数理解析研究所講究録
第 829 巻 1993 年 152-167

Optimization-Based Iterative Methods for Solving Nonlinear Complementarity Problems

TAJI Kouichi (田地宏一)
FUKUSHIMA Masao (福島雅夫)
IBARAKI Toshihide (茨木俊秀)

Department of Applied Mathematics and Physics,
Faculty of Engineering, Kyoto University, Kyoto 606, JAPAN

1 Introduction

We consider the nonlinear complementarity problem, which is to find a vector $x \in R^n$ such that

$$x \geq 0, F(x) \geq 0 \text{ and } x^T F(x) = 0, \tag{1}$$

where $F(x) = (F_1(x), F_2(x), \ldots, F_n(x))^T$ is a given continuously differentiable mapping from R^n into itself and ^{T} denotes transposition. This problem has been used to formulate and study various equilibrium problems including the traffic equilibrium problem, the spatial economic equilibrium problem and Nash equilibrium problem [4, 12, 16].

The nonlinear complementarity problem (1) is a special case of the variational inequality problem, which is to find a vector $x^* \in S$ such that

$$(x - x^*)^T F(x^*) \geq 0 \quad \text{for all} \quad x \in S, \tag{2}$$

where S is a nonempty closed convex set in R^n. Problem (1) corresponds to the case $S = R^n_+$, the nonnegative orthant of R^n. The variational inequality problem has also been used to formulate and study various equilibrium problems arising in economics, operations research, transportation and regional sciences.

To solve the nonlinear complementarity problem (1) and the variational inequality problem (2), various iterative algorithms, such as fixed point algorithms, projection methods, nonlinear Jacobi method, successive over-relaxation methods and Newton method, have been proposed [6, 7, 13]. Among these, fixed point algorithms originally have been used

in the proof that the problem has a solution. Fixed point algorithms are useless for practical computations because their convergence are extremely slow [14]. The other methods are generalizations of methods for systems of nonlinear equations and their convergence results have been obtained [7, 13]. But, in general, these methods do not have globally convergent property.

Assuming the monotonicity of mapping F, Fukushima [5] has recently proposed a differentiable optimization formulation for variational inequality problem (2) and proposed a decent algorithm to solve (2). Based on this optimization formulation, Taji et al. [15] proposed a modification of Newton method for solving the variational inequality problem (2), and proved that, under the strong monotonicity assumption, the method is globally and quadratically convergent.

In this paper we apply the methods of Fukushima [5] and Taji et al. [15] to the nonlinear complementarity problem. We show that those specialized methods can take full advantage of the special structure of problem (1), thereby yielding new globally convergent algorithms for solving monotone complementarity problems. We remark that the constraint set of nonlinear complementarity problem is R^n_+ which is clearly not compact, while the descent method by Fukushima [5] has assumed that the constraint set is compact. In this paper we show that the compactness assumption can be removed for the nonlinear complementarity problem. We also present some computational results to demonstrate that those methods are practically efficient.

2 Equivalent optimization problem

We first review the results obtained by Fukushima [5] for the general variational inequality problem (2). Let G be an $n \times n$ symmetric positive definite matrix. The projection under the G-norm of $x \in R^n$ onto the closed convex set S, denoted $\mathrm{Proj}_{S,G}(x)$, is defined as the unique solution of the following mathematical program:

$$\text{minimize} \quad \| y - x \|_G \quad \text{subject to} \quad y \in S,$$

where $\| \cdot \|_G$ denotes the G-norm in R^n, which is defined by $\| x \|_G = (x^T G x)^{\frac{1}{2}}$.

Using this notation, we define function $f : R^n \to R$ by

$$f(x) = \max\{-(y-x)^T F(x) - \frac{1}{2}(y-x)^T G(y-x) \mid y \in S\} \tag{3}$$

$$= -(H(x)-x)^T F(x) - \frac{1}{2}(H(x)-x)^T G(H(x)-x), \tag{4}$$

where the mapping $H : R^n \to R^n$ is defined by

$$H(x) = \mathrm{Proj}_{S,G}(x - G^{-1}F(x)). \tag{5}$$

It can be shown [5] that the function f is continuously differentiable whenever so is the mapping F, and its gradient is given by

$$\nabla f(x) = F(x) - [\nabla F(x) - G](H(x) - x). \tag{6}$$

The function f has the property that $f(x) \geq 0$ for all $x \in S$ and $f(x) = 0$ whenever x is a solution to the variational inequality problem (2). Hence problem (2) is equivalent to the following optimization problem:

$$\text{minimize } f(x) \text{ subject to } x \in S. \tag{7}$$

In addition, when $\nabla F(x)$ is positive definite for all $x \in S$, it can be shown that, if $x \in S$ is a stationary point of problem (7), i.e.,

$$(y - x)^T \nabla f(x) \geq 0 \text{ for all } y \in S, \tag{8}$$

then x is a global optimal solution of problem (7), and hence it solves the nonlinear complementarity problem (1).

Let us now specialize the above results to the complementarity problem. For general variational inequality problems, it may be expensive to evaluate the function f unless S is tractable. In the case of the complementarity problem (1), the set S turns out to be R^n_+. So, if we in particular let G be a diagonal matrix $D = diag(\delta_1, \ldots, \delta_n)$, where δ_i are positive constants, then the mapping H takes the explicit form

$$H(x) = \max(0, x - D^{-1}F(x)) \tag{9}$$

where maximum operator is taken component-wise, i.e.

$$H_i(x) = \max(0, x_i - \delta_i^{-1}F_i(x)), \quad i = 1, \ldots, n. \tag{10}$$

Hence the function f and its gradient can be represented as

$$\begin{aligned} f(x) &= \frac{1}{2}F(x)^T D^{-1}F(x) - \frac{1}{2}\max(0, D^{-1}F(x) - x)^T D \max(0, D^{-1}F(x) - x) \\ &= \sum_{i=1}^{n} \frac{1}{2\delta_i}\left\{F_i(x)^2 - (\max(0, F_i(x) - \delta_i x_i))^2\right\}, \end{aligned} \tag{11}$$

and

$$\nabla f(x) = \nabla F(x)D^{-1}F(x) + (I - \nabla F(x)D^{-1})\max(0, F(x) - Dx) \tag{12}$$

respectively, and the optimization problem (7) becomes

$$\text{minimize } f(x) \text{ subject to } x \geq 0. \tag{13}$$

In this special case, it is therefore straight forward to evaluate the function f and its gradient. Furthermore, the optimization problem (13) has a simple constraint.

3 Descent methods

In this section, we specialize the methods of Fukushima [5] and Taji et al. [15], which were originally proposed for variational inequality problems, to the nonlinear complementarity problem (1). Throughout this section, we let D be a positive definite diagonal matrix and the function f be defined by (11). We also suppose that the mapping F is strongly monotone on R^n_+ with modulus $\mu > 0$ i.e.,

$$(x-y)^T(F(x)-F(y)) \geq \mu \parallel x-y \parallel^2 \quad \text{for all} \quad x, y \geq 0. \tag{14}$$

The first method uses the vector

$$\begin{aligned} d &= H(x) - x \\ &= \max(0, x - D^{-1}F(x)) - x \end{aligned} \tag{15}$$

as a search direction at x. When the mapping F is strongly monotone with modulus μ, it is shown [5] that the vector d given by (15) satisfies the descent condition

$$d^T \nabla f(x) \leq -\mu \parallel d \parallel^2 .$$

Thus the direction d can be used to determine the next iterate by using the following Armijo-type line search rule: Let $\alpha := \beta^{\hat{m}}$, where $\hat{m}$ is the smallest nonnegative integer m such that

$$f(x) - f(x + \beta^m d) \geq \sigma \beta^m \parallel d \parallel^2,$$

where $0 < \beta < 1$ and $0 < \sigma$.

Algorithm 1a:

choose $x^0 \geq 0$, $\beta \in (0,1)$, $\sigma > 0$ and a positive diagonal matrix D;
$k := 0$
while convergence criterion is not satisfied **do**
  $d^k := \max(0, x^k - D^{-1}F(x^k)) - x^k$;
  $m := 0$
  while $f(x^k) - f(x^k + \beta^m d^k) < \sigma \beta^m \parallel d^k \parallel^2$ **do**
    $m := m + 1$
  endwhile
  $x^{k+1} := x^k + \beta^m d^k$;
  $k := k + 1$
endwhile

In line search procedure of this algorithm, we examine only the points shorter than unit step size. But we expect to decrease the value of the function f when the longer step size is chosen. Therefore we propose the algorithm in which we modify Algorithm 1a so that the longer step size can be selected.

Algorithm 1b:

choose $x^0 \geq 0$, $\beta_1 > 1$, $\beta_2 \in (0,1)$, $\sigma > 0$ and a positive diagonal matrix D;
$k := 0$
while convergence criterion is not satisfied **do**
  $d^k := \max(0, x^k - D^{-1}F(x^k)) - x^k$;
  $\hat{t} := \max\{t | x^k + td^k \geq 0,\ t \geq 0\}$;
  $m := 0$
  if $f(x^k) - f(x^k + d^k) \geq \sigma \parallel d^k \parallel^2$ **then**
    while $\beta_1^m \leq \hat{t}$ and $f(x^k) - f(x^k + \beta_1^m d^k) \geq \sigma \beta_1^m \parallel d^k \parallel^2$
      and $f(x^k + \beta_1^{m+1} d^k) \leq f(x^k + \beta_1^m d^k)$ **do**
      $m := m + 1$
    endwhile
    $x^{k+1} := x^k + \beta_1^m d^k$
  else
    while $f(x^k) - f(x^k + \beta_2^m d^k) < \sigma \beta_2^m \parallel d^k \parallel^2$ **do**
      $m := m + 1$
    endwhile
    $x^{k+1} := x^k + \beta_2^m d^k$
  endif
  $k := k + 1$
endwhile

Note that, since an evaluation of f at a given point x is equivalent to evaluating the vector $\max(0, x - D^{-1}F(x))$, the vector $H(x^k) = \max(0, x^k - D^{-1}F(x^k))$ has already been found at the previous iteration as a by-product of evaluating $f(x^k + \beta^m d^k)$. Therefore one need not compute the search direction d^k at the beginning of each iteration.

In the descent method proposed by Fukushima, to prove that the method is globally convergent it is necessary not only the strong monotonicity of mapping but the compactness of constraint set. However, in the case of nonlinear complementarity problem, we can prove that Algorithms 1a and 1b convergence globally if only F is strongly monotone.

Proposition 3.1 *If F is strongly monotone on R^n_+, then*

$$\lim_{x \geq 0,\ \|x\| \to \infty} f(x) = +\infty.$$

Proof. For convenience, we define

$$f_i(x) = F_i(x)^2 - (\max(0, F_i(x) - \delta_i x_i))^2, \tag{16}$$

hence f is written as $f(x) = \sum_{i=1}^{n} \frac{1}{2\delta_i} f_i(x)$. We first show that $f_i(x) \geq 0$ for all $x \geq 0$. If $F_i(x) - \delta_i x_i \leq 0$, then $f_i(x) = F_i(x)^2 \geq 0$. So we consider the case $F_i(x) - \delta_i x_i > 0$. Since $\delta_i > 0$, $x_i \geq 0$ and $F_i(x) > \delta_i x_i$ hold, we see, from (16),

$$\begin{aligned} f_i(x) &= F_i(x)^2 - (F_i(x) - \delta_i x_i)^2 \\ &= \delta_i x_i (2F_i(x) - \delta_i x_i) \\ &\geq (\delta_i x_i)^2 \\ &\geq 0. \end{aligned}$$

Let $\{x^k\}$ be a sequence such that $x^k \geq 0$ and $\| x^k \| \to \infty$. Taking a subsequence, if necessary, there exists a set $I \subset \{1, 2, \ldots, n\}$ such that $x_i^k \to +\infty$ for $i \in I$ and $\{x_i^k\}$ is bounded for $i \notin I$. Without loss of generality, $\{x^k\}$ itself has a such set I. From $\{x^k\}$, we define a sequence $\{y^k\}$ where $y_i^k = 0$ if $i \in I$ and $y_i^k = x_i^k$ if $i \notin I$. From (14) and the definition of y^k, we have

$$\sum_{i \in I} (F_i(x^k) - F_i(y^k)) x_i^k \geq \mu \sum_{i \in I} x_i^{k2}.$$

By Cauchy's inequality

$$\| F(x^k) - F(y^k) \| \| x^k - y^k \| \geq (F(x^k) - F(y^k))^T (x^k - y^k),$$

we have

$$(\sum_{i \in I} (F_i(x^k) - F_i(y^k))^2)^{\frac{1}{2}} (\sum_{i \in I} x_i^{k2})^{\frac{1}{2}} \geq \sum_{i \in I} (F_i(x^k) - F_i(y^k)) x_i^k,$$

hence we have

$$\sum_{i \in I} (F_i(x^k) - F_i(y^k))^2 \geq \mu^2 \sum_{i \in I} x_i^{k2}. \tag{17}$$

Since, from the definition of y^k, $\{y^k\}$ is bounded, $\{F_i(y^k)\}$are also bounded for all i, and $x_i^k \to +\infty$ for all $i \in I$, (17) implies

$$\sum_{i \in I} F_i(x^k)^2 \to \infty.$$

As shown at the beginning of the proof, $f_i(x) = F_i(x)^2 \geq 0$ if $F_i(x) - \delta_i x_i \leq 0$, and $f(x^k) \geq (\delta_i x_i)^2$ if $F_i(x) - \delta_i x_i > 0$. Therefore we have

$$\begin{aligned} f(x^k) &= \sum_{i=1}^{n} \frac{1}{2\delta_i} f_i(x) \\ &\geq \sum_{i \in I} \frac{1}{2\delta_i} f_i(x) \\ &\geq \sum_{i \in I} \frac{1}{2\delta_i} \min(F_i(x)^2, (\delta_i x_i)^2). \end{aligned}$$

Since $x_i^k \to +\infty$ for all $i \in I$ and $\sum_{i \in I} F_i(x^k)^2 \to \infty$, it follows that $f(x^k) \to +\infty$. □

Theorem 3.1 *Suppose that the mapping F is continuously differentiable and strongly monotone with modulus μ on R_+^n. Suppose also that ∇F is Lipschitz continuous on any bounded subset of R_+^n. Then, for any starting point $x^0 \geq 0$, the sequence $\{x^k\}$ generated by Algorithm 1a or Algorithm 1b converges to the unique solution of problem* (1) *if the positive constant σ is chosen to be sufficiently small such that $\sigma < \mu$.*

Proof. By proposition 3.1, the level set $S = \{x | f(x) \leq f(x^0)\}$ is shown to be bounded. Hence ∇F is Lipschitz continuous on S. Since F is continuously differentiable it is easy to show that F is also Lipschitz continuous on S. Under these conditions, it is not difficult to show that ∇f is Lipschitz continuous on S, i.e., there exists a constant $L > 0$ such that

$$\| \nabla f(x) - \nabla f(y) \| \leq L \| x - y \| \quad \text{for all} \quad x, y \in S.$$

The remainder of this proof is the same as the proof of [5, theorem4.2]. □

The second method is a modification of Newton method, which incorporates the line search strategy. The original Newton method for solving the nonlinear complementarity problem (1) generates a sequence $\{x^k\}$ such that $x^0 \geq 0$ and x^{k+1} is determined as $x^{k+1} := \bar{x}$, where $\bar{x}$ is a solution to the following linearized complementarity problem:

$$x \geq 0, F(x^k) + \nabla F(x^k)^T (x - x^k) \geq 0 \text{ and } x^T (F(x^k) + \nabla F(x^k)^T (x - x^k)) = 0. \quad (18)$$

It is shown [13] that, under suitable assumptions, the sequence generated by (18) quadratically converges to a solution x^* of the problem (1), provided that the starting point x^0 is chosen sufficiently close to x^*. Taji et al. [15] have shown that, when the mapping F is strongly monotone with modulus μ, the vector $d := \bar{x} - x^k$ obtained by solving the linearized complementarity problem (18) satisfies the inequality

$$d^T \nabla f(x^k) < -(\mu - \frac{1}{2} \| D \|) \| d \|^2 .$$

Therefore, d is actually a feasible descent direction of f at x^k, if the matrix D is chosen to satisfy $\| D \| = \max_i(\delta_i) < 2\mu$. Using this result, we can construct a modified Newton method for solving the nonlinear complementarity problem (1).

Algorithm 2:

choose $x^0 \geq 0$, $\beta \in (0,1)$, $\gamma \in (0,1)$, $\sigma \in (0,1)$ and a positive diagonal matrix D;
$k := 0$
while convergence criterion is not satisfied **do**
    find $\bar{x}$ such that
        $x \geq 0, F(x^k) + \nabla F(x^k)^T(x - x^k) \geq 0$ and $x^T(F(x^k) + \nabla F(x^k)^T(x - x^k)) = 0$;
    $d^k := \bar{x} - x^k$
    if $f(x^k + d^k) \leq \gamma f(x^k)$ **then**
        $\alpha_k := 1$
    else
        $m := 0$
        while $f(x^k) - f(x^k + \beta^m d^k) < -\sigma\beta^m {d^k}^T \nabla f(x^k)$ **do**
            $m := m + 1$
        endwhile
        $\alpha_k := \beta^m$
    endif
    $x^{k+1} := x^k + \alpha_k d^k$;
    $k := k + 1$
endwhile

By applying the results proved in [15] for the general variational inequality problem, we see that, when the mapping F is strongly monotone with modulus μ, this algorithm is convergent to the solution of (1) for any $x^0 \geq 0$ if the matrix D is chosen such that $\| D \| = \max_i(\delta_i) < 2\mu$. From [15], we also see that the rate of convergence is quadratic if ∇F is Lipschitz continuous on a neighborhood of the unique solution x^* of problem (1) and the strict complementarity condition holds at x^* i.e., $x_i^* = 0$ implies $F_i(x^*) > 0$ for all $i = 1, 2, \ldots, n$.

4 Computational results

In this section, we report some numerical results for Algorithms 1a, 1b and 2 discussed in the previous section. All computer programs were coded in FORTRAN and the runs were made in double precision on a personal computer called Fujitsu FMR-70.

Throughout the computational experiments, the parameters used in algorithms were set as $\beta = \beta_1 = \beta_2 = 0.5$, $\gamma = 0.5$ and $\sigma = 0.0001$. The positive diagonal matrix D was chosen to be the identity matrix multiplied by a positive parameter $\delta > 0$. Therefore the merit function (11) can be written more simply as

$$f(x) = \frac{1}{2\delta} \sum_{i=1}^{n} \left\{ F_i(x)^2 - (\max(0, F_i(x) - \delta x_i))^2 \right\}. \tag{19}$$

The search direction of Algorithms 1a and 1b can also be written as

$$d^k := \max\left(0, x^k - \frac{1}{\delta} F(x^k)\right) - x^k.$$

The convergence criterion was

$$|\min(x_i, F_i(x))| \leq 10^{-5} \quad \text{for all } i = 1, 2, \ldots, n. \tag{20}$$

For comparison purposes, we also tested two popular methods for solving the nonlinear complementarity problem, the projection method [3] and the Newton method without line search [9]. The projection method generates a sequence $\{x^k\}$ such that $x^0 \geq 0$ and x^{k+1} is determined from x^k by

$$x^{k+1} := \max\left(0, x^k - \frac{1}{\delta} F(x^k)\right), \tag{21}$$

for all k. Note that this method may be considered a fixed step-size variant of Algorithms 1a and 1b. When the mapping F is strongly monotone and Lipschitz continuous with constants μ and L, respectively, this method is globally convergent if δ is chosen large enough to satisfy $\delta > L^2/2\mu$ (see [13, Corollary 2.11.]).

The mappings used in our numerical experiments are of the form

$$F(x) = Ix + \rho(N - N^T)x + \phi(x) + c, \tag{22}$$

where I is the $n \times n$ identity matrix, N is an $n \times n$ matrix such that each row contains only one nonzero element, and $\phi(x)$ is a nonlinear monotone mapping with components $\phi_i(x_i) = p_i x_i^4$, where p_i are positive constants. Elements of matrix N and vector c as well as coefficients p_i are randomly generated such that $-5 \leq N_{ij} \leq 5$, $-25 \leq c_i \leq 25$ and $0.001 \leq p_i \leq 0.006$. The results are shown in Tables $1 \sim 4$. All starting points were chosen to be $(0, 0, \ldots, 0)$. In the tables, $\#f$ is the total number of evaluating the merit function f and all CPU times are in seconds and exclude input/output times. The parameter ρ is used to change the degree of asymmetry of F, namely F deviates from symmetry as ρ becomes large. Since the matrix $I + \rho(N - N^T)$ is positive definite for any ρ and $\phi_i(x_i)$ are monotonically increasing for $x_i \geq 0$, the mapping F defined by (22) is strongly monotone on R_+^n.

4.1 Comparison of Algorithms 1a, 1b and the projection method

First we have compared Algorithms 1a, 1b and the projection method (21) by using a 10-dimensional example, in which mapping F is given by

$$F(x) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & -2 & 0 & -3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & -5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 1 & 0 & -5 & 0 & 0 \\ 0 & 0 & -3 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -5 & 0 & 0 & 0 & 0 & 5 & 0 & 1 & 0 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5 & 4 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{pmatrix} + \begin{pmatrix} 0.004x_1^4 \\ 0.004x_2^4 \\ 0.003x_3^4 \\ 0.003x_4^4 \\ 0.006x_5^4 \\ 0.006x_6^4 \\ 0.004x_7^4 \\ 0.004x_8^4 \\ 0.004x_9^4 \\ 0.002x_{10}^4 \end{pmatrix} + \begin{pmatrix} 2 \\ 10 \\ 2 \\ 9 \\ -15 \\ 12 \\ -9 \\ 5 \\ 7 \\ -17 \end{pmatrix} . \quad (23)$$

The results for this problem are shown in Table 1.

In general, the projection method is guaranteed to converge, only if the parameter δ is chosen sufficiently large. In fact, Table 1 shows that when δ is large, the projection method is always convergent, but as δ becomes small, the behavior of the method becomes unstable and eventually it fails to converge.

Table 1 also shows that Algorithms 1a and 1b are always convergent even if δ is chosen small, since line search determines an adequate step size at each iteration. Note that, in Algorithm 1b, the number of iterations is almost constant. This is because we may choose a larger step size when the magnitude of vector d^k is small, i.e. δ is large. This is in contrast with Algorithm 1a, in which step size is bounded by 1 so that the number of iterations increases as δ becomes large.

Algorithms 1a and 1b spend more CPU times per iteration than the projection method, because the former algorithms require overheads of evaluating the merit function f. Moreover, when δ is between 0.1 and 20, Algorithm 1b spends more CPU time than Algorithm 1a, though the number of iterations are almost equal. This is because Algorithm 1b attempts to find a larger step size at each iteration. But, when δ becomes large, Algorithm 1b tends to spend less CPU time than not only Algorithm 1a but also the projection method, because the number of iterations of Algorithm 1b does not increase so drastically.

4.2 Comparison of Algorithm 2 and Newton method

Next we have compared Algorithm 2 and the pure Newton method (18) without line search. For each of the problem sizes $n = 30, 50$ and 90, we randomly generated five test problems. The parameters ρ and δ were set as $\rho = 1$ and $\delta = 10$. The starting point was chosen to be $x = 0$. In solving the linearized subproblem at each iteration of Algorithm 2 and Newton method, we used Lemke's complementarity pivoting method [10] coded by Fukushima [8]. The results are given in Table 2. All numbers shown in Table 2 are

averages of the results for five problems for each case and #Lemke is the total number of pivotings in Lemke's method.

Table 2 shows that the number of iterations of Newton method is consistently larger than that of Algorithm 2 as far as the test problems used in the experiments are concerned. Therefore, since it is time consuming to solve a linear subproblem at each iteration, Algorithm 2 required less CPU time than the pure Newton method in spite of the overheads in line search. Finally we note that, the pure Newton method (18) is not guaranteed to be globally convergent, although it actually converged for all test problems reported in Table 2.

4.3 Comparison of Algorithms 1a and 2

Finally we have compared Algorithms 1a and 2. For each of the problem sizes $n = 30, 50$ and 90, we randomly generated five test problems. To see how these algorithms behave for various degrees of asymmetry of the mapping F, we have tested several values of ρ between 0.1 and 2.0. The starting point was chosen to be $x = 0$. The results are given in Table 3. All numbers shown in Table 3 are averages of the results for five test problems for each case.

Table 3 shows that when the mapping F is close to symmetry, Algorithm 1a converges very fast, and when the mapping becomes asymmetric, the number of iterations and CPU time of Algorithm 1a increase rapidly. On the other hand, in Algorithm 2, while the total number of pivotings of Lemke's method increases in proportion to problem size n, the number of iterations stays constant even when problem size and the degree of asymmetry of F are varied. Hence, when the degree of asymmetry of F is relatively small, that is when ρ is smaller than 1.0, Algorithm 1a requires less CPU time than Algorithm 2.

Note that, since the mapping F used in our computational experience is sparse, complexity of each iteration in Algorithm 1a is small. On the other hand, the code [8] of Lemke's method used in Algorithm 2 to solve a linear subproblem does not make use of sparsity, so that it requires a significant amount of CPU time at each iteration for large problems. If a method that can make use of sparsity is available to solve a linear subproblem, CPU time of Algorithm 2 may decrease. The projected Gauss-Seidel method [2, pp. 397] for solving linear complementarity problem is one of such methods. In Table 4, results of Algorithm 2 using the projected Gauss-Seidel method in place of Lemke's method are given. Table 4 shows that, if the mapping F is almost symmetric, Algorithm 2 converges very fast. But Algorithm 2 fails to converge when the degree of asymmetry increased, because the projected Gauss-Seidel method could not to find a solution to linear subproblem.

Figure 1 illustrates how Algorithms 1a and 2 converged for two typical test problems

with $n = 30$ and 50. In the figure, the vertical axis represents the accuracy of a generated point to the solution, which is evaluated by

$$\mathrm{ACC} = \max_i \{ |\min(x_i, F_i(x))| \mid i = 1, 2, \ldots, n \}. \tag{24}$$

Figure 1 indicates that Algorithm 2 is quadratically convergent near the solution. Figure 1 also indicates that Algorithm 1a is linearly convergent though it has not been proved theoretically.

参考文献

[1] B. H. Ahn, "Solution of Nonsymmetric Linear Complementarity Problems by Iterative Methods," *Journal of Optimization Theory and Applications* 33 (1981) 175-185.

[2] R. W. Cottle, J. S. Pang and R. E. Stone, *The Linear Complementarity Problem* (Academic Press, San Diego, 1992)

[3] S. Dafermos, "Traffic Equilibrium and Variational Inequalities," *Transportation Science* 14 (1980) 42-54.

[4] M. Florian, "Mathematical Programming Applications in National, Regional and Urban Planning," in: M. Iri and K. Tanabe eds., *Mathematical Programming: Recent Developments and Applications* (KTK Scientific Publishers, Tokyo, 1989) pp.283-307.

[5] M. Fukushima, "Equivalent Differentiable Optimization Problems and Descent Methods for Asymmetric Variational Inequality Problems," *Mathematical Programming* 53 (1992) 99-110.

[6] C. B. Garcia and W. I. Zangwill, *Pathways to Solutions, Fixed Points, and Equilibria* (Prentice-Hall, Eaglewood Cliffs N.J. 1981).

[7] 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.

[8] T. Ibaraki and M. Fukushima, *FORTRAN 77 Optimization Programming (in Japanese)* (Iwanami-Shoten, Tokyo, 1991).

[9] N. H. Josephy, "Newton's Method for Generalized Equations," Technical Report No. 1965, Mathematics Research Center, University of Wisconsin (Madison, WI, 1979).

[10] C. E. Lemke, "Bimatrix Equilibrium Points and Mathematical Programming," *Management Science* 11 (1965) 681-689.

[11] O. L. Mangasarian and M. V. Solodov, "Nonlinear Complementarity as Unconstrained and Constrained Minimization," Technical Report No. 1074, Computer Sciences Department, University of Wisconsin (Madison, WI, 1992).

[12] L. Mathiesen, "An Algorithm based on a Sequence of Linear Complementarity Problems Applied to a Walrasian Equilibrium Model : An Example," *Mathematical Programming* 37 (1987) 1-18.

[13] J. S. Pang and D. Chan, "Iterative Methods for Variational and Complementarity Problems," *Mathematical Programming* 24 (1982) 284-313.

[14] P. K. Subramanian, "Fixed Point Methods for The Complementarity Problem," Technical Report No. 2857, Mathematics Research Center, University of Wisconsin (Madison, WI, 1985).

[15] K. Taji, M. Fukushima and T. Ibaraki, "A Globally Convergent Newton Method for Solving Strongly Monotone Variational Inequalities," to appear in *Mathematical Programming.*

[16] R. L. Tobin, "A Variable Dimension Solution Approach for the General Spatial Price Equilibrium Problem," *Mathematical Programming* 40 (1988) 33-51.

Table 1:Results for Algorithm 1a, 1b and the projection method ($n = 10$, $\rho = 1$)

	Algorithm 1a			Algorithm 1b			projection method	
δ	#Iterations	$\#f$	CPU	#Iterations	$\#f$	CPU	#Iterations	CPU
0.1	1380	9683	18.43	1380	9683	18.45	oscillation	
0.5	307	1527	3.01	307	1527	3.04	oscillation	
1	328	1305	2.61	328	1305	2.63	oscillation	
2	338	1009	2.02	338	1009	2.13	oscillation	
3	353	950	1.91	353	951	2.04	oscillation	
4	342	684	1.42	342	685	1.55	oscillation	
5	256	504	1.05	256	511	1.16	oscillation	
6	350	591	1.25	351	701	1.72	oscillation	
6.2	327	552	1.17	337	674	1.66	oscillation	
6.3	377	600	1.29	377	754	1.91	9118	13.94
6.5	380	583	1.25	376	752	1.93	1594	2.43
7	388	524	1.15	385	770	2.07	610	0.93
8	338	339	0.78	337	676	1.98	338	0.51
9	271	272	0.62	270	542	1.59	271	0.43
10	244	245	0.56	242	487	1.41	244	0.36
12	229	230	0.52	232	468	1.36	229	0.34
15	239	240	0.55	239	488	1.41	239	0.35
20	272	273	0.62	254	636	1.72	272	0.41
50	549	550	1.23	229	920	2.17	549	0.79
100	1036	1037	2.30	229	1149	2.63	1036	1.47
200	2008	2009	4.51	239	1385	3.05	2008	2.88
500	5007	5008	11.09	239	1674	3.59	5007	7.08
1000	9998	9999	22.18	372	2605	5.55	9998	14.18

Table 2:Results for Algorithm 2 and Newton method ($\rho = 1$)

	Algorithm 2				Newton method		
n	#Iterations	$\#f$	#Lemke	CPU	#Iterations	#Lemke	CPU
30	5.6	8.0	80.8	4.202	8	115.2	5.840
50	5.6	7.6	156.0	19.554	8	216.4	26.142
90	6.0	9.2	277.8	104.084	8	358.8	135.690

Table 3:Results for Algorithm 1a and 2

		Algorithm 1a			Algorithm 2			
ρ	n	#Iterations	$\#f$	CPU	#Iterations	$\#f$	#Lemke	CPU
	30	83.6	84.6	0.410	5.6	9.2	82.4	4.244
0.1	50	91.0	92.0	0.912	6.0	10.0	142.2	18.588
	90	110.4	111.4	1.972	6.0	10.0	250.6	94.678
	30	98.6	99.6	0.408	6.4	9.6	95.2	4.946
0.2	50	107.4	108.4	0.880	6.4	10.0	162.8	20.702
	90	110.2	111.2	2.032	6.6	10.2	275.0	103.580
	30	91.0	92.0	0.620	6.4	9.2	90.8	4.342
0.3	50	94.8	95.8	0.890	6.2	9.0	157.6	19.804
	90	120.0	121.0	1.812	6.4	9.6	267.6	100.892
	30	88.0	89.0	0.302	5.8	8.4	86.0	4.458
0.5	50	108.0	109.0	0.864	6.2	8.6	164.4	20.502
	90	128.6	129.6	1.776	6.2	9.0	275.0	100.776
	30	168.4	189.0	1.066	5.6	8.0	82.0	4.452
0.8	50	290.2	316.8	2.892	6.0	8.0	160.4	20.054
	90	765.2	965.4	15.680	6.2	9.4	279.0	104.734
	30	407.2	653.2	3.430	5.6	8.0	80.8	4.202
1.0	50	518.6	983.4	8.426	5.6	7.6	156.0	19.554
	90	865.4	2119.8	32.582	6.0	9.2	277.8	104.084
	30	1197.0	3712.4	19.432	5.4	7.8	80.0	4.106
1.5	50	1604.0	4925.4	41.982	5.2	7.2	145.8	18.158
	90	2856.0	9777.0	149.040	6.0	8.8	278.0	104.276
	30	3194.4	12689.2	65.636	5.0	7.4	76.0	3.888
2.0	50	3844.0	15934.4	135.604	5.2	7.2	145.2	17.824
	90	4957.4	20903.6	312.178	5.8	8.6	281.4	105.118

Table 4:Results for Algorithm 2 (Gauss-Seidel version)

		Algorithm 2			
ρ	n	#Iterations	$\#f$	#Gauss	CPU
	30	5.6	9.2	42.4	0.248
0.1	50	6.0	10.0	41.8	0.486
	90	6.0	10.0	55.8	1.200
	30	3 of 5 failed			
0.2	50	3 of 5 failed			
	90	failed			
	30				
0.3 ~ 2.0	50	failed			
	90				

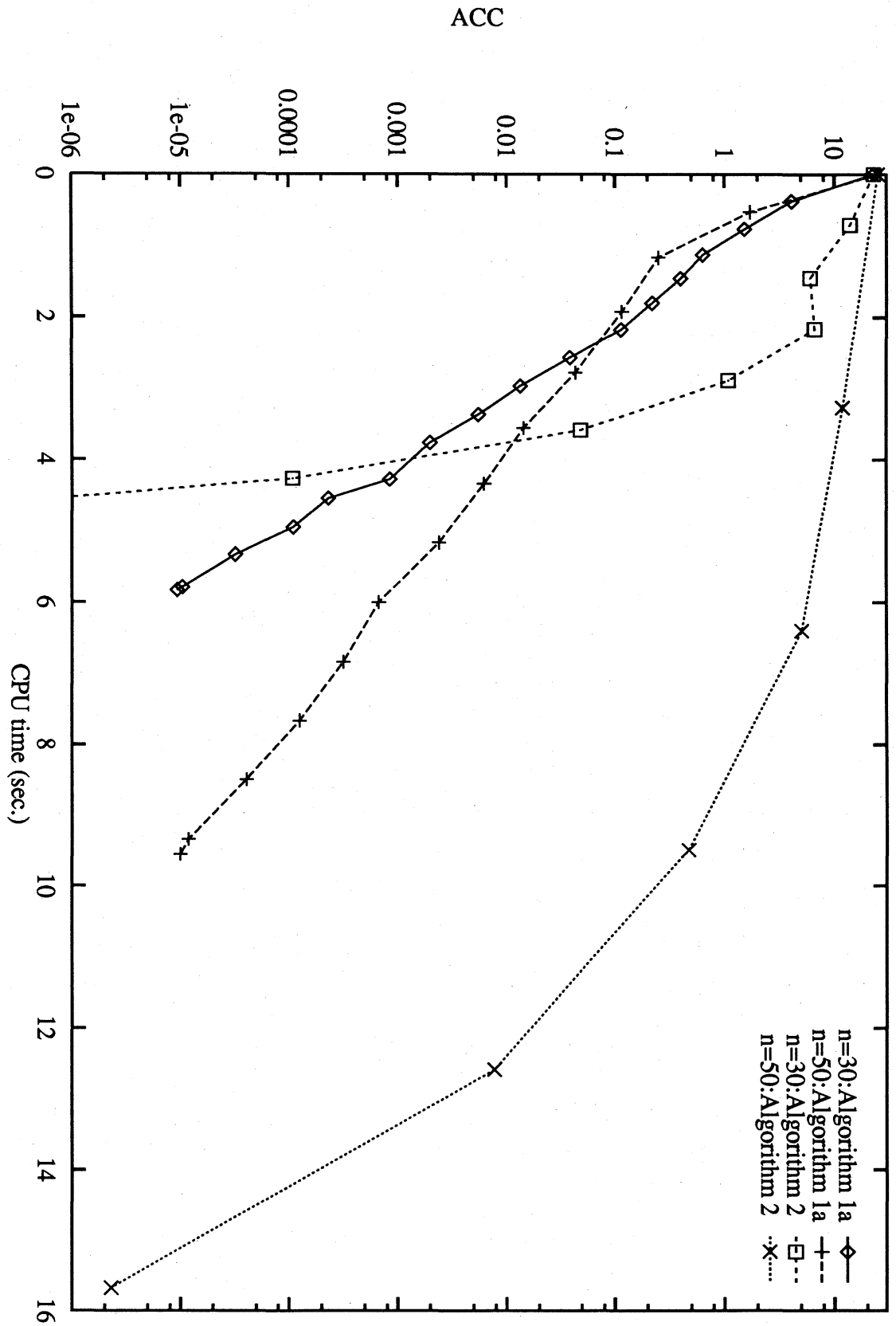
ACC
10
1
0.1
0.01
0.001
0.0001
1e-05
1e-06
0
2
4
6
8
10
12
14
16
CPU time (sec.)
n=30:Algorithm 1a
n=50:Algorithm 1a
n=30:Algorithm 2
n=50:Algorithm 2

Figure 1