J. Nonlinear Funct. Anal. 2024 (2024) 23 https://doi.org/10.23952/jnfa.2024.23

MATHRES Journal of Nonlinear Functional Analysis
Available online at http://jnfa.mathres.org

A SMOOTHING NEWTON ALGORITHM FOR LINEAR WEIGHTED COMPLEMENTARITY PROBLEMS

PANJIE TIAN[1], ZHENSHENG YU[2], MINGWANG ZHANG[1,3,*], WEN YE[1]

[1]Mathematics Department, Anhui Institute of Information Technology, Wuhu 241000,Anhui, China
[2]College of Science, University of Shanghai for Science and Technology, Shanghai 200093, China
[3]College of Science, China Three Gorges University, Yichang 443002, Hubei, China

Abstract. In this paper, we propose a smoothing Newton algorithm for linear weighted complementarity problems (LWCP). A new smoothing function is constructed, and the LWCP is reformulated as a smooth equations based on the function, and solved by the Newton method. Under mild conditions, the convergence of the method is proved. Finally, numerical results are provided to show the effectiveness of the algorithm in solving the free boundary problems of obstacle and the large-scale Fisher equilibrium problems.

Keywords. Smoothing Newton method; Smoothing function; Weighted complementarity problem.
2020 MSC. 65K05, 90C33.

1. INTRODUCTION

In [11], Tian, Yu, and Yuan studied the solution of the LWCP [7, 8], which is to find a pair of $(x,s,y) \in R^n \times R^n \times R^m$ such that

$$x \geq 0, s \geq 0, xs = w, Px + Qs + Ry = a, \tag{1.1}$$

where $P \in R^{(n+m)\times n}$, $Q \in R^{(n+m)\times n}$, $R \in R^{(n+m)\times m}$, and $a \in R^{n+m}$ are given matrices and vector, and the matrix R is assumed to have full column rank.

Problem (1.1), introduced by Potra [7] in 2012, has been widely studied for its significant applications in economy, management, and so on. Some equilibrium problems were transformed into nonlinear complementarity problem (NCP), such as the famous Fisher market equilibrium problem [14], however LWCP can be solved faster and more efficiently. Linear programming and weighted center problem [1], and the more general quadratic programming and weighted center problem [7] can all be transformed into LWCP.

*Corresponding author.
E-mail address: zmwang@ctgu.edu.cn (M. Zhang).
Received June 11, 2024; Accepted October 17, 2024.

Recently, numerical effective algorithms were used to solve problems (1.1) [2, 15, 10]. For instances, Asadi et al. [2] proposed the full-Newton step interior-point method for solving the monotone LWCP; Zhang [15] studied the smoothing Newton type method for solving the LWCP; and Tang and Zhou [10] considered the L-M method for the Nonlinear Weighted Complementarity Problem (NWCP). The interior point method relies on the feasible initial point, and the convergence rate of the L-M method is slow. The smoothing Newton method does not depend on the choice of the initial point and converges very fast. Therefore, in this paper, the smoothing Newton method is considered to solve problem (1.1). For the solution of problem (1.1), we consider constructing a new smoothing function from the generalization of the NCP function. Due to the existence of weighted terms, not all the NCP functions can be directly generalized. We mention here that the NCP functions in the form of the FB function were generalized to the WCP functions in [6, 5].

In this paper, motivated by the NCP function in [12], we introduce a smoothing function. The LWCP is reformulated as a system of smooth equations and then we use the Newton method to obtain an approximate solution of problem (1.1). The numerical experiments demonstrate the effectiveness of the algorithm in solving the stochastic problem, the obstacle problem free boundary and the Fisher equilibrium problems.

This paper is structured as follows. In Section 2, we introduce a family of new WCP functions and reformulate the LWCP as a smooth equation. In Section 3, we propose a smoothing Newton-type method to solve the equation and demonstrate the convergence of the algorithm. Numerical results are presented in Section 4. Some conclusions are drawn in Section 5, the last section.

Some notation used throughout the paper is as follows. The superscript T denotes transpose. R^n represents the space of n-dimensional real column vectors, and R_{++} denotes the positive orthant in R^n. The matrix I denotes the identity matrix, and $\|\cdot\|$ denotes the 2-norm. All vectors in this article are column vectors.

2. The Smooth Reformulation of the LWCP

In this section, we introduce a smoothing function which can be used to reformulate the LWCP as a system of smooth equations.

Definition 2.1. [10] For a given $c \geq 0$, a function $\phi : R^2 \to R$ is called a weighted complementarity function if it satisfies $\phi^c(a,b) = 0 \Leftrightarrow a \geq 0, b \geq 0, ab = c$. If $c = 0$, then $\phi^c(a,b)$ reduces to the NCP function.

In this paper, to solve LWCP (1.1), we consider using the WCP function obtained by the extension of the NCP function. However, not all NCP functions can be directly generalized to WCP functions due to the existence of weighting terms. For example, The cosine-type NCP function given in [13],

$$\phi(\mu,a,b) = a+b-\mu\left(\ln 2+\ln\left(1+\cos\left((a-b)/\mu\right)\right)\right).$$

Numerous scholars explored the extensions of the Fischer-Burmeiste (FB) function and its generalized forms to WCP functions. For example, based on the symmetric perturbation function referenced in [3], Liu et al. [6] proposed the subsequent WCP function:

$$\phi_c(\mu,a,b) = (1+\mu)(a+b) - \sqrt{(a+\mu b)^2+(\mu a+b)^2+2c+2\mu^2}.$$

Jiang[5], grounded on the FB function in[4], constructed the ensuing WCP function:

$$\phi_c(\mu,a,b)=(1+\mu)(a+b)-\sqrt{(1-\mu)^2(a-b)^2+4c+4\mu}.$$

In [11], based on the NCP function in [12], we introduce the following WCP function:

$$\phi_\tau^c(a,b)=a+b-\sqrt{\tau(a-b)^2+(1-\tau)(a^2+b^2)+2(1+\tau)c}, \tag{2.1}$$

where $\tau\in[0,1]$.

Theorem 2.2. [11] *Let $\phi_\tau^c(a,b)$ be defined by (2.1), where $\tau\in[0,1]$ and c is a positive constant. Then $\phi_\tau^c(a,b)=0\Leftrightarrow a\geq 0,\ b\geq 0$, and $ab=c$.*

In this paper, we introduce a new smoothing function $\phi_\tau^c:R^3\to R$ which is defined as follows:

$$\phi_\tau^c(\mu,a,b)=a+b-\sqrt{\tau(a-b)^2+(1-\tau)(a^2+b^2)+2(1+\tau)c+2\mu^2}, \tag{2.2}$$

with $\tau\in[0,1]$.

Theorem 2.3. *Let $\phi_\tau^c(a,b),\phi_\tau^c(\mu,a,b)$ be defined by (2.1) and (2.2) respectively. Then, for any $\mu>0$ and $(a,b)\in R^2$, if $\mu\to 0$, $|\phi_\tau^c(a,b)-\phi_\tau^c(\mu,a,b)|\to 0$.*

Given weight vector $w\in R_+^n$, Let $z:=(\mu,x,s,y)\in R^{2n+m+1}$ and

$$H(z)=H(x,s,y)=\begin{pmatrix}\mu\\ Px+Qs+Ry-a\\ \Phi_\tau^w(\mu,x,s)\end{pmatrix}, \tag{2.3}$$

where

$$\Phi_\tau^w(\mu,x,s)=\begin{pmatrix}\phi_\tau^{w_1}(\mu,x_1,s_1)\\ \vdots\\ \phi_\tau^{w_n}(\mu,x_n,s_n)\end{pmatrix}. \tag{2.4}$$

Then problem (1.1) is equivalent to $H(z)=0$.

Definition 2.4. [7] LWCP (1.1) is called monotone if $P\Delta x+Q\Delta s+R\Delta y=0$ implies $\Delta x^T\Delta s=0$, and it is called skew-symmetric if $P\Delta x+Q\Delta s+R\Delta y=0$ implies $\Delta x^T\Delta s=0$.

Lemma 2.5. *Let $H(z):R^{2n+m+1}\to R^{2n+m+1},\Phi_\tau^w:R^{2n+1}\to R^n$ be defined by (2.3) and (2.4) respectively. Then*

(i) Φ_τ^w is continuously differentiable at any $z=(\mu,x,s,y)\in R_{++}\times R^{2n+m}$.

(ii) $H(z)$ is continuously differentiable at any $z=(\mu,x,s,y)\in R_{++}\times R^{2n+m}$ with its Jacobian

$$H'(z)=\begin{pmatrix}I & 0 & 0 & 0\\ 0 & P & Q & R\\ A & D_1 & D_2 & D_2\end{pmatrix}, \tag{2.5}$$

where

$$D_1=diag\left\{1-\frac{x_i-\tau s_i}{h_\tau^{w_i}}\right\},i=1,2,\cdots,n.$$

$$D_2=diag\left\{1-\frac{s_i-\tau x_i}{h_\tau^{w_i}}\right\},i=1,2,\cdots,n.$$

$$A=vec\left\{-\frac{2\mu}{h_\tau^{w_i}}\right\},i=1,2,\cdots,n.$$

$$h_\tau^{w_i}(\mu,x_i,s_i)=\sqrt{\tau(x_i-s_i)^2+(1-\tau)(x_i^2+s_i^2)+2(1+\tau)w_i+2\mu^2},i=1,2,\cdots,n.$$

(iii) If the LWCP (1.1) is monotone, then the matrix $H'(z)$ is non-singular on $R_{++}\times R^{2n+m}$.

Proof. It is easy to see that $\Phi_\tau^w(\mu,x,s)$ is continuously differentiable at any $z=(\mu,x,s,y)\in R_{++}\times R^{2n+m}$ with $\mu>0$. For any $\mu>0$, a straightforward calculation from (2.3) yields (2.5). If $H'(z)\Delta z=0$, then

$$\begin{cases}\Delta\mu=0,\\ A\Delta\mu+D_1\Delta x+D_2\Delta s=0,\\ P\Delta x+Q\Delta s+R\Delta y=0.\end{cases}$$

Subsequently, we demonstrate $\Delta z=0$. It is not difficult to see that $D_1\Delta x+D_2\Delta s=0$. Firstly, we prove that D_1 and D_2 are positive definite. That is to say, it is only necessary to prove that their diagonal elements are all greater than 0, which means

$$1-\frac{x_i-\tau s_i}{\sqrt{\tau(x_i-s_i)^2+(1-\tau)\left(x_i^2+s_i^2\right)+2\left(1+\tau\right)w_i+2\mu^2}}>0.$$

By

$$\tau(x_i-s_i)^2+(1-\tau)\left(x_i^2+s_i^2\right)+2\left(1+\tau\right)w_i+2\mu^2=x_i^2+s_i^2-2\tau x_is_i+2\left(1+\tau\right)w_i+2\mu^2.$$

we have

$$1-\frac{x_i-\tau s_i}{\sqrt{x_i^2+s_i^2-2\tau x_is_i+2\left(1+\tau\right)w_i+2\mu^2}}>0.$$

Observe that

$$\begin{aligned}x_i^2+s_i^2-2\tau x_is_i+2\left(1+\tau\right)w_i+2\mu^2&=x_i^2-2\tau x_is_i+\tau^2s_i^2-\tau^2s_i^2+s_i^2+2\left(1+\tau\right)w_i+2\mu^2\\&=x_i^2-2\tau x_is_i+\tau^2s_i^2+\left(1-\tau^2\right)s_i^2+2\left(1+\tau\right)w_i+2\mu^2\\&=(x_i-\tau s_i)^2+\left(1-\tau^2\right)s_i^2+2\left(1+\tau\right)w_i+2\mu^2\\&=(x_i-\tau s_i)^2+m_0,\end{aligned}$$

where $m_0=\left(1-\tau^2\right)s_i^2+2\left(1+\tau\right)w_i+2\mu^2>0$, and $\sqrt{(x_i-\tau s_i)^2+m_0}>\sqrt{(x_i-\tau s_i)^2}$. Thus

$$1-\frac{1}{\sqrt{(x_i-\tau s_i)^2+m_0}}>1-\frac{1}{\sqrt{(x_i-\tau s_i)^2}}.$$

$$1-\frac{|x_i-\tau s_i|}{\sqrt{(x_i-\tau s_i)^2+m_0}}>1-\frac{|x_i-\tau s_i|}{\sqrt{(x_i-\tau s_i)^2}}.$$

which implies

$$1-\frac{x_i-\tau s_i}{\sqrt{(x_i-\tau s_i)^2+m_0}}>1-\frac{x_i-\tau s_i}{\sqrt{(x_i-\tau s_i)^2}}=1-\frac{x_i-\tau s_i}{x_i-\tau s_i}=0$$

with $x_i-\tau s_i>0$, and

$$1<1-\frac{x_i-\tau s_i}{\sqrt{(x_i-\tau s_i)^2+m_0}}<1-\frac{x_i-\tau s_i}{\sqrt{(x_i-\tau s_i)^2}}=1-\frac{x_i-\tau s_i}{-(x_i-\tau s_i)}=2$$

with $x_i - \tau s_i < 0$. Therefore, it can be concluded that

$$1 - \frac{x_i - \tau s_i}{\sqrt{x_i^2 + s_i^2 - 2\tau x_i s_i + 2(1+\tau) w_i + 2\mu^2}} > 0,$$

which leads to the conclusion that D_1 is a positive definite matrix. Similarly D_2 is also a positive matrix. Thus $\Delta x = -D_1^{-1} D_2 \Delta s$. On one hand, we have $\Delta x^T \Delta s = -\Delta s^T D_1^{-1} D_2 \Delta s \leq 0$. On the other hand, from the monotone property of problem (1.1), we have $\Delta x^T \Delta s \geq 0$. Again, from the positive definite of matrix $D_1^{-1} D_2$, we have $\Delta s = 0, \Delta x = -D_1^{-1} D_2 \Delta s = 0$. Since R has full column rank, it is obvious that $\Delta y = 0$. This completes the proof. □

3. Algorithm and Convergence Analysis

In this section, we propose a smoothing Newton-type algorithm based on the smoothing function introduced in Section 2, and prove its convergence.

(Algorithm 3.1. A smoothing Newton method)

Step 0. Choose parameters $\gamma, \delta \in (0,1), \sigma \in (0, 1/2)$. Let $z^0 = (\mu_0, x^0, s^0, y^0) \in R_{++} \times R^{2n+m}$ be arbitrary initial point. Choose $0 < \varepsilon < 1$. Set $k := 0$.

Step 1. If $\left\| H(z^k) \right\| \leq \varepsilon$, then stop.

Step 2. Compute $d_k = (\Delta\mu_k, \Delta x^k, \Delta s^k, \Delta y^k)$ by

$$H(z^k) + H'(z^k) d_k = \sigma_k r_k, \tag{3.1}$$

$$\text{where } \sigma_k = \min\{\sigma, \mu_k\}, r_k = \begin{pmatrix} \mu_k \\ 0 \end{pmatrix}.$$

Step 3. Choose $\alpha_k = \delta^{j_k}$, where j_k is the smallest nonnegative integer j satisfying

$$\left\| H\left(z^k + \delta^j d_k\right) \right\|^2 \leq [1 - \gamma(1 - 2\sigma_k)\delta^j] \left\| H\left(z^k\right) \right\|^2. \tag{3.2}$$

Step 4. Set $z^{k+1} = z^k + \alpha_k d_k, k = k+1$. Go to Step1.

The following theorem shows that the algorithm is well-defined.

Theorem 3.1. *Let $z^0 = (\mu_0, x^0, s^0, y^0) \in R_{++} \times R^{2n+m}$ be the initial point of Algorithm 3.1. If the LWCP (1.1) is monotone, then Algorithm 3.1 is well-defined and generates an iteration sequence $\left\{ z^k \middle| z^k = \left(\mu_k, x^k, s^k, y^k\right) \in R_{++} \times R^{2n+m} \right\}$ satisfies $0 < \mu_{k+1} < \mu_k \leq \mu_0$ for all k.*

Proof. By Lemma 2.5, we see that $H'(z)$ is non-singular on $R_{++} \times R^{2n+m}$ and hence Step 2 is well-defined. Thus

$$H\left(z^k + \alpha d_k\right) = H\left(z^k\right) + \alpha H'\left(z^k\right) d_k + o(\alpha).$$

Using (3.1), we have

$$H\left(z^k + \alpha d_k\right) = (1 - \alpha) H\left(z^k\right) + \alpha \sigma_k r_k + o(\alpha).$$

In view of Step 2, we have

$$\begin{aligned}\left\|H(z^k+\alpha d_k)\right\|^2 &\le (1-\alpha)^2\left\|H(z^k)\right\|^2+2\alpha(1-\alpha)\sigma_k {r_k}^T H(z^k)+\alpha^2\sigma_k^2\mu_k^2+o(\alpha)\\ &\le (1-\alpha)^2\left\|H(z^k)\right\|^2+2\alpha(1-\alpha)\sigma_k\left\|H(z^k)\right\|^2+\alpha^2\sigma_k^2\left\|H(z^k)\right\|^2+o(\alpha)\\ &\le [1-(1-2\sigma_k)\alpha]\left\|H(z^k)\right\|^2+o(\alpha).\end{aligned}$$

If α is sufficiently small, we have $\left\|H(z^k+\alpha d_k)\right\|^2 \le [1-\gamma(1-2\sigma_k)\alpha]\left\|H(z^k)\right\|^2$. which implies that Step 3 is well-defined. From (3.1), we obtain $\mu_{k+1}=\mu_k+\alpha_k\Delta\mu_k=(1-\alpha_k)\mu_k+\alpha_k\sigma_k\mu_k>0$. Thus, from $\mu_0>0$, we further obtain that Algorithm 3.1 is well defined. By Step 2, we have

$$\mu_{k+1}=(1-\alpha_k)\mu_k+\alpha_k\sigma_k\mu_k<(1-\alpha_k)\mu_k+\alpha_k\mu_k=\mu_k.$$

Therefore, $0<\mu_{k+1}<\mu_k\le\mu_0$. The proof is complete. □

Assumption 3.2. Let $\left\{z^k\,\middle|\,z^k=\left(\mu_k,x^k,s^k,y^k\right)\in R_{++}\times R^{2n+m}\right\}$ be the iteration sequence generated by Algorithm 3.1, and let $\left\{z^k\right\}\subset\Omega$, where Ω is a bounded set.

Lemma 3.3. *Let Assumption 3.2 hold. Then $\{\mu_k\}$ converges to zero.*

Proof. By Algorithm 3.1, we see that $\{\mu_k\}$ is monotonically decreasing and bounded. Then these exists a constant $\bar{\mu}\ge 0$ such that $\lim_{k\to\infty}\mu_k=\bar{\mu}$. If $\bar{\mu}=0$, then the conclusion holds. Now we suppose that $\bar{\mu}>0$. Let $z^*=(\bar{\mu},x^*,s^*,y^*)$. Then $\lim_{k\to\infty}\left\{z^k\right\}=z^*$. From Step 2 of Algorithm 3.1, we have $H'(z^k)d_k=\sigma_k r_k-H(z^k),\sigma_k\le\sigma,\|r_k\|\le\mu_0$. Using this fact, we further obtain from $\left\|H\left(z^k\right)\right\|\le\left\|H\left(z^0\right)\right\|$ that $H'(z^k)d_k$ is uniformly bounded, which shows that $\left\|H'(z^k)d_k\right\|\le M_0$, where M_0 is a positive constant. Since j_k is the smallest non-negative integer such that (3.2) holds, we have

$$\left\|H\left(z^k+\delta^{j_k-1}d_k\right)\right\|^2>[1-\gamma(1-2\sigma_k)\delta^{j_k-1}]\left\|H\left(z^k\right)\right\|^2,$$

i.e.,

$$\frac{\left\|H\left(z^k+\delta^{j_k-1}d_k\right)\right\|^2-\left\|H\left(z^k\right)\right\|^2}{\delta^{j_k-1}}>-\gamma(1-2\sigma_k)\delta^{j_k-1}\left\|H\left(z^k\right)\right\|^2. \tag{3.3}$$

Moreover, by Theorem 3.1, we have

$$H\left(z^k+\delta^{j_k-1}d_k\right)=H\left(z^k\right)+\delta^{j_k-1}H'\left(z^k\right)d_k+o\left(\delta^{j_k-1}\right).$$

Thus

$$\begin{aligned}&\left\|H\left(z^k+\delta^{j_k-1}d_k\right)\right\|^2\\ &=\left\|H\left(z^k\right)+\delta^{j_k-1}H'\left(z^k\right)d_k\right\|^2+2\left(H\left(z^k\right)+\delta^{j_k-1}H'\left(z^k\right)d_k\right)^T o\left(\delta^{j_k-1}\right)+o\left(\delta^{j_k-1}\right)\\ &\le\left\|H\left(z^k\right)\right\|^2+2\delta^{j_k-1}H\left(z^k\right)^T H'\left(z^k\right)d_k+\left(\delta^{j_k-1}\right)^2\left\|H'\left(z^k\right)d_k\right\|^2+o\left(\delta^{j_k-1}\right)\\ &\le\left\|H\left(z^k\right)\right\|^2+2\delta^{j_k-1}H\left(z^k\right)^T H'\left(z^k\right)d_k+\left(M_0\delta^{j_k-1}\right)^2+o\left(\delta^{j_k-1}\right).\end{aligned}$$

It follows that

$$\frac{\left\|H\left(z^k+\delta^{j_k-1}d_k\right)\right\|^2-\left\|H\left(z^k\right)\right\|^2}{\delta^{j_k-1}}$$
$$\leq 2H\left(z^k\right)^T\left(\sigma_k r_k-H\left(z^k\right)\right)+M_0^2\delta^{j_k-1}+\frac{o\left(\delta^{j_k-1}\right)}{\delta^{j_k-1}}.$$

Using this fact, we further obtain from (3.3) that

$$2H\left(z^k\right)^T\left(\sigma_k r_k-H\left(z^k\right)\right)+M_0^2\delta^{j_k-1}+\frac{o\left(\delta^{j_k-1}\right)}{\delta^{j_k-1}}\geq-\gamma(1-2\sigma_k)\left\|H\left(z^k\right)\right\|^2. \tag{3.4}$$

Taking limits on both sides of (3.4), we have

$$2H(z^*)^T\left(\sigma^* r^*-H\left(z^*\right)\right)\geq-\gamma(1-2\sigma^*)\left\|H\left(z^*\right)\right\|^2. \tag{3.5}$$

Since $\sigma^*=\min\{\sigma,\mu^*\}, r^*=\begin{pmatrix}\mu^*\\0\end{pmatrix}$, we have

$$2\left(\sigma^*-1\right)\left\|H\left(z^*\right)\right\|^2\geq-\gamma(1-2\sigma^*)\left\|H\left(z^*\right)\right\|^2.$$

Moreover, we have $2\left(\sigma^*-1\right)\geq-\gamma(1-2\sigma^*)$, i.e.,

$$\sigma^*\geq\frac{2-\gamma}{2(1-\gamma)}>\frac{1}{2}>\sigma>\sigma^*,$$

which is a contradiction. Therefore, we have $\bar{\mu}=0$. Hence, the result of this lemma holds. □

Theorem 3.4. *Let 3.2 hold. Let $z=(\mu,x,s,y)\in R_{++}\times R^{2n+m}$, $H'(z)$ be non-singular on $R_{++}\times R^{2n+m}$. Then each accumulation point of $\{z^k\}$ is a solution to $H(z)=0$.*

Proof. Let z^* be the accumulation point of $\{z^k\}$. If $\|H(z^*)\|=0$, then the conclusion holds. Now we suppose that $\|H(z^*)\|>0$. Observe that $\{\mu_k\}$ converges to 0 by Lemma 3.3. Thus $\{\sigma_k=\min\{\sigma,\mu_k\}\}$ also converges to 0, i.e., $\sigma^*=0$. In addition, by (3.5), we have

$$-2\|H\left(z^*\right)\|^2\geq-\gamma\|H\left(z^*\right)\|^2.$$

This contradicts the fact that $\gamma\in(0,1)$. Thus $\|H(z^*)\|=0$. This completes the proof. □

4. Numerical Results

In this section, we report some numerical results. All experiments are conducted on a ThinkPad480 with a 1.8GHz CPU and 8.0GB RAM. The codes are run in MATLAB R2018b under Win10.

4.1. **Algorithm 3.1 for solving LWCP.**

In this subsection, Algorithm 3.1 is compared with the algorithm in [15].

Example 4.1. We first generate the matrices P, Q, R and vector a as follows:

$$P=\begin{pmatrix}A\\M\end{pmatrix},Q=\begin{pmatrix}0\\-I\end{pmatrix},P=\begin{pmatrix}0\\-A^T\end{pmatrix},a=\begin{pmatrix}b\\-f\end{pmatrix}, \tag{4.1}$$

where $B=rand\,(m,n-m),A=(eye\,(m),-B)\in R^{m\times n}$, and $M=diag\,(rand\,(n,1))\in R^{n\times n}$. We choose $x^0=rand\,(n,1),f=rand\,(n,1)$. Let $b:=Ax^0,s^0=Mx^0+f,w=x^0s^0$, which implies the LWCP is monotone. The parameters used in Algorithm 3.1 are chosen as: $\sigma=0.001,\delta=0.3,\gamma=0.001$.

In the experiments, we set $\tau = rand\,(1)$ and μ_0 is chosen as 0.1, 0.01, 0.001, 0.0001, respectively. The numerical results are presented in Figures 1, 2, 3, Tables 1 and 2, respectively, where, AIT, ACPU, and ANH denote respectively the average number of iterations, the average CPU time (unit seconds), and the average number of iterations at the end of 10 random experiments. SNW represents our experimental result, and ZSNW is the experimental result in [15].

Figure 1 gives the convergence curve of $\left\|H\left(z^k\right)\right\|$, with $\mu_0 = 0.1, n = 3000, m = \frac{n}{2}$. We can clearly see that Algorithm 3.1 converges faster for the identical conditions.

Figures 2 and 3 show the numerical results of the comparison for ACPU and AIT, respectively, taken as $\mu_0 = 0.1, m = \frac{n}{2}; \mu_0 = 0.01, m = \frac{n}{2}$. It can be seen from the figures that with the increase of dimension, the AIT of Algorithm 3.1 fluctuates slightly, but it is always smaller than the AIT in [15]. The ACPU increases steadily and always smaller than the ACPU in [15].

Tables 1 and 2 show the numerical results with $\mu_0 = 0.001; \mu_0 = 0.0001$. From the tables, it can be seen that regardless of the initial value of μ_0, Algorithm 3.1 can complete the experiment in less time and through fewer iterations.

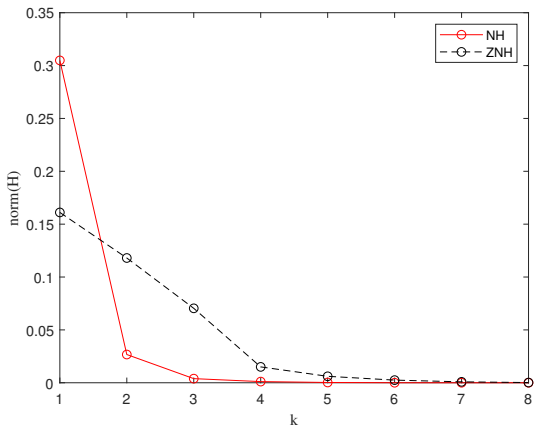
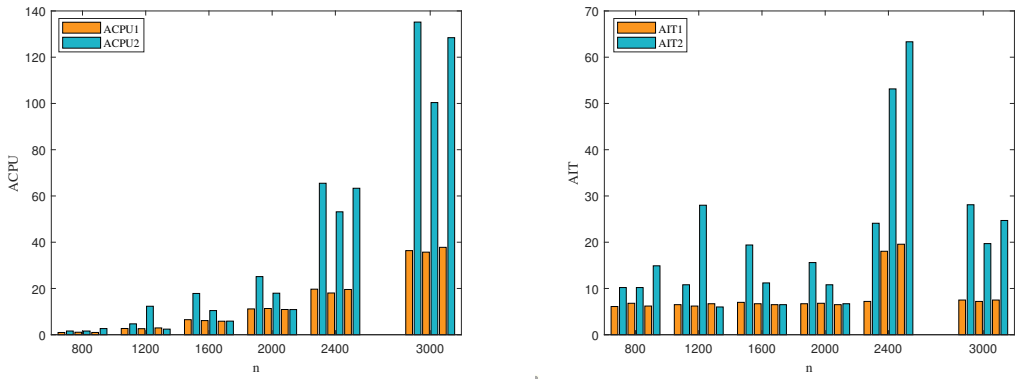


FIGURE 1. Convergence curve of $\left\|H\left(z^k\right)\right\|$ at the k-th iteration.



(a) Average CPU time (b) Average number of iterations

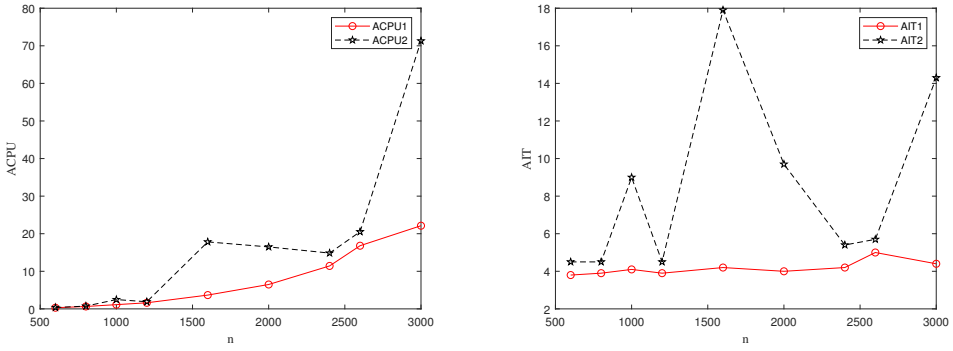
FIGURE 2. Comparison bars of solving LWCP (4.1) ($\mu_0 = 0.1, m = \frac{n}{2}$)

TABLE 1. Numerical results of solving LWCP(4.1) ($\mu_0 = 0.001$)

m	n	SNW			ZSNW		
		AIT	ACPU	ANH	AIT	ACPU	ANH
800	1000	3.1	1.0906	9.4211×10^{-13}	4.0	1.3375	1.1128×10^{-11}
		3.0	1.0037	1.8647×10^{-12}	3.5	1.1832	2.4864×10^{-11}
		3.0	0.9997	1.0186×10^{-12}	3.7	1.2715	1.3293×10^{-11}
1000	1500	3.0	2.5545	5.5081×10^{-12}	3.7	3.1124	3.7169×10^{-12}
		3.0	2.5837	3.1485×10^{-12}	4.1	3.5862	1.3404×10^{-11}
		3.1	2.6129	4.6913×10^{-12}	3.8	3.3008	4.5907×10^{-12}
1500	2000	3.0	6.0378	4.8616×10^{-12}	4.4	9.0275	1.0778×10^{-11}
		3.0	6.1157	3.7813×10^{-12}	3.8	7.6807	7.2493×10^{-12}
		3.0	6.0974	4.4438×10^{-12}	4.1	8.4660	1.7469×10^{-11}
2000	2500	3.1	12.1373	8.1028×10^{-12}	4.5	17.7169	1.7398×10^{-11}
		3.1	12.1696	5.2714×10^{-12}	4.0	15.2845	6.5150×10^{-12}
		3.0	11.8252	5.2305×10^{-12}	4.2	16.5506	5.4734×10^{-12}
2500	3000	3.0	20.1076	5.2506×10^{-12}	8.5	57.4832	1.3548×10^{-11}
		3.0	20.7938	6.2666×10^{-12}	4.5	30.3109	1.4381×10^{-11}
		3.1	21.9351	1.3075×10^{-11}	9.0	60.5698	2.0491×10^{-11}

TABLE 2. Numerical results of solving LWCP(4.1) ($\mu_0 = 0.0001$)

m	n	SNW			ZSNW		
		AIT	ACPU	ANH	AIT	ACPU	ANH
800	1000	2.0	0.7339	2.0675×10^{-13}	3.0	1.0046	1.9991×10^{-12}
		2.0	0.6844	1.3316×10^{-12}	3.0	0.9967	1.2123×10^{-11}
		2.0	0.6759	2.6964×10^{-12}	2.9	0.9847	3.4484×10^{-12}
1000	1500	2.0	1.6679	1.3429×10^{-11}	2.9	2.4122	3.8746×10^{-12}
		2.0	1.6756	3.5739×10^{-12}	2.7	2.2569	1.4781×10^{-11}
		2.0	1.6749	6.5931×10^{-12}	2.9	2.4047	7.8742×10^{-12}
1500	2000	2.1	4.1635	1.1445×10^{-11}	3.0	5.9572	7.0194×10^{-12}
		2.0	3.9387	5.5518×10^{-12}	2.9	5.8103	8.4229×10^{-12}
		2.0	3.9556	4.2912×10^{-12}	3.2	6.3675	1.8609×10^{-11}
2000	2500	2.0	7.5811	2.0139×10^{-11}	3.1	11.9231	5.9660×10^{-12}
		2.0	7.7082	4.7118×10^{-12}	3.1	11.6813	1.8023×10^{-11}
		2.0	7.5739	1.3293×10^{-12}	3.2	12.2772	4.3976×10^{-12}
2500	3000	2.0	13.2917	7.3664×10^{-12}	3.2	20.6451	1.0601×10^{-11}
		2.2	14.5042	9.1004×10^{-12}	3.5	22.8527	4.9231×10^{-12}
		2.0	13.2484	1.3913×10^{-11}	3.2	60.5698	6.4824×10^{-12}



(a) Average CPU time (b) Average number of iterations

FIGURE 3. Comparison curves of solving LWCP (4.1) ($\mu_0 = 0.01, m = \frac{n}{2}$)

4.2. **Algorithm 3.1 for solving obstacle problems with free boundaries.** In this subsection, to evaluate the performance of Algorithm 3.1 in solving obstacle problems with free boundaries, Algorithm 3.1 (TSN) in this paper was compared with algorithm 1 (RSN) in [9]. The experimental results are demonstrated in Table 3 and Figures 4 and 5, where each curve in Figure 4 is generated from the average of 10 random experiments.

Example 4.2. The matrix M and the vector q are as in [9], i.e.,

$$M = \begin{pmatrix} 2 & -1 & 0 & 0 & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & -1 & 2 & -1 \\ 0 & \cdots & 0 & 0 & -1 & 2 \end{pmatrix}, q = \begin{pmatrix} -2f(x_1) + f(x_2) \\ f(x_1) - 2f(x_2) + f(x_3) \\ \vdots \\ f(x_{n-3}) - 2f(x_{n-2}) + f(x_{n-1}) \\ f(x_{n-2}) - 2f(x_{n-1}) \end{pmatrix}.$$

TABLE 3. Numerical results in different dimensions

n	TSN			RSN		
	AIT	ACPU	ANH	AIT	ACPU	ANH
100	4.6	0.0083	0.1046×10^{-8}	6.6	0.3435	0.2350×10^{-8}
500	9.6	0.1662	0.1051×10^{-8}	14.4	0.8082	0.4001×10^{-8}
1000	15.2	1.0215	0.1055×10^{-8}	20.2	2.4224	0.5730×10^{-8}
1500	21.0	3.4034	0.1120×10^{-8}	28.0	8.7584	0.8060×10^{-8}
2000	26.4	8.1983	0.1123×10^{-8}	35.8	21.7515	0.1013×10^{-7}
2500	31.8	16.9556	0.1301×10^{-8}	42.6	42.4845	0.1158×10^{-7}
3000	36.4	29.3599	0.2084×10^{-8}	51.0	84.9590	0.1601×10^{-7}

4.3. **Algorithm 3.1 for solving Fisher market equilibrium problem.** In this subsection, we aim to evaluate the performance of Algorithm 3.1 in solving the Fisher market equilibrium

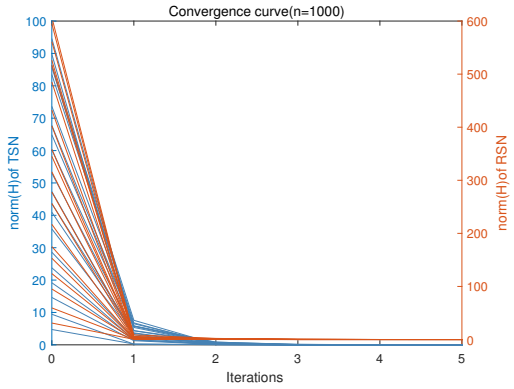


FIGURE 4. Convergence curve of $\left\|H\left(z^k\right)\right\|$ at the k-th iteration.

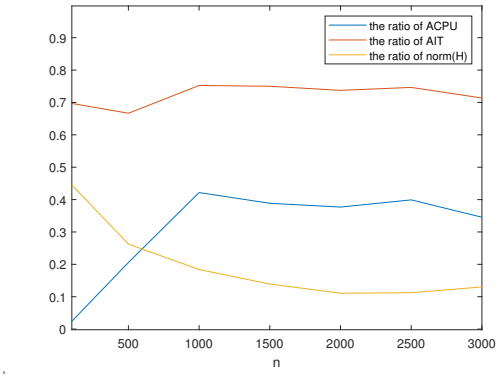


FIGURE 5. Performance ratio curves.

problem. We first solve a 4×6 dimensional LWCP (1.1) using Algorithm 3.1, and then compare the algorithm 3.1 in this paper and the LM Algorithm 3.1 [11]. The numerical results were shown in Tables 4 and 5.

Example 4.3. Solve a 4×6 dimensional LWCP (1.1). Let

$$A=\begin{bmatrix} 0 & 5 & -2 & 3 & 4 & 1\\ -1 & 3 & -2 & 5 & 1 & 3\\ 5 & -1 & 0 & 0 & 3 & 3\\ 4 & 0 & -1 & -2 & 3 & 2 \end{bmatrix}, b=(3,1,2,-1)^T, w=(1,4,6,4,6,6)^T.$$

We choose $x^0=(1,2,3,2,3,3)^T, y^0=(2,-1,-1,-1)^T$, and $s^0=(1,2,2,2,2,2)^T$. Let $\varepsilon=10^{-9}, \mu^0=0.1,\ \delta=0.3$, then the $\varepsilon-$ approximate solution is:

$$\begin{aligned}\tilde{x}&=(0.0656,1.2102,3.0236,0.2796,0.3989,0.5618)^T,\\ \tilde{s}&=(15.2470,3.3053,1.9844,14.3072,15.0399,10.6797)^T,\\ \tilde{y}&=(2.6610,-0.9209,7.2368,-5.4645)^T.\end{aligned}$$

where the iteration time is 0.0014 seconds and the number of iterations is 7.

Example 4.4. Solve a general Fisher market equilibrium problem consisting of n_c consumers and n_p producers. Let $z^0=\left(x^0,s^0,y^0\right)$, where x^0,s^0,y^0, be generated in the following way:

x^0 is a $n-$dimensional vector, with $n = n_c(n_p+1)$, and its first n_c coordinates formed by: $u_i^0 = \frac{1}{n_p}\sum_{k=1}^{n_p} u_{ik}, i = 1,2,\cdots,n_c$, and the remaining $n_c n_p$ coordinates consisting of the variables:

$$x_{ij}^0 = \frac{1}{n_p}, i = 1,2,\cdots,n_c, j = 1,2,\cdots,n_p.$$

$y^0 = \left(q^0, p^0\right)$, where, $q_i^0 = \frac{\hat{\beta}}{u_i^0}, i = 1,2,\cdots,n_c, p_j^0 = 2n_p\hat{\beta}, j = 1,2,\cdots,n_p, \hat{\beta} = \frac{n_p+1}{2n_p}\|w\|_\infty$.

$$s^0 = \left(v_1^0,\cdots,v_{n_c}^0, s_{11}^0,\cdots,s_{1n_p}^0, s_{21}^0,\cdots,s_{2n_p}^0,\cdots,s_{n_c1}^0,\cdots,s_{n_cn_p}^0\right)^T,$$

where, $v_i^0 = q_i^0, i = 1,\cdots,n_c, s_{ij}^0 = p_j^0 - q_i^0 u_{ij}, i = 1,\cdots,n_c, j = 1,\cdots,n_p$.

TABLE 4. Numerical results for Algorithm 3.1

n_c	n_c	m	n	AIT	ACPU	ANH
10	15	25	160	5.9	0.0119	6.6741×10^{-12}
20	30	50	620	5.8	0.3095	5.5798×10^{-12}
30	50	80	1530	6.0	2.6141	8.0510×10^{-12}
50	60	110	3050	4.0	10.955 7	1.5242×10^{-12}
60	80	140	4860	5.0	48.9692	4.2889×10^{-12}
80	90	170	7280	4.0	130.7075	5.3101×10^{-12}
90	90	180	8190	4.0	187.8999	5.5722×10^{-12}
90	100	180	9090	4.0	253.8758	6.0976×10^{-12}
90	10	200	9990	5.8	358.1430	8.8027×10^{-12}

TABLE 5. Numerical results for the LM Algorithm 3.1

n_c	n_c	m	n	AIT	ACPU	ANH
10	15	25	160	9.0	0.0310	1.5802×10^{-6}
10	20	30	210	13.9	0.1129	1.0649×10^{-6}
15	20	35	315	11.4	0.2291	9.6566×10^{-7}
20	25	45	520	12.6	0.8679	1.1736×10^{-6}
20	30	50	620	15.9	1.6755	3.4462×10^{-7}
25	30	55	775	15.4	2.7024	1.6511×10^{-6}
30	35	65	1080	17.5	7.3052	8.8871×10^{-7}
35	35	70	1260	16.3	9.9991	2.1409×10^{-6}
40	50	90	2040	28.9	64.1361	1.4174×10^{-6}

Tables 4 and 5 show the effectiveness of the smoothing Newton-type algorithm in solving the Fisher equilibrium problem, especially in solving the large-scale Fisher equilibrium problem. The smoothing Newton method is more effective.

Overall, for the random problems generated in numerical experiments, Algorithm 3.1 converges in very few iterations. The number of iterations varies slightly with the dimension of the tested problem.

5. Conclusions

Based on the idea of the smoothing Newton method, with the help of a new class of WCP functions $\phi_\tau^c(\mu,a,b)$, we proposed a new algorithm, Algorithm 3.1, for solving the LWCP (1.1). Numerical results show the significant advantage of the proposed algorithm in solving large-scale LWCP and the effectiveness of solving the free boundary of the obstacle problem.

Acknowledgements
The research of this paper was supported by Optimisation theory and algorithm research team (23kytdzd004), 2023 Annual Youth Scientific Research Fund project of Anhui Institute of Information Technology (No.23QNJJKJ017) and the General Programs for Young Teacher Cultivation of Educational Commission of Anhui Province of China (No.YQYB2023090).

References

[1] K. M. Anstreicher, Interior-point algorithms for a generalization of linear programming and weighted centring, Optim. Meth. Softw. 27 (2012) 605-612.
[2] S. Asadi, Z. Darvay, G. Lesaja, N. Mahdavi-Amiri, F. A. Potra, A full-Newton step interior-point method for monotone weighted linear complementarity problems, J. Optim. Theory Appl. 186 (2020) 864–878.
[3] Z. H. Huang, J. Y. Han, D. C. Xu, L. P. Zhang, The non-interior continuation methods for solving the P0 function nonlinear complementarity problem, Sci. China Ser. A Math. 44 (2001) 1107–1114.
[4] Z. H. Huang, J.Y. Han, Z. W. Chen, Predictor-corrector smoothing Newton method, based on a new smoothing function,for solving the NCP with a P0 function, J. Optim. Theory Appl. 117 (2003) 39-68.
[5] X. Q. Jiang, Y. J. Huang, A smoothing-type algorithm for solving monotone weighted complementarity problems, J. Phys.: Conf. Ser. (2020) 1642.
[6] Z. Y. Liu, J. Y. Tang, A new smoothing-type algorithm for nonlinear weighted complementarity problem, J. Appl. Math. Comput. 64 (2020) 215–226.
[7] F. A. Potra, Weighted complementarity problems-a new paradigm for computing equilibria, SIAM J. Optim. 22 (2012) 1634–1654.
[8] F. A. Potra, Sufficient weighted complementarity problems, Comput. Optim. Appl. 64 (2016) 467-488.
[9] S. P. Rui, C. X. Xu, An inexact smoothing method for solving obstacle and free boundary problems, Appl. Math. J. Chin. Univ. 27 (2012) 449-455.
[10] J. Y. Tang, J. C. Zhou, Quadratic convergence analysis of a nonmonotone Levenberg-Marquardt type method for the weighted nonlinear complementarity problem. Comput. Optim. Appl. 80 (2021) 213–244.
[11] P. J. Tian, Z. S. Yu, Y. Yuan, A smoothing Levenberg-Marquardt algorithm for linear weighted complementarity problem, AIMS Math. 8 (2023) 9862-9876.
[12] Z. Wan, H. H. Li, S. Huang, A smoothing inexact Newton method for nonlinear complementarity problems, Abstr. Appl. Anal. 2015 (2015) 731026.
[13] Z. S. Yu, Y. Qin, A cosh-based smoothing Newton method for P0 nonlinear complementarity problem, Nonlinear Anal. Real. 12 (2011) 875–884.
[14] Y. Y. Ye, A path to the arrow-debreu competitive market equilibrium, Math. Program. 111 (2008) 315–348.
[15] J. Zhang, A smoothing Newton algorithm for weighted linear complementarity problem, Optim. Lett. 10 (2016) 499–509.