or $x_i^k = u_i$. However, this constraint is unnecessary for the convergence results, so we omit it from our algorithm.

To ensure descent of the merit function θ, we will need to perform a linesearch along the direction d^k. To describe this linesearch, we use a forcing function $z : \mathbb{B} \times \mathbb{R}^n \to \mathbb{R}_+$, defined by $z(x,d) := \sum_{i=1}^n z_i(x,d)$, where

$$z_i(x,d) := \begin{cases} \frac{1}{2} d_i^2 & i \notin I_f(x) \\ \frac{1}{2}(\nabla f_i(x)^\top d)^2 & i \in I_f(x) \end{cases} \qquad i = 1, \ldots, n.$$

This forcing function will be used to guarantee sufficient decrease in the merit function at each iteration. The following proposition summarizes some essential properties of the functions ϕ and z:

Proposition 3.1 ([1], Lemmas 2.2.5 and 2.2.6) *The following properties hold:*

1. *If $x^k \in \mathbb{B}$, then (QP_k) has at least one optimal solution.*
2. *$\phi(x,d) - \phi(x,0) - z(x,d) \geq \theta'(x;d)$ for all $(x,d) \in \mathbb{B} \times \mathbb{R}^n$.*
3. *If d^k is an optimal solution to (QP_k) and $\phi(x,d^k) < \phi(x,0)$, then for any $\sigma \in (0,1)$, there exists a scalar $\bar{\tau} > 0$ such that for all $\tau \in [0,\bar{\tau}]$*
$$\theta(x + \tau d^k) - \theta(x) \leq -\sigma \tau z(x, d^k).$$
4. *If d^k is an optimal solution to (QP_k), then $z(x^k, d^k) \leq \theta(x^k)$.*

Item (1) in the above proposition ensures that each QP is solvable. Item (2) ensures that the solution to the QP will be a descent direction for θ unless x is a stationary point of θ. Item (3) allows us to use a Armijo type linesearch which will be guaranteed to terminate in a finite number of iterations. Item (4) will be used in the proof of Theorem 3.14. We now state the algorithm.

Algorithm NE/SQP

Step 1 [Initialization] Select $\rho, \sigma \in (0,1)$, and a starting vector $x^0 \in \mathbb{B}$. Set $k = 0$.

Step 2 [Direction generation] Solve (QP$_k$), giving the direction d^k.
If $\phi(x^k, d^k) = \theta(x^k)$, terminate the algorithm; otherwise, continue.

Step 3 [Steplength determination] Let m_k be the smallest nonnegative integer m such that

$$\theta(x^k + \rho^m d^k) - \theta(x^k) \leq -\sigma \rho^m z(x^k, d^k); \tag{5}$$

set $x^{k+1} = x^k + \rho^{m_k} d^k$.

Step 4 [Termination check] If x^{k+1} satisfies a prescribed stopping rule, stop. Otherwise, return to Step 2, with k replaced by $k+1$.

The convergence results of this algorithm are based on two regularity conditions: *b-regularity* and *s-regularity*. It is convenient to partition the index sets as follows in order to define these regularity conditions.

$$\begin{aligned} I_{el}^+(x) &= \{i \in I_{el} : x_i - l_i > 0\} \\ I_{el}^0(x) &= \{i \in I_{el} : x_i - l_i = 0\} \\ I_f^l(x) &= \{i \in I_f : x_i - l_i = 0\} \\ I_f^n(x) &= \{i \in I_f : x_i - u_i < 0 < x_i - l_i\} \end{aligned} \qquad \begin{aligned} I_f^u(x) &= \{i \in I_f : x_i - u_i = 0\} \\ I_{eu}^0(x) &= \{i \in I_{eu} : x_i - u_i = 0\} \\ I_{eu}^-(x) &= \{i \in I_{eu} : x_i - u_i < 0\}. \end{aligned}$$

Note that for $x \in \mathbb{B}$, the sets $I_l(x), I_{el}^+(x), I_{el}^0(x), I_f^l(x), I_f^n(x), I_f^u(x), I_{eu}^0, I_{eu}^-(x)$, and $I_u(x)$ form a partition of the indices $\{1, \ldots, n\}$.

Definition 3.2 *A nonnegative vector x is said to be b-regular if for every index set α satisfying*

$$I_f^n(x) \subset \alpha \subset I_f(x) \bigcup I_{el}(x) \bigcup I_{eu}(x),$$

the principal submatrix $\nabla_\alpha f_\alpha(x)$ is nonsingular.

Definition 3.3 *A nonnegative vector x is said to be s-regular if the following linear inequality system has a solution in y:*

$$\begin{array}{rlrl} x_i - l_i + y_i = 0 & i \in I_l(x) & f_i(x) + \nabla f_i(x)^\top y \leq 0 & i \in I_f^u(x) \\ x_i - u_i + y_i = 0 & i \in I_u(x) & x_i - l_i + y_i \leq 0 & i \in I_{el}^+(x) \\ f_i(x) + \nabla f_i(x)^\top y = 0 & i \in I_f^n(x) & f_i(x) + \nabla f_i(x)^\top y \leq 0 & i \in I_{el}^+(x) \\ x_i - l_i + y_i \geq 0 & i \in I_f^l(x) & x_i - u_i + y_i \geq 0 & i \in I_{eu}^-(x) \\ f_i(x) + \nabla f_i(x)^\top y \geq 0 & i \in I_f^l(x) & f_i(x) + \nabla f_i(x)^\top y \geq 0 & i \in I_{eu}^-(x) \\ x_i - u_i + y_i \leq 0 & i \in I_f^u(x) & y_i = 0 & i \in I_{el}^0(x) \bigcup I_{eu}^0(x). \end{array} \tag{6}$$

Note that when $l = 0, u = \infty$ the above definition is identical to the concept of s-regularity [14, Definition 1].

The following theorem parallels the convergence results of [14, Theorems 1 and 2] and establishes the fact that the NE/SQP algorithm has very good local convergence behavior.

Theorem 3.4 ([1], Theorems 2.2.12 and 2.2.15) *Let $f : \mathbb{B} \to \mathbb{R}^n$ be a once continuously differentiable function. Let $x^0 \in \mathbb{B}$ be arbitrary. The following statements hold:*

1. *NE/SQP generates a well defined sequence of iterates $\{x^k\}$, with $x^k \in \mathbb{B}$, along with a sequence of optimal solutions $\{d^k\}$ for the subproblems (QP$_k$);*

2. *if x^* is an accumulation point of $\{x^k\}$, and if x^* is both b-regular and s-regular, then the following hold:*

   (a) *x^* is a solution of MCP($f, \mathbb{B}$).*

   (b) *there exists an integer $K > 0$ such that for all $k \geq K$, the stepsize $\tau_k = \rho^{m_k} = 1$, hence, $x^{k+1} = x^k + d^k$;*

   (c) *the sequence $\{x^k\}$ converges to x^* Q-superlinearly;*

   (d) *if ∇f is Lipschitzian in a neighborhood of x^*, then the convergence is Q-quadratic.*

The global convergence results contained above are not very useful from a practical standpoint. The problem is that the s-regularity and b-regularity conditions are dependent not only on the problem, but also on the algorithm. In particular, they depend on the particular choice of merit function used. A result that will be more useful for our purposes is available as a result of the following lemma:

Lemma 3.5 ([1], Lemma 2.2.17) *If f is strongly monotone, then all points $x \in \mathbb{B}$ are both b-regular and s-regular.*

It should be noted that the strong monotonicity assumption above is essential. For example, consider the monotone function $f : \mathbb{R}_+ \to \mathbb{R}$ given by $f(x) = 1$, and let $\mathbb{B} := \mathbb{R}_+$. For this choice of f and $\mathbb{B}$, it is easily verified that $\forall x > 1$, x is neither b-regular or s-regular. As a consequence, even though MCP($f, \mathbb{B}$) has the trivial solution $x = 0$, NE/SQP fails to find it with any starting point $x > 1$. Thus, we see that NE/SQP cannot be relied upon to solve monotone linear complementarity problems.

We now state our main convergence result of the NE/SQP algorithm.

Theorem 3.6 *Suppose f is strongly monotone. If x^* is an accumulation point of the iterates $\{x^k\}$ produced by the NE/SQP algorithm, then x^* is a solution of MCP($f, \mathbb{B}$) and the sequence $\{x^k\}$ converges to x^* with the local convergence rates specified in Theorem 3.4.*

Proof By Lemma 3.5, x^* is both b-regular and s-regular. Therefore, by Theorem 3.4, x^* is a solution of MCP($f, \mathbb{B}$) and the iterates $\{x^k\}$ generated by the NE/SQP algorithm converge to x^* with convergence rates specified in Theorem 3.4. □

3.2 Modification of NE/SQP to Guarantee Finite Termination

The NE/SQP algorithm has the drawback that it does not necessarily terminate in a finite number of iterations unless it converges to a solution. In particular, while the algorithm guarantees descent of θ at every iteration, the sequence $\{\theta(x^k)\}$ may not converge to 0. This can happen either by generating an unbounded sequence of points, or by converging slowly to an irregular point. This will clearly be unacceptable if we are to use the algorithm to solve a sequence of perturbed subproblems. We therefore present a modified NE/SQP algorithm which has the same local convergence properties as the original NE/SQP algorithm, but which also guarantees finite termination, even when it fails.

Modified NE/SQP Algorithm

Step 1 [Initialization] Given a starting vector $x^0 \in \mathbb{B}$, a convergence tolerance *tol*, and termination parameters $\gamma \in (0,1)$, and $\Delta \geq 11$, select $\rho, \sigma \in (0,1)$, and set $k = 0$.

Step 2 [Direction generation] Solve (QP$_k$), giving the direction d^k.
If $\phi(x^k, d^k) \geq (1-\gamma)\theta(x^k)$, or if $\left\|d^k\right\|^2 > \Delta\theta(x^0)$, then terminate the algorithm, returning the point x^k along with a failure message; otherwise, continue.

Step 3 [Steplength determination] Let m_k be the smallest nonnegative integer m such that

$$\theta(x^k + \rho^m d^k) - \theta(x^k) \leq -\sigma\rho^m z(x^k, d^k). \tag{7}$$

Set $x^{k+1} = x^k + \rho^{m_k} d^k$ and continue.

Step 4 [Termination check] If $\theta(x^{k+1}) \leq tol$ terminate the algorithm, returning the solution x^{k+1}. Otherwise, return to Step 2, with k replaced by $k+1$.

Note that by setting $\gamma = 0$ and $\Delta = \infty$, the modified algorithm is identical to NE/SQP, with the addition of a particular stopping criteria in Step 4. However, by choosing $\gamma \in (0,1)$ and $\Delta < \infty$, we can ensure that the algorithm will terminate in a finite number of iterations, which we will prove in Theorem 3.14. This has the drawback that the modified algorithm may fail when the original algorithm would have succeeded. However, we shall overcome this drawback in the QPCOMP algorithm by carefully controlling the parameter γ. Moreover, the modified algorithm also has the same local convergence properties as the original algorithm. To establish this fact, we use the following two lemmas to show that if x^k is near a b-regular solution of MCP($f, \mathbb{B}$), then the tests in Step 2 can never cause failure.

Lemma 3.7 ([1], Lemma 2.2.14) *Let $\bar{x}$ be a solution of MCP($f, \mathbb{B}$). If $\bar{x}$ is b-regular, then there exists a constant $c > 0$ such that for any vector $x^k \in \mathbb{B}$ close enough to $\bar{x}$,*

$$\left\|d^k\right\| \leq c\left\|H(x^k)\right\|,$$

where d^k is any solution to the quadratic program (QP$_k$).

Observe, that when x^k is close enough to a b-regular solution, $\left\| H(x^k) \right\| \leq \left\| H(x^0) \right\| / c$, so $\left\| d^k \right\| \leq \left\| H(x^0) \right\|$, and therefore, $\left\| d^k \right\|^2 \leq \Delta\theta(x^0)$. Thus, when x^k is close to a b-regular solution, the second test in Step 2 of the Modified NE/SQP algorithm cannot cause failure. We now show that the first test in Step 2 cannot cause failure either.

Lemma 3.8 ([1], Lemma 2.2.19) *Let $\bar{x}$ be a solution of MCP($f, \mathbb{B}$). If $\bar{x}$ is b-regular, then for any $\epsilon \in (0, 1/2)$, there is a neighborhood $N \subset \mathbb{B}$ of $\bar{x}$ such that if $x^k \in N$, then*

$$\phi(x^k, d^k) \leq \epsilon\theta(x^k),$$

where d^k is an optimal solution of (QP_k).

The above lemmas show that for x^k close enough to $\bar{x}$, the modified algorithm will not terminate in Step 2, as long as $\bar{x}$ is b-regular. Thus, the modified algorithm has the same local convergence properties as the original algorithm. This establishes the following theorem:

Theorem 3.9 *Under the conditions of Theorem 3.4, the Modified NE/SQP algorithm generates a well defined sequence of iterates $\{x^k\} \subset \mathbb{B}$, along with a sequence of optimal solutions $\{d^k\}$ for the subproblems (QP_k). Furthermore, if x^* is an accumulation point of $\{x^k\}$, and if either f is strongly monotone, or x^* is both b-regular and s-regular, then x^* is a solution of MCP($f, \mathbb{B}$) and the iterates converge to x^* at the rates specified in Theorem 3.4.*

The remainder of this section is aimed at proving that the Modified NE/SQP algorithm terminates. This is accomplished by considering what happens if the algorithm does not terminate. In this case, we shall show that the iterates $\{x^k\}$ converge to a point x^*. Using this fact, we will place bounds on certain quantities, which will then be used to establish a minimum rate of decrease for the merit function θ. This will then force the merit function to zero, which means that the algorithm will terminate after all, by the test in Step 4.

For ease of discussion, we define the function $\phi_x(d) := \phi(x, d)$. The following lemma is a technical result needed in several ensuing proofs.

Lemma 3.10 ([1], Lemma 2.2.21) *If $\phi_x(d) \leq (1-\gamma)\theta(x)$ then $z(x, d) \geq \frac{1}{2}\gamma^2\theta(x)$.*

We now prove that the iterates converge.

Lemma 3.11 *Suppose f is continuously differentiable. If the Modified NE/SQP algorithm, with $\gamma \in (0, 1)$ and $\Delta < \infty$, fails to terminate, then the iterates $\{x^k\}$ produced by the algorithm will converge to a point $x^* \in \mathbb{B}$ with $\theta(x^*) > 0$.*

Proof Let $\phi_k(d) := \phi(x^k, d)$ and let $z_k(d) := z(x^k, d)$. By the test in Step 2 of the algorithm, $\phi_k(d) \leq (1-\gamma)\theta(x^k)$. Thus, by Lemma 3.10, $z_k(d) \geq \frac{1}{2}\gamma^2\theta(x^k)$.

Let $\{\tau_k\}$ be the sequence of steplengths generated in step 3 of the algorithm, i.e., $\tau_k := \rho^{m_k}$. Then,

$$\begin{aligned}
\theta(x^{k+1}) &= \theta(x^k + \tau_k d^k) \\
&\leq \theta(x^k) - \sigma\tau_k z_k(d^k) \qquad \text{(by the linesearch test (7))} \\
&\leq \theta(x^k) - \sigma\tau_k\gamma^2\theta(x^k)/2 \quad \text{(by Lemma 3.10)} \\
&= (1 - \tfrac{\sigma\tau_k\gamma^2}{2})\theta(x^k).
\end{aligned}$$

Let $\hat{\beta}_k := \sigma\tau_k\gamma^2/2$. Then

$$\theta(x^{k+1}) \le \theta(x^0)\prod_{j=0}^{k}(1-\hat{\beta}_j).$$

Since $\theta(x^k)$ is bounded away from 0, it follows that

$$\prod_{k=0}^{\infty}(1-\hat{\beta}_k) > 0.$$

But this implies that $\sum_{i=0}^{\infty}\hat{\beta}_k$ is finite, which means that $\sum_{i=0}^{\infty}\tau_k$ is finite.

Now, by the test in Step 2 of the algorithm, $\left\|d^k\right\|^2 \le \Delta\theta(x^0)$. Thus, $\left\|d^k\right\|$ is bounded, so

$$\sum_{k=0}^{\infty}\tau_k\left\|d^k\right\| < \infty.$$

From this it follows that the sequence of iterates $\{x^k\}$ converges to some point x^*. Clearly, $\theta(x^*) > 0$, or the algorithm would terminate in Step 4. □

Using the fact that the iterates converge, together with straightforward continuity arguments, bounds can be placed on several quantities, which will be useful in proving Lemma 3.13.

Lemma 3.12 ([1], Lemma 2.2.23) *Under the hypotheses of Lemma 3.11, there exist constants M_1, M_2, and L, depending on the starting point x^0, such that for all $\tau \in [0,1]$, the following inequalities hold:*

$$|f_i(x^k+\tau d^k)| \le M_1, \qquad |\nabla f_i(x^k+\tau d^k)| \le M_2 \tag{8}$$

and

$$f_i(x^k) - \tau L\left\|d^k\right\| \le f_i(x^k+\tau d^k) \le f_i(x^k) + \tau L\left\|d^k\right\|. \tag{9}$$

Furthermore, for any $\delta > 0$, we can choose $\hat{\tau}(\delta) > 0$ such that for k sufficiently large, the following holds for all $\tau \in [0,\hat{\tau}(\delta)]$:

$$|f_i(x^k+\tau d^k)| \le \left|f_i(x^k) + \tau\nabla f_i(x^k)^\top d^k\right| + \tau\delta\left\|d^k\right\|. \tag{10}$$

We are now able to establish a minimum rate of decrease for the merit function.

Lemma 3.13 *Under the hypotheses of Lemma 3.11, there exists a constant $\hat{\rho} \in (0,1)$ such that*

$$\theta(x^{k+1}) \le \hat{\rho}\theta(x^k), \quad \forall\ k \text{ sufficiently large.}$$

Proof Suppose $\delta \in (0,1)$, and let $\tau \in [0,\hat{\tau}(\delta)]$ where $\hat{\tau}(\delta)$ is chosen according to Lemma 3.12. Suppose that k is large enough that (10) holds. We shall examine the terms $H_i(x^k+\tau d^k)^2$ in order to establish an upper bound on $\theta(x^{k+1}) = \sum_i H_i(x^k+\tau d^k)^2/2$.

To simplify notation, we drop the superscripts k. Thus, we let $x := x^k$ and $d := d^k$, etc. We shall also find it convenient to define the scalar function $\hat{\phi}_i : \mathbb{R}_+ \to \mathbb{R}_+$, as follows:

$$\hat{\phi}_i(\tau) \quad := \quad \phi_i(x,\tau d).$$

Observe that $\hat{\phi}_i''(0) = z_i(x,d)$, so

$$\sum_{i=1}^{n}\hat{\phi}_i''(0) = z(x,d). \tag{11}$$

To bound $H_i(x+\tau d)^2$, we have to look at two different cases:

Case 1: $i \in I_f(x)$**.** Note that $|H_i(x+\tau d)| \le |f_i(x+\tau d)|$. Thus, by (10),

$$H_i(x+\tau d)^2 \le (f_i(x)+\tau\nabla f_i(x)^\top d)^2 + 2\tau\delta\left|f_i(x)+\tau\nabla f_i(x)^\top d\right|\|d\| + \tau^2\delta^2\|d\|^2.$$

But, $(f_i(x)+\tau\nabla f_i(x)^\top d)^2 = 2\hat\phi_i(\tau) = 2\hat\phi_i(0) + 2\tau\hat\phi_i'(0) + \tau^2\hat\phi_i''(0)$, so

$$H_i(x+\tau d)^2 \le 2\hat\phi_i(0) + 2\tau\hat\phi_i'(0) + \tau^2\hat\phi_i''(0) + 2\tau\delta\left|f_i(x)+\tau\nabla f_i(x)^\top d\right|\|d\| + \tau^2\delta^2\|d\|^2. \quad (12)$$

Case 2: $i \notin I_f(x)$**.** We look only at the case $i \in I_l(x)\bigcup I_{el}(x)$; the argument for $i \in I_u(x)\bigcup I_{eu}(x)$ is similar.

If $H_i(x+\tau d)$ is negative, then

$$\begin{aligned} H_i(x+\tau d) &= f_i(x+\tau d) \\ &\ge f_i(x) - \tau L\|d\| \quad \text{by (9)} \\ &\ge x_i - l_i + \tau d_i - \tau(d_i + L\|d\|) \quad \text{since } f_i(x) \ge x_i - l_i \\ &\ge x_i - l_i + \tau d_i - \tau(L+1)\|d\|. \end{aligned}$$

Thus,

$$\begin{aligned} H_i(x+\tau d)^2 &\le (x_i - l_i + \tau d_i)^2 - 2\tau(x_i - l_i + \tau d_i)(L+1)\|d\| + \tau^2(L+1)^2\|d\|^2 \\ &\le (x_i - l_i + \tau d_i)^2 + \tau^2(L+1)^2\|d\|^2. \end{aligned}$$

If $H_i(x+\tau d)$ is nonnegative, this inequality holds trivially since $H_i(x+\tau d_i) \le x_i - l_i + \tau d_i$. Finally, $(x_i - l_i + \tau d_i)^2 = 2\hat\phi_i(\tau) = 2\hat\phi_i(0) + 2\tau\hat\phi_i'(0) + \tau^2\hat\phi_i''(0)$, so

$$H_i(x+\tau d)^2 \le 2\hat\phi_i(0) + 2\tau\hat\phi_i'(0) + \tau^2\hat\phi_i''(0) + \tau^2(L+1)^2\|d\|^2. \quad (13)$$

Summing over all components, we get

$$\theta(x+\tau d) = \frac{1}{2}\sum H_i(x+\tau d)^2 \le \phi_x(0) + \tau\phi_x'(0;d) + \tau\delta\eta + \tau^2\zeta, \quad (14)$$

where

$$\eta := \sum_{i\in I_f(x)}\left|f_i(x)+\tau\nabla f_i(x)^\top d\right|\|d\|,$$

and

$$\zeta := \sum_{i=1}^n \hat\phi_i''(0) + \sum_{i\notin I_f(x)}(L+1)^2\|d\|^2 + \sum_{i\in I_f(x)}\delta^2\|d\|^2.$$

We now establish bounds for η and ζ. By (8),

$$\sum_{i\in I_f(x)}\left|f_i(x)+\tau\nabla f_i(x)^\top d\right| \le \|f(x)\| + \tau M_2\|d\| \le M_1 + \tau M_2\sqrt{\Delta\theta(x^0)} =: C_1.$$

Thus, $\eta \le C_1\|d\| \le C_1\sqrt{\Delta\theta(x^0)} =: K_1$.

For ζ, we deduce from (11) that

$$\sum_{i=1}^n \hat\phi_i''(0) = z(x,d) \le \theta(x), \qquad \text{by item 4 of Proposition 3.1.}$$

Thus,

$$\begin{aligned}\zeta &\le \theta(x) + \|d\|^2\left(n(L+1)^2 + n\delta^2\right)\\ &\le \left(1 + n\Delta((L+1)^2+\delta^2)\right)\theta(x^0), \quad \text{since } \|d\|^2 \le \Delta\theta(x^0)\\ &\le K_2,\end{aligned}$$

where $K_2 := (1+n\Delta((L+1)^2+1))\theta(x^0)$. This last inequality holds since $\delta \le 1$. Returning to (14),

$$\begin{aligned}\theta(x+\tau d) &\le \phi_x(0) + \tau\phi_x'(0;d) + \tau\delta K_1 + \tau^2 K_2\\ &= \theta(x) + \tau\theta'(x;d) + \tau\delta K_1 + \tau^2 K_2.\end{aligned}$$

By Item 2 of Proposition 3.1,

$$\begin{aligned}\theta'(x;d) &\le \phi_x(d) - \phi_x(0) - z(x,d)\\ &\le (1-\gamma)\theta(x) - \theta(x) - z(x,d), \quad \text{by the test in Step 2}\\ &= -\gamma\theta(x) - z(x,d).\end{aligned}$$

Thus,

$$\theta(x+\tau d) - \theta(x) \le \tau\left(-\gamma\theta(x) - z(x,d)\right) + \tau\delta K_1 + \tau^2 K_2.$$

Note that the definitions of K_1 and K_2 are independent of δ. We can therefore consider a particular choice of δ: let $\delta := \min(1, \gamma\theta(x^*)/(2K_1))$ and let $\bar{\tau} := \min(\hat{\tau}(\delta), \gamma\theta(x^*)/(2K_2))$. Note that $\delta > 0$ and $\bar{\tau} > 0$, since $\theta(x^*) > 0$. It follows that for all $\tau \le \bar{\tau}$, and for k sufficiently large,

$$\begin{aligned}\theta(x+\tau d) - \theta(x) &\le -\tau z(x,d) - \tau\gamma\theta(x) + \tau\gamma\theta(x^*)/2 + \tau\bar{\tau}K_2\\ &\le -\tau z(x,d) - \tau\gamma\theta(x) + \tau\gamma\theta(x)/2 + \tau\gamma\theta(x)/2, \quad \text{since } \theta(x^*) \le \theta(x)\\ &= -\tau z(x,d)\\ &\le -\sigma\tau z(x,d), \quad \forall\ \sigma \le 1.\end{aligned} \tag{15}$$

Observe that the steplength ρ^m generated by Step 3 of the algorithm is chosen such that m is the *smallest* integer satisfying (7). Thus, $\tau := \rho^{m-1}$ cannot satisfy (15). But this means that

$$\rho^{m-1} \ge \bar{\tau}, \text{ which implies } \rho^m \ge \rho\bar{\tau}.$$

It follows by the linesearch test (7) and Lemma 3.10 that

$$\theta(x+\rho^m d) \le \theta(x) - \sigma\rho\bar{\tau}z(x,d) \le (1 - \frac{\sigma\rho\bar{\tau}\gamma^2}{2})\theta(x).$$

By setting $\hat{\rho} := 1 - \sigma\rho\bar{\tau}\gamma^2/2$, the proof is complete. □

Theorem 3.14 *If $\gamma \in (0,1)$ and $\Delta < \infty$, then the modified NE/SQP algorithm will terminate in a finite number of iterations provided that f is continuously differentiable on $\mathbb{B}$.*

Proof Let *tol* > 0 be the stopping tolerance used in the algorithm. If the algorithm does not terminate, then by Lemma 3.13, there exists $\hat{\rho} \in (0,1)$ such that for k sufficiently large, $\theta(x^{k+1}) \le \hat{\rho}\theta(x^k)$. Thus, after sufficiently many iterations, $\theta(x^k) <$ *tol*, and the algorithm will terminate in Step 4. □

4 The QPCOMP Algorithm

The basic idea behind QPCOMP is simple. The algorithm first tries to solve the problem using the modified NE/SQP algorithm. If this fails, QPCOMP then solves a sequence of perturbed problems in order to find a point with an improved value of the merit function. Once this point is found, QPCOMP returns to running the modified NE/SQP algorithm on the original problem, starting from this improved point.

One complication of the algorithm is that the subproblems must be solved inexactly in order to guarantee that they are each completed in a finite amount of time. To handle this we have introduced a sequence of tolerances $\{\eta_j\}$ which control the accuracy demanded by each subproblem.

Another complication is that the best choices of the parameters λ and γ cannot be known in advance. We now state the algorithm, including a description of how these parameters are adaptively chosen.

Algorithm QPCOMP

Step 1 [Initialization] Given a starting vector $x^0 \in \mathbb{B}$ and a convergence tolerance $\epsilon > 0$, choose $\delta > 0$, $\mu \in (0,1)$, $\gamma \in (0,1)$, $\nu \in (0,1)$, and set $k = 0$.

Step 2 [Attempt NE/SQP] Run the Modified NE/SQP algorithm with starting point x^k, with $tol = \epsilon$, to generate a point $\tilde{x}$.

Step 3 [Termination check] If $\tilde{x}$ solves MCP$(f, \mathbb{B})$, stop; otherwise continue with step 4.

Step 4 [Generate better starting point] Set $\theta_{best} := \theta(\tilde{x})$, set $y^0 = \tilde{x}$, set $j = 0$, and choose $\lambda > 0$, and choose a positive sequence $\{\eta_j\} \downarrow 0$.

Step 4a Run the Modified NE/SQP algorithm to solve the perturbed problem MCP$(f^{\lambda,y^j}, \mathbb{B})$ from starting point y^j, with $tol = \eta_j/(1+\|y^j\|)$. This generates a point $\tilde{y}$.

Step 4b If $\tilde{y}$ fails to solve the perturbed problem to the requested accuracy, set $\lambda \geq \lambda + \delta$ and $\gamma \leq \nu\gamma$, and goto step 4a; otherwise, continue.

Step 4c [Check point] If $\theta(\tilde{y}) \leq \mu\theta_{best}$, set $x^{k+1} = \tilde{y}$ and return to step 2, with k replaced by $k+1$. Otherwise, set $y^{j+1} := \tilde{y}$ and return to step 4a, with j replaced by $j+1$.

Observe, that the QPCOMP algorithm has the same local convergence properties as NE/SQP. In particular, by Theorem 3.9, for any b-regular solution x^*, there is a neighborhood such that the modified NE/SQP algorithm is identical to NE/SQP within this neighborhood. Thus, in Step 2 of the QPCOMP algorithm, if x^k is sufficiently close to x^*, then the modified NE/SQP algorithm will converge to x^* at the rates specified by Theorem 3.4.

We now establish global convergence properties for the algorithm:

Theorem 4.1 *If f is Lipschitz continuous and continuously differentiable on $\mathbb{B}$, and if MCP$(f, \mathbb{B})$ satisfies Assumption 2.2, then for any $\epsilon > 0$ the QPCOMP algorithm generates an iterate x^k satisfying $\theta(x^k) < \epsilon$ in a finite number of iterations.*

The remainder of this section is devoted to proving this theorem. As an introduction to the proof, note that if Step 4 is always successful at generating an improved starting point, then even if the Modified NE/SQP always fails in Step 2, the merit function values $\{\theta(x^k)\}$ will converge to 0 at least linearly, since $\theta(x^{k+1}) \leq \mu\theta(x^k)$ for all k. Thus, our convergence analysis is reduced to proving that Step 4 always generates an improved starting point.

In the analysis that follows, it will be convenient to define perturbed index sets by

$$\begin{array}{rcl} I_l^{\lambda,\bar{x}}(x) & := & \{i : x_i - l_i < f_i^{\lambda,\bar{x}}(x)\} \\ I_{el}^{\lambda,\bar{x}}(x) & := & \{i : x_i - l_i = f_i^{\lambda,\bar{x}}(x)\} \\ I_f^{\lambda,\bar{x}}(x) & := & \{i : x_i - u_i < f_i^{\lambda,\bar{x}}(x) < x_i - l_i\} \end{array} \qquad \begin{array}{rcl} I_{eu}^{\lambda,\bar{x}}(x) & := & \{i : x_i - u_i = f_i^{\lambda,\bar{x}}(x)\} \\ I_u^{\lambda,\bar{x}}(x) & := & \{i : x_i - u_i > f_i^{\lambda,\bar{x}}(x)\}. \end{array}$$

We shall also use the following obvious perturbations of the functions H, θ, ϕ, and z:

$$\begin{array}{l} H^{\lambda,\bar{x}}(x) := \min(x_i - l_i, \max(x_i - u_i, f_i^{\lambda,\bar{x}}(x))), \\ \theta^{\lambda,\bar{x}}(x) := \frac{1}{2}\left\|H^{\lambda,\bar{x}}(x)\right\|^2, \\ \phi_x^{\lambda,\bar{x}}(d) := \phi^{\lambda,\bar{x}}(x,d) := \sum \phi_i^{\lambda,\bar{x}}(x,d), \text{ where} \\ \qquad \phi_i^{\lambda,\bar{x}}(x,d) := \left\{ \begin{array}{ll} \frac{1}{2}(x_i - l_i + d_i)^2 & i \in I_l^{\lambda,\bar{x}}(x) \bigcup I_{el}^{\lambda,\bar{x}}(x) \\ \frac{1}{2}(x_i - u_i + d_i)^2 & i \in I_u^{\lambda,\bar{x}}(x) \bigcup I_{eu}^{\lambda,\bar{x}}(x) \\ \frac{1}{2}(f_i^{\lambda,\bar{x}}(x) + \nabla f_i^{\lambda,\bar{x}}(x)^\top d)^2 & i \in I_f^{\lambda,\bar{x}}(x) \end{array} \right. \qquad i = 1, \ldots, n. \\ z_x^{\lambda,\bar{x}}(d) := z^{\lambda,\bar{x}}(x,d) := \sum z^{\lambda,\bar{x}}(x,d), \text{ where} \\ \qquad z_i^{\lambda,\bar{x}}(x,d) := \left\{ \begin{array}{ll} \frac{1}{2}d_i^2 & i \notin I_f^{\lambda,\bar{x}}(x) \\ \frac{1}{2}(\nabla f_i^{\lambda,\bar{x}}(x)^\top d)^2 & i \in I_f^{\lambda,\bar{x}}(x) \end{array} \right. \qquad i = 1, \ldots, n. \end{array}$$

To show that Step 4 is always successful at generating an improved starting point, we begin by assuming that the Modified NE/SQP algorithm in Step 4a of QPCOMP fails at most a finite number of times. Later, we will remove this assumption. It follows that after a finite number of iterations, $\tilde{y}$ always solves the perturbed problem to the desired accuracy, so the algorithm always continues past Step 4b to Step 4c. Thus, either an improved point will eventually be found, or the algorithm will generate a sequence of iterates $\{y^j\}$ such that

$$\left\|H^{\lambda,y^j}(y^{j+1})\right\| \leq \frac{\eta_j}{1 + \|y^j\|}.$$

We then use the fact that $\{\eta_j\}$ converges to 0 to show that $\theta(y^j) \to 0$. This result is proved in the following lemma:

Lemma 4.2 *Let f be a Lipschitz continuous function and let $\{\eta_k\}$ be a sequence of positive numbers that converges to 0. Let $\lambda > 0$ and let $\{x^k\}$ be a sequence of points in $\mathbb{B}$ such that*

$$\left\|H^{\lambda,x^k}(x^{k+1})\right\| \leq \frac{\eta_k}{1 + \|x^k\|}, \qquad \forall k. \tag{16}$$

Suppose MCP($f, \mathbb{B}$) satisfies Assumption 2.2, then for any $\epsilon > 0$, there exists an iterate $x^j \in \{x^k\}$ such that $\theta(x^j) \leq \epsilon$.

Proof Let x^* be the solution to MCP($f, \mathbb{B}$) guaranteed by Assumption 2.2 which satisfies (1), and let $y^k := H^{\lambda,x^k}(x^{k+1})$. In the same spirit as the proof to Theorem 2.3, we establish a lower bound on the term $(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1})$.

Case 1: $y_i^k = x_i^{k+1} - l_i$ and $x_i^* < x_i^{k+1}$. Observe that

$$(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1}) = (x_i^{k+1} - x_i^*)\frac{f_i(x^{k+1}) - y_i}{\lambda} + w_i^k, \tag{17}$$

where

$$w_i^k := (x_i^{k+1} - x_i^*)\left(x_i^k - x_i^{k+1} - \frac{f_i(x^{k+1}) - y_i}{\lambda}\right).$$

Now, $0 < (x_i^{k+1} - x_i^*) \le x_i^{k+1} - l_i = y_i^k$. Also, $x_i^k - x_i^{k+1} \ge l_i - x_i^{k+1} = -y_i^k$. Thus,

$$\begin{aligned} w_i^k &= (x_i^{k+1} - x_i^*)\left(x_i^k - x_i^{k+1} + y_i/\lambda\right) - (x_i^{k+1} - x_i^*)(f_i(x^{k+1})/\lambda) \\ &\ge y_i^k\left(-y_i^k + y_i^k/\lambda\right) - |y_i^k|\left|f_i(x^{k+1})\right|/\lambda \\ &\ge -(y_i^k)^2 - |y_i^k|\left|f_i(x^{k+1})\right|/\lambda. \end{aligned}$$

Returning to (17), we get

$$(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1}) \quad \ge \quad (x_i^{k+1} - x_i^*)\left(f_i(x^{k+1}) - y_i\right)/\lambda - (y_i^k)^2 - \tfrac{|y_i^k|}{\lambda}\left|f_i(x^{k+1})\right|. \tag{18}$$

Case 2: $y_i^k = x_i^{k+1} - l_i$, and $x_i^* \ge x_i^{k+1}$. In this case, $f_i^{\lambda,x^k}(x^{k+1}) \ge x_i^{k+1} - l_i = y_i^k$. Thus, $f_i(x^{k+1}) + \lambda(x_i^{k+1} - x_i^k) \ge y_i^k$, so $x_i^k - x_i^{k+1} \le (f_i(x^{k+1}) - y_i^k)/\lambda$. Since $x_i^{k+1} - x_i^* \le 0$, we get

$$(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1}) \ge (x_i^{k+1} - x_i^*)\left(f_i(x^{k+1}) - y_i\right)/\lambda. \tag{19}$$

Case 3: $y_i^k = f_i^{\lambda,x^k}(x^{k+1})$. In this case, $y_i^k = f_i(x^{k+1}) + \lambda(x_i^{k+1} - x_i^k)$, so $x_i^k - x_i^{k+1} = (f_i(x^{k+1}) - y_i^k)/\lambda$. Thus,

$$(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1}) = (x_i^{k+1} - x_i^*)\left(f_i(x^{k+1}) - y_i^k\right)/\lambda.$$

Case 4: $y_i^k = x_i^{k+1} - u_i, x_i^{k+1} \ge x_i^*$. By similar arguments to Case 2, inequality (19) is satisfied.

Case 5: $y_i^k = x_i^{k+1} - u_i, x_i^{k+1} < x_i^*$. By similar arguments to Case 1, inequality (18) is satisfied.

In every case above, inequality (18) holds. Thus,

$$\begin{aligned} (x_i^k - x_i^*)^2 &= (x_i^{k+1} - x_i^* + x_i^k - x_i^{k+1})^2 \\ &= (x_i^{k+1} - x_i^*)^2 + 2(x_i^{k+1} - x_i^*)(x_i^k - x_i^{k+1}) + (x_i^k - x_i^{k+1})^2 \\ &\ge (x_i^{k+1} - x_i^*)^2 + 2(x_i^{k+1} - x_i^*)\left(f_i(x^{k+1}) - y_i^k\right)/\lambda - 2(y_i^k)^2 \\ &\quad -\tfrac{2}{\lambda}|y_i^k|\left|f_i(x^{k+1})\right| + (x_i^k - x_i^{k+1})^2, \qquad \text{by (18)}. \end{aligned}$$

Summing over all components, we get

$$\begin{aligned} \left\|x^k - x^*\right\|^2 \quad \ge \quad & \left\|x^{k+1} - x^*\right\|^2 + 2f(x^{k+1})^\top\left(x^{k+1} - x^*\right)/\lambda - 2\left(y^k\right)^\top\left(x^{k+1} - x^*\right)/\lambda \\ & -2\left\|y^k\right\|^2 - 2n\left\|y^k\right\|\left\|f(x^{k+1})\right\|/\lambda + \left\|x^k - x^{k+1}\right\|^2. \end{aligned}$$

Now, let L be the Lipschitz constant for f. Then $\left\|f(x^{k+1})\right\| \leq \left\|f(x^{k+1}) - f(x^*)\right\| + \|f(x^*)\| \leq L\left\|x^{k+1} - x^*\right\| + \|f(x^*)\|$. Further, by Assumption 2.2, $f(x^{k+1})^\top\left(x^{k+1} - x^*\right) \geq 0$. Thus,

$$\begin{aligned}
\left\|x^k - x^*\right\|^2 &\geq \left\|x^{k+1} - x^*\right\|^2 - 2\left\|y^k\right\|\left\|x^{k+1} - x^*\right\|/\lambda - 2\left\|y^k\right\|^2 \\
&\quad -2n\left\|y^k\right\|\left(L\left\|x^{k+1} - x^*\right\| + \|f(x^*)\|\right)/\lambda + \left\|x^k - x^{k+1}\right\|^2 \\
&\geq \left\|x^{k+1} - x^*\right\|^2 - 2\tfrac{\eta_k}{\lambda}\left\|x^{k+1} - x^*\right\|/\left(1 + \left\|x^k\right\|\right) - 2\eta_k^2/\left(1 + \left\|x^k\right\|\right)^2 \quad \text{by (16)} \\
&\quad -2n\tfrac{\eta_k}{\lambda}\left(L\left\|x^{k+1} - x^*\right\| + \|f(x^*)\|\right)/\left(1 + \left\|x^k\right\|\right) + \left\|x^k - x^{k+1}\right\|^2 \\
&= \left\|x^{k+1} - x^*\right\|^2 + \left\|x^k - x^{k+1}\right\|^2 - 2\eta_k\beta_k,
\end{aligned} \tag{20}$$

where

$$\beta_k := \frac{\left\|x^{k+1} - x^*\right\|}{\lambda(1 + \|x^k\|)} + \frac{\eta_k}{(1 + \|x^k\|)^2} + \frac{n\left(L\left\|x^{k+1} - x^*\right\| + \|f(x^*)\|\right)}{\lambda(1 + \|x^k\|)}.$$

Note that

$$\beta_k \leq \left\|x^{k+1} - x^*\right\|(nL + 1)/\lambda + \eta_0 + n\|f(x^*)\|/\lambda. \tag{21}$$

Let $C := (nL + 1)/\lambda + \eta_0 + n\|f(x^*)\|/\lambda$. Then $\beta_k \geq C$ implies that $\left\|x^{k+1} - x^*\right\| \geq 1$. Now, let $\{\beta_k : k \in \kappa\}$ be the subsequence of $\{\beta_k\}$ for which $\beta_k \geq C, \forall k \in \kappa$. It follows then that $\left\|x^{k+1} - x^*\right\| \geq 1, \forall k \in \kappa$. If we divide each side of (21) by $\left\|x^{k+1} - x^*\right\|^2$, it is easily seen that $\{\beta_k/\left\|x^{k+1} - x^*\right\|^2 : k \in \kappa\}$ is bounded.

However, dividing (20) by $\left\|x^{k+1} - x^*\right\|^2$ gives

$$\frac{\left\|x^k - x^*\right\|^2}{\|x^{k+1} - x^*\|^2} \geq 1 + \frac{\left\|x^k - x^{k+1}\right\|^2}{\|x^{k+1} - x^*\|^2} - \frac{2\eta_k\beta_k}{\|x^{k+1} - x^*\|^2}.$$

Since $\eta_k \downarrow 0$, the last term above converges to 0. Thus, for $k \in \kappa$ large enough,

$$\frac{\left\|x^k - x^*\right\|}{\|x^{k+1} - x^*\|} > \frac{1}{2},$$

and

$$\beta_k \leq \frac{2\left\|x^k - x^*\right\|}{\lambda(1 + \|x^k\|)} + \frac{\eta_k}{(1 + \|x^k\|)^2} + \frac{n\left(2L\left\|x^k - x^*\right\| + \|f(x^*)\|\right)}{\lambda(1 + \|x^k\|)}.$$

Observe that

$$\frac{\left\|x^k - x^*\right\|}{(1 + \|x^k\|)} \leq \frac{\|x^*\| + \left\|x^k\right\|}{(1 + \|x^k\|)} \leq \|x^*\| + 1.$$

Thus, the subsequence $\{\beta_k : k \in \kappa\}$ is bounded, from which it follows that $\{\beta_k\}$ is bounded.

Now, assume the lemma is false. Then there exists an $\epsilon > 0$ such that for all k, $\theta(x^k) > \epsilon^2/2$, which implies $\left\|H(x^k)\right\| > \epsilon$. Furthermore, for k large enough, $\eta_k < \epsilon^2$. Without loss of generality, we can assume that this inequality holds for all k.

Since f is Lipschitz continuous, H^{λ,x^k} is also Lipschitz continuous with some Lipschitz constant K. But then,

$$\begin{aligned}
\epsilon - \epsilon^2 &< \left\|H(x^k)\right\| - \eta_k \\
&\leq \left\|H^{\lambda,x^k}(x^k)\right\| - \left\|H^{\lambda,x^k}(x^{k+1})\right\| \left(1 + \left\|x^k\right\|\right) \\
&\leq \left\|H^{\lambda,x^k}(x^k) - H^{\lambda,x^k}(x^{k+1})\right\| \\
&\leq K \left\|x^{k+1} - x^k\right\|
\end{aligned}$$

Thus, for ϵ small enough,

$$\epsilon/(2K) < (\epsilon - \epsilon^2)/K < \left\|x^{k+1} - x^k\right\|.$$

Finally, since the sequence $\{\eta_k \beta_k\}$ converges to 0, then for all k sufficiently large, $\eta_k\beta_k < \epsilon/(8K)$. Thus, from (20)

$$\begin{aligned}
\left\|x^k - x^*\right\|^2 &\geq \left\|x^{k+1} - x^*\right\|^2 + \left\|x^{k+1} - x^k\right\|^2 - 2\eta_k\beta_k \\
&\geq \left\|x^{k+1} - x^*\right\|^2 + \epsilon/(2K) - \epsilon/(4K) \\
&= \left\|x^{k+1} - x^*\right\|^2 + \epsilon/(4K).
\end{aligned}$$

But, then

$$\left\|x^k - x^*\right\|^2 \geq \sum_{k+1}^{\infty} \epsilon/(4K) = \infty > \left\|x^k - x^*\right\|^2.$$

The lemma is thus proved by contradiction. $\square$

Note that Lemma 4.2 did not make any assumption on the choice of λ other than that it is greater than 0. Thus, even if λ is smaller than the Lipschitz constant, we can guarantee convergence.

The next stage in our analysis is to prove that the Modified NE/SQP algorithm can fail at most a finite number of times in Step 4a of QPCOMP. This is accomplished by observing that after each failure, the value of λ is increased, while the value of γ is decreased. Thus, the result will be proved if we can show that for λ large enough, and γ small enough, the Modified NE/SQP algorithm will always solve the perturbed problem MCP(f^{λ,y^j}, $\mathbb{B}$). This is accomplished by the following two lemmas.

Lemma 4.3 ([1], Lemma 2.3.3) *Suppose f is Lipschitz continuous with Lipschitz constant L, and let x and $\bar{x}$ be arbitrary points in $\mathbb{B}$. If $\lambda > 2L + 2$, and if $\bar{d}$ satisfies $\phi_x^{\lambda,\bar{x}}(\bar{d}) \leq \phi_x^{\lambda,\bar{x}}(0)$, then*

$$\left\|\bar{d}\right\|^2 < 11\, \theta^{\lambda,\bar{x}}(x).$$

Lemma 4.4 *Suppose f is Lipschitz continuous. There exist constants $\bar{\gamma} > 0$, and $\bar{\lambda} \geq 0$, such that for any $\lambda \geq \bar{\lambda}$, the modified NE/SQP algorithm applied to MCP($f^{\lambda,\bar{x}}$) will not terminate in Step 2 for any $\gamma \leq \bar{\gamma}$ and $\bar{x} \in \mathbb{B}$.*

Proof Suppose the lemma is false. Then there must exist a sequence $\{\lambda_j, \gamma_j\}$, with $\lambda \to \infty$ and $\gamma \downarrow 0$ such that for each j there exists a perturbed problem MCP($f^{\lambda_j,\bar{x}^j}$, $\mathbb{B}$) where the modified NE/SQP algorithm with $\gamma := \gamma_j$ fails in Step 2 when run on MCP($f^{\lambda_j,\bar{x}^j}$, $\mathbb{B}$)}.

Define $f^j(x)$, $H^j(x)$, $\theta_j(x)$, and $\phi_j(x,d)$, to be the f, H, θ, and ϕ functions corresponding to the jth perturbed problem. For example $f^j(x) := f^{\lambda_j,\bar{x}^j}(x)$, etc. Then for the jth problem to fail in Step 2, there must exist a point x^j and a direction d^j such that d^j is an optimal solution to the quadratic program (QP$_j$) defined by

$$\min_{x^j + d \in \mathbb{B}} \phi_j(x^j, d)$$

and also d^j fails one of the two tests in Step 2 of the algorithm. Without loss of generality, we can assume $\lambda_j \geq 2L+2, \forall j$. By Lemma 4.3, $\left\|d^j\right\|^2 < 11\,\theta_j(x^j) \leq \Delta\theta_j(x^j)$. Thus, the failure must occur because of the first test in Step 2. In other words,

$$\phi_j(x^j, d^j) \geq (1-\gamma_j)\theta_j(x^j), \quad \forall j. \tag{22}$$

Since $\phi_j(x^j, d^j) \leq \phi_j(x^j, 0) = \theta_j(x^j)$, and also, $\gamma_j \downarrow 0$, we see that

$$\lim \frac{\phi_j(x^j, d^j)}{\theta_j(x^j)} = 1. \tag{23}$$

Let $I_j := I_f^{\lambda_j, \bar{x}^j}(x^j)$, J_j be the set of indices not in I_j,

$$A_j := \frac{\left\|H^j_{I_j}(x^j)\right\|}{\|H^j(x^j)\|} \quad \text{and} \quad B_j := \frac{\left\|H^j_{J_j}(x^j)\right\|}{\|H^j(x^j)\|}.$$

We first show that $\lim_{j\to\infty} A_j = 0$. To do this, we examine a particular choice of j. Let $H^j := H^j(x^j)$. We can then rewrite $\phi_j(x^j, d)$, as follows:

$$\phi_j(x^j, d) := \frac{1}{2}\left\|(M^j + D^j)d + H^j\right\|^2$$

where

$$M^j_{i,\cdot} := \begin{cases} \nabla f_i^j(x^j)^\top & \text{if } i \in I_j \\ 0 & \text{if } i \in J_j. \end{cases} \qquad D^j_{ii} := \begin{cases} \lambda & \text{if } i \in I_j \\ 1 & \text{if } i \in J_j. \end{cases}$$

Observe that $x_i^j - u_i \leq H_i^j \leq x_i^j - l_i$. Note that for $\tilde{d}$ defined by

$$\tilde{d}_i := \begin{cases} -H_i^j/\lambda & \text{if } i \in I_j \\ 0 & \text{if } i \in J_j \end{cases}$$

it follows that $x^j + \tilde{d} \in \mathbb{B}$, since $\lambda \geq 1$. Furthermore

$$(M^j + D^j)\tilde{d} + H^j = \begin{bmatrix} (M^j\tilde{d})_{I_j} \\ H^j_{J_j} \end{bmatrix}.$$

Now, since d^j is an optimal solution to (QP$_j$),

$$\begin{aligned} \phi_j(x^j, d^j) \leq \phi_j(x^j, \tilde{d}) &= \tfrac{1}{2}\left\|(M^j + D^j)\tilde{d} + H^j\right\|^2 = \tfrac{1}{2}\left(\left\|(M^j\tilde{d})_{I_j}\right\|^2 + \left\|H^j_{J_j}\right\|^2\right) \\ &\leq \tfrac{1}{2}\left(L^2\left\|\tilde{d}\right\|^2 + \left\|H^j_{J_j}\right\|^2\right) \leq \tfrac{1}{2}\left\|H^j\right\|^2\left(L^2 {A_j}^2/{\lambda_j}^2 + {B_j}^2\right). \end{aligned}$$

Thus, by (23),

$$1 = \lim \frac{\phi_j(x^j, d^j)}{\theta_j(x^j)} \leq \liminf\left(\frac{L^2 A_j^2}{\lambda_j^2} + B_j^2\right).$$

But, since $\{A_j\}$ is bounded, and $\lambda_j \to \infty$, we see that $1 \leq \liminf B_j^2$. Furthermore, $B_j \leq 1$, so $\lim B_j = 1$, which implies that $A_j \to 0$.

Let us now examine the direction finding subproblem (QP_j) for large j. For some $\alpha \in [0,1]$, define $\tilde{d}$ by

$$\tilde{d}_i := \begin{cases} 0 & \text{if } i \in I_j \\ -\alpha H_i^j & \text{if } i \in J_j. \end{cases}$$

Here we see that

$$(M^j + D^j)\tilde{d} + H^j = \begin{bmatrix} H_{I_j}^j + (M^j \tilde{d})_{I_j} \\ (1-\alpha) H_{J_j}^j \end{bmatrix}.$$

Thus,

$$\begin{aligned}
\phi_j(x^j, d^j) &\leq \tfrac{1}{2} \left\| (M^j + D^j)\tilde{d} + H^j \right\|^2 \\
&= \tfrac{1}{2} \left(\left\| H_{I_j}^j + (M^j \tilde{d})_{I_j} \right\|^2 + (1-\alpha)^2 \left\| H_{J_j}^j \right\|^2 \right) \\
&\leq \tfrac{1}{2} \left(\left\| H_{I_j}^j \right\|^2 + 2 \left\| H_{I_j}^j \right\| \left\| M^j \tilde{d} \right\| + \left\| M^j \tilde{d} \right\|^2 + (1-\alpha)^2 \left\| H_{J_j}^j \right\|^2 \right) \\
&\leq \tfrac{1}{2} \left((A_j \left\| H^j \right\|)^2 + 2 A_j \left\| H^j \right\| L \left\| \tilde{d} \right\| + L^2 \left\| \tilde{d} \right\|^2 + ((1-\alpha) B_j \left\| H^j \right\|)^2 \right) \\
&\leq \tfrac{1}{2} \left\| H^j \right\|^2 \left(A_j^2 + 2 A_j \alpha L + \alpha^2 L^2 + (1-\alpha)^2 B_j^2 \right) \\
&\leq \theta(x^j) \left(A_j^2 + 2 A_j \alpha L + (1 - 2\alpha + (L^2+1)\alpha^2) \right), \quad \text{since } B_j \leq 1.
\end{aligned}$$

Choosing $\alpha = 1/(1+L^2)$, we get $\phi_j(x^j, d^j) \leq \theta(x^j) \left(A_j (2L/(1+L^2) + A_j) + 1 - 1/(1+L^2) \right)$. But since $\lim A_j = 0$, we see that

$$\limsup \frac{\phi_j(x^j, d^j)}{\theta(x^j)} \leq 1 - \frac{1}{1+L^2} < 1,$$

contradicting (23). Thus, the lemma is proved by contradiction. □

We can now combine the results of the previous three lemmas to prove that Step 4 always generates an improved starting point.

Lemma 4.5 *Suppose that f is Lipschitz continuous and continuously differentiable on $\mathbb{B}$ and that MCP($f, \mathbb{B}$) satisfies Assumption 2.2. If the QPCOMP algorithm fails to terminate, it will execute Step 2 an infinite number of times.*

Proof Assume the lemma is false. It then follows that after a finite number of statements are executed, the algorithm never returns to Step 2. But this means that, thereafter, the test in Step 4c of the algorithm is never satisfied.

By Theorem 3.14, the modified NE/SQP algorithm will always terminate in a finite number of steps. Thus, Step 4b of the QPCOMP algorithm will be executed an infinite number of times. But the test in Step 4b can fail only a finite number of times. After that, λ will be large enough and γ will be small enough that by Lemma 4.4 the Modified NE/SQP algorithm will always find a solution to the perturbed problems. Thus we see that Step 4c is visited an infinite number of times, and moreover, after a finite number of iterations, the value of λ is fixed. But then Lemma 4.2 guarantees that the test in Step 4c will be satisfied after a finite number of iterations. But this contradicts our original assumption, so the lemma is true. □

We are now ready to prove Theorem 4.1

Proof (of Theorem 4.1)
By Lemma 4.5 either the algorithm will terminate with a solution in Step 3, or Step 2 will be executed an infinite number of times. But if Step 2 is executed an infinite number of times, then we have

$$\theta(x^{k+1}) < \mu\theta(x^k) \Longrightarrow \theta(x^k) < \mu^k\theta(x^0),$$

so $\theta(x^k)$ converges to zero. □

5 Implementation and Testing

The QPCOMP algorithm was coded in ANSI C, using double precision arithmetic. The Fortran package MINOS [12] was used to solve the quadratic subproblems. The algorithm allows for a great deal of flexibility in the choice of parameters, which can be specified in an options file. For testing purposes, we used the following choices of parameters in the QPCOMP and Modified NE/SQP algorithms: $\mu = .9$, $\Delta = 1.0e4$, $\rho = .5$, $\sigma = .5$. The sequence $\{\eta_j\}$ used in Step 4 of the QPCOMP algorithm was given by $\eta_{j+1} = 0.999 * \eta_j$, with η_0 set to 1000. This effectively caused the Modified NE/SQP algorithm to perform only one iteration before returning control back to QPCOMP. The parameter λ was updated as follows:

1. In Step 4, λ is set to θ_{best}.

2. In Step 4b, if $\tilde{y}$ fails to solve the perturbed problem, λ is set to $\max(.1, 10\lambda)$; otherwise, it is multiplied by .9.

Finally, the parameter γ is initially chosen to be .01. Thereafter, in Step 4b, it is set to $\min(1/\lambda, \gamma)$. For practical considerations, we also placed a limit on the number of allowable iterations of the linesearch in Step 3 of the modified NE/SQP algorithm. This limit is set to 10 when the Modified NE/SQP algorithm is called from Step 2 of QPCOMP, and is increased by 4 whenever the Modified NE/SQP algorithm fails, up to a maximum of 30.

QPCOMP was interfaced with the GAMS modeling language [2, 6], allowing problems to be easily specified in GAMS, and the algorithm to be tested using MCPLIB [4] and GAMSLIB [2]. Specifically, we tested QPCOMP on every problem with fewer than 110 variables in MCPLIB and GAMSLIB. Larger problems were excluded because our implementation of QPCOMP uses a dense solver for the QP subproblems. Table 2 summarizes the features of the problems tested.

We also tested NE/SQP, PATH version 2.8 [5], and SMOOTH version 3.0 [3] on the problems in Table 2. To run NE/SQP, we simply used the QPCOMP algorithm with $\Delta = \infty$ and $\gamma = 0$. A comparison of the performance of the algorithms is given in Table 3. Many of the problems in the library are specified with more than one starting point. The particular starting point used is shown in the second column of the table. For each problem we report the execution time (in seconds) and the number of function and Jacobian evaluation, f and J. To save space, we have omitted from this table any problems which all four algorithms solved in less than a second. All of the problems were solved to an accuracy of 10^{-6}. Specifically, for QPCOMP the stopping criteria was $\|H(x)\| \leq 10^{-6}$.

The results of the testing demonstrate the high degree of robustness of the QPCOMP algorithm. We note that although it did not solve the Von Thünen problems, QPCOMP was able to solve these problems to an accuracy of 10^{-4}. Experimentation with the Von Thünen problems suggests that the Jacobian matrix is singular at the solution. Thus, near the solution, the Jacobian matrix is poorly conditioned. This ill-conditioning is exacerbated in QPCOMP by the fact that the QP subproblems are formulated using the square of the Jacobian matrix, resulting in extremely ill-conditioned QP

Table 2: Models

GAMS file	Model origin	Type	Size	Nonzeros
bertsekas.gms	Traffic assignment	NCP	15	75
billups.gms	Section 2	NCP	1	1
cafemge.gms	GAMSLIB (139)	MCP	47	316
choi.gms	Nash equil.	NCP	13	169
colvncp.gms	Colville #2	NLP	15	99
colvdual.gms	Colville #2 (Dual)	NLP	20	149
ehl_k60.gms	Lubrication	MCP	61	3721
ehl_k80.gms	"	MCP	81	6561
ehl_kost.gms	"	MCP	101	10201
freebert.gms	Traffic assignment	MCP	15	75
gafni.gms	"	MCP	5	25
hanskoop.gms	Capital stock	NCP	14	116
hansmcp.gms	GAMSLIB (135)	MCP	43	356
hansmge.gms	"(147)	MCP	43	793
harkmcp.gms	GAMSLIB (128)	MCP	32	103
harmge.gms	"(148)	MCP	9	81
hydroc06.gms	Distillation	NE	29	223
hydroc20.gms	"	NE	99	740
josephy.gms	MCPLIB	NCP	4	16
kehomge.gms	GAMSLIB (149)	MCP	9	81
kojshin.gms	MCPLIB	NCP	4	16
kormcp.gms	GAMSLIB (130)	MCP	78	346
mathi*.gms	Walrasian	NCP	4	11
methan08.gms	Distillation	NE	31	226
nash.gms	Nash equil.	MCP	10	100
oligomcp.gms	GAMSLIB (133)	MCP	6	16
pgvon105.gms	Von Thünen	NCP	105	795
pgvon106.gms	"	NCP	106	899
pies.gms	PIES model	MCP	42	184
powell.gms	Powell	NLP	16	188
powell_mcp.gms	"	NCP	8	47
sammge.gms	GAMSLIB (151)	MCP	14	170
scarfa*.gms	Walrasian	NCP	14	96
scarfb*.gms	"	NCP	40	575
scarfmge.gms	"	NCP	20	348
shovmge.gms	GAMSLIB (153)	MCP	10	100
sppe.gms	Spatial price	MCP	27	84
tobin.gms	"	MCP	42	202
transmcp.gms	GAMSLIB (126)	MCP	11	24
two3mcp.gms	GAMSLIB (131)	MCP	6	24
unstmge.gms	GAMSLIB (155)	MCP	5	25
vonthmge.gms	Von Thünen	MCP	80	842
wallmcp.gms	GAMSLIB (127)	MCP	6	20

Table 3: Performance Results

Problem Name	st. pt.	NE/SQP sec.	NE/SQP f(J)	PATH sec.	PATH f(J)	QPCOMP sec.	QPCOMP f(J)	SMOOTH sec.	SMOOTH f(J)
bertsekas	1	fail	fail	0.08	27(6)	2.83	151(44)	0.24	113(27)
bertsekas	2	fail	fail	0.04	5(5)	2.41	126(40)	0.05	7(7)
billups	1	fail	fail	fail	fail	0.11	23(22)	fail	fail
cafemge	1	18.16	16(10)	0.29	9(7)	20.11	16(10)	0.41	9(8)
cafemge	2	16.57	15(8)	0.26	6(6)	14.19	15(8)	0.25	6(6)
choi	1	2.00	5(4)	2.09	5(5)	2.28	5(4)	2.10	5(5)
colvdual	1	fail	fail	0.11	15(13)	5.76	252(78)	0.11	40(15)
colvdual	2	fail	fail	0.09	16(12)	5.39	184(59)	0.10	52(17)
ehl_k60	1	16.11	11(8)	1.56	6(6)	16.91	11(8)	1.59	6(6)
ehl_k60	2	fail	fail	25.16	84(66)	147.22	186(84)	14.71	106(34)
ehl_k60	3	fail	fail	44.97	99(50)	492.33	1030(98)	fail	fail
ehl_k80	1	fail	fail	2.37	6(6)	313.15	98(95)	2.93	6(6)
ehl_k80	2	fail	fail	131.99	541(44)	129.02	72(33)	6.57	24(12)
ehl_k80	3	435.77	442(86)	56.58	132(43)	729.89	556(135)	85.26	425(72)
ehl_kost	1	fail	fail	3.86	6(6)	611.41	108(105)	4.73	6(6)
ehl_kost	2	248.79	97(30)	13.56	19(19)	250.28	97(30)	12.58	21(12)
ehl_kost	3	fail	fail	9.76	11(11)	866.08	409(79)	90.38	262(55)
freebert	1	fail	fail	0.07	5(5)	2.72	151(44)	0.04	6(6)
freebert	3	fail	fail	0.05	5(5)	2.86	173(45)	0.04	6(6)
freebert	4	fail	fail	0.09	27(6)	2.47	151(44)	fail	fail
freebert	5	fail	fail	0.04	5(5)	1.38	116(23)	0.04	5(5)
freebert	6	fail	fail	0.08	27(6)	3.02	173(45)	fail	fail
hanskoop	5	fail	fail	0.09	19(11)	0.70	27(11)	0.30	102(34)
hanskoop	7	fail	fail	0.05	11(6)	0.86	45(13)	0.22	83(25)
hansmcp	1	fail	fail	0.47	45(18)	fail	fail	0.13	10(8)
hansmge	1	3.14	11(8)	0.36	12(8)	2.86	11(8)	0.64	26(13)
harkmcp	1	1.27	34(11)	0.05	8(8)	1.06	23(10)	0.07	10(8)
harkmcp	4	6.96	29(13)	0.12	13(6)	9.31	27(14)	0.37	31(17)
harmge	1	fail	fail	0.06	11(7)	1.86	132(57)	0.09	33(11)
harmge	2	fail	fail	0.03	5(5)	0.14	5(4)	0.03	5(5)
harmge	3	fail	fail	0.04	5(5)	0.13	5(4)	0.04	5(5)
harmge	4	fail	fail	0.05	8(6)	0.15	5(4)	0.04	8(6)
harmge	5	fail	fail	0.05	8(6)	0.16	8(5)	0.04	8(6)
harmge	6	fail	fail	0.06	13(8)	3.24	379(78)	2.08	1117(139)
hydroc20	1	16.11	10(8)	0.38	11(9)	13.31	10(8)	0.36	10(9)
josephy	1	fail	fail	0.03	7(7)	0.08	13(7)	0.03	24(9)
josephy	2	fail	fail	0.04	15(11)	0.07	15(7)	0.02	9(6)
josephy	4	fail	fail	0.02	4(4)	0.04	5(4)	0.02	5(4)
josephy	6	0.04	4(3)	fail	fail	0.05	12(6)	0.02	9(6)

Table 3: Performance Results (cont.)

Problem	st.	NE/SQP		PATH		QPCOMP		SMOOTH	
Name	pt.	sec.	f(J)	sec.	f(J)	sec.	f(J)	sec.	f(J)
kojshin	1	fail	fail	0.03	6(6)	0.07	16(7)	0.03	33(10)
kojshin	3	fail	fail	0.06	17(17)	0.12	35(10)	0.11	189(27)
kormcp	1	2.82	4(3)	0.08	4(4)	2.82	4(3)	0.05	4(4)
pgvon105	1	fail	fail	1.54	64(16)	fail	fail	fail	fail
pgvon105	2	41.51	199(39)	0.77	27(10)	50.91	213(30)	fail	fail
pgvon105	3	33.47	153(32)	1.58	63(14)	58.80	322(40)	fail	fail
pgvon105	4	fail	fail	fail	fail	fail	fail	fail	fail
pgvon106	1	fail	fail	19.77	772(101)	fail	fail	125.46	6428(482)
pgvon106	2	fail	fail	1.80	48(36)	fail	fail	5.37	109(37)
pgvon106	3	fail	fail	1.29	39(20)	fail	fail	8.48	233(49)
pgvon106	4	fail	fail	fail	fail	fail	fail	fail	fail
pgvon106	5	fail	fail	fail	fail	fail	fail	fail	fail
pgvon106	6	fail	fail	fail	fail	fail	fail	3.76	58(27)
pies	1	fail	fail	0.13	13(13)	7.26	54(49)	0.27	41(14)
sammge	1	fail	fail	0.01	1(1)	fail	fail	0.00	1(1)
sammge	10	fail	fail	0.01	1(1)	fail	fail	0.01	1(1)
scarfasum	2	fail	fail	0.04	5(5)	1.51	73(26)	0.10	23(6)
scarfbnum	1	6.27	70(20)	0.39	24(14)	6.42	76(21)	0.32	71(20)
scarfbnum	2	6.01	97(22)	0.44	25(15)	6.09	58(19)	0.32	95(24)
scarfbsum	1	fail	fail	fail	fail	8.77	26(22)	0.24	24(11)
scarfbsum	2	fail	fail	2.37	66(18)	31.11	157(83)	0.66	103(24)
scarfmge	4	1.02	17(12)	0.18	25(11)	0.97	17(12)	0.17	20(12)
shovmge	2	1.02	4(3)	0.09	4(4)	1.11	4(3)	0.10	4(4)
shovmge	4	1.19	10(4)	0.08	4(4)	1.96	20(4)	0.08	4(4)
tobin	1	1.33	15(10)	0.08	12(9)	1.49	15(10)	0.13	31(12)
tobin	2	1.83	18(11)	0.10	13(9)	1.78	18(11)	0.09	17(12)
transmcp	1	fail	fail	0.04	12(12)	1.22	69(67)	0.05	24(15)
transmcp	2	fail	fail	0.01	1(1)	fail	fail	0.00	1(1)
vonthmge	1	fail	fail	1.06	34(22)	fail	fail	17.14	730(278)

subproblems. The inability of QPCOMP to achieve higher accuracy on these problems appears to be a symptom of this difficulty.

The fact that QPCOMP is not as fast as PATH and SMOOTH is not surprising; our intent was to demonstrate the robustness of our approach. In contrast to the slow execution times of QPCOMP, note that the number of function and Jacobian evaluations required by the QPCOMP algorithm is often quite reasonable. This indicates that a much more efficient version of the code might be attainable by using a faster QP solver. It is important also to recognize that PATH and SMOOTH are finely tuned codes which include numerous enhancements that greatly improve their performance. For example both algorithms employ a projected Newton preprocessor, which although unreliable, often produces an approximate solution extremely quickly. In addition, version 2.8 of PATH uses a proximal perturbation heuristic that was motivated by the success of QPCOMP. In contrast, the version of QPCOMP we tested here is exactly the version for which we proved our convergence results.

6 Conclusions

In this paper, we have demonstrated that our strategy for solving a sequence of perturbed subproblems is very effective in enhancing the robustness of an algorithm. Our numerical results indicate that the NE/SQP algorithm is considerably less robust than either PATH, or SMOOTH. However, it is certainly capable of being used as a solver for perturbed problems that are strongly monotone. We were thus able to develop the QPCOMP algorithm which is theoretically more robust than any superlinearly or quadratically convergent algorithms currently available. The test results demonstrate a dramatic improvement in robustness over the NE/SQP algorithm.

There are several weaknesses in the NE/SQP solver which became evident in developing and testing QPCOMP. The first lies in the definition of the H function, which is fundamental to the calculation of the direction taken at each step. In our opinion, search directions for complementarity problems are best calculated by incorporating both function information and boundary information. However, the H function used by NE/SQP uses only one or the other at each iteration. The second weakness lies in the fact that a quadratic program is solved at each iteration. This is not only more expensive than solving a linear system, but also causes problems with ill-conditioning. While this approach was necessary in NE/SQP to ensure that the subproblems were always solvable, it is not required if a perturbation strategy is used, since any unsolvable subproblem can be handled by a simple perturbation. We are therefore anxious to try our perturbation strategy on more promising fundamental algorithms.

References

[1] S. C. Billups, *Algorithms for Complementarity Problems and Generalized Equations* (Ph.D. thesis, University of Wisconsin–Madison, Madison, Wisconsin, 1995).

[2] A. Brooke, D. Kendrick and A. Meeraus, *GAMS: A User's Guide* (The Scientific Press, South San Francisco, CA, 1988).

[3] C. Chen and O. L. Mangasarian, "A class of smoothing functions for nonlinear and mixed complementarity problems", *Computational Optimization and Applications* 5 (1996) 97–138.

[4] S. P. Dirkse and M. C. Ferris, "MCPLIB: A collection of nonlinear mixed complementarity problems", *Optimization Methods and Software* 5 (1995) 319–345.

[5] S. P. Dirkse and M. C. Ferris, "The PATH solver: A non-monotone stabilization scheme for mixed complementarity problems", *Optimization Methods and Software* 5 (1995) 123–156.

[6] S. P. Dirkse, M. C. Ferris, P. V. Preckel and T. Rutherford, "The GAMS callable program library for variational and complementarity solvers", Mathematical Programming Technical Report 94-07, Computer Sciences Department, University of Wisconsin (Madison, Wisconsin, 1994), Available from ftp://ftp.cs.wisc.edu/math-prog/tech-reports/.

[7] M. C. Ferris and J. S. Pang, "Engineering and economic applications of complementarity problems", Discussion Papers in Economics 95–4, Department of Economics, University of Colorado (Boulder, Colorado, 1995), Available from ftp://ftp.cs.wisc.edu/math-prog/tech-reports/.

[8] S. A. Gabriel, *Algorithms for the Nonlinear Complementarity Problem: The NE/SQP Method and Extensions* (Ph.D. thesis, The Johns Hopkins University, Baltimore, Maryland, 1992).

[9] P. T. Harker and J. S. Pang, "Finite–dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications", *Mathematical Programming* 48 (1990) 161–220.

[10] P. T. Harker and B. Xiao, "Newton's method for the nonlinear complementarity problem: A B–differentiable equation approach", *Mathematical Programming* 48 (1990) 339–358.

[11] O. L. Mangasarian, *Nonlinear Programming* (McGraw–Hill, New York, 1969), SIAM Classics in Applied Mathematics 10, SIAM, Philadelphia, 1994.

[12] B. A. Murtagh and M. A. Saunders, "MINOS 5.0 user's guide", Technical Report SOL 83.20, Stanford University (Stanford, California, 1983).

[13] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables* (Academic Press, San Diego, California, 1970).

[14] J. S. Pang and S. A. Gabriel, "NE/SQP: A robust algorithm for the nonlinear complementarity problem", *Mathematical Programming* 60 (1993) 295–338.

[15] D. Ralph, "Global convergence of damped Newton's method for nonsmooth equations, via the path search", *Mathematics of Operations Research* 19 (1994) 352–389.

[16] R. T. Rockafellar, "Monotone operators and augmented Lagrangian methods in nonlinear programming", in: O. L. Mangasarian, R. R. Meyer and S. M. Robinson eds., *Nonlinear Programming 3* (Academic Press, London, 1978) pp. 1–26.

[17] T. F. Rutherford, "MILES: A mixed inequality and nonlinear equation solver", Working Paper, Department of Economics, University of Colorado, Boulder.

[18] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill–Posed Problems* (John Wiley & Sons, New York, 1977).