Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 6(4): 3606–3623.
DOI:10.3934/math.2021215
Received: 08 November 2020
Accepted: 15 January 2021
Published: 22 January 2021

Research article

An active set quasi-Newton method with projection step for monotone nonlinear equations

Zhensheng Yu*and Peixin Li

College of Science, University of Shanghai for Science and Technology, Shanghai, 200093, China

* **Correspondence:** Email: zhsh-yu@163.com; Tel: +862155270430; Fax: +862155270430.

Abstract: In this paper, an active set quasi-Newton method for bound constrained nonlinear equation is proposed. By using this active set technique, we only need to solve a reduced dimension linear equation at each iteration to generate the search direction. The algorithm is a combination of the quasi-Newton method and projection method. Firstly we use the quasi-Newton step as the trial step and then use a projection technique to generate the next iteration point. Our key observation is that the algorithm generates a bounded iteration sequence automatically even if the bounds are equal to infinity and the global convergence is obtained in the sense that the whole sequence converges to the stationary point. The numerical tests show the efficiency of the algorithm.

Keywords: constrained nonlinear equations; active set; quasi-Newton; global convergence; projection
Mathematics Subject Classification: 65K05, 90C30

1. Introduction

In this paper, we consider the following bound constrained nonlinear systems of equations:

$$F(x) = 0, \ \ s.t. \ x \in \Omega, \tag{1.1}$$

where $F(x) = (F_1(x), F_2(x), \cdots, F_n(x))^T$, and $F_i : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear continuously differentiable function whose gradient is available. We denote by $F'(x) = (\nabla F_1(x), \nabla F_2(x), \cdots, \nabla F_n(x))^T$ the Jacobian matrix of F at a given point x. The set $\Omega \subseteq \mathbb{R}^n$ is defined as

$$\Omega := \{x \in \mathbb{R}^n | l_i \le x_i \le u_i, \forall\ i = 1, 2, \cdots, n\}$$

for some given lower and upper bounds satisfying $-\infty \le l_i < u_i \le +\infty$ for all $i = 1, 2, \cdots, n$.

The bound constrained nonlinear equation of the type (1.1) is an important problem in the practical problems. There are a couple of different mathematical programming problems like Karush-Kuhn-Tucker systems and complementarity problems can be reformulated as the problem (1.1), see [1–9]. On the other hand, in many cases, the function $F_i(x)$ is not always defined on the whole space $\mathbb{R}^n$, and one usually puts some suitable bounds on some or all of the variables.

The Newton type method is one of the most important numerical methods for problem (1.1) and many researchers are interested in this method [7, 10–14]. Given a current iterate $x^k \in \Omega$, the Newton method considers the least-squares solutions d^k of the following nonlinear constrained equation:

$$\min \frac{1}{2}\|F(x^k) + F'(x^k)d\|^2 \quad s.t.\ x^k + d \in \Omega. \tag{1.2}$$

We set the next iterate to be $x^{k+1} = x^k + d^k$ and call (1.2) the constrained Gauss-Newton method.

Another natural possibility is to consider solving the basic unconstrained Newton equation:

$$F(x^k) + F'(x^k)d = 0. \tag{1.3}$$

Denote the solution of (1.3) by d_N^k if it exists and then define x^{k+1} as the projection of $x^k + d_N^k$ onto Ω. This scheme can be called the projected Newton method.

On the other hand, there are many versions of the Newton type method, such as the constrained Levenberg-Marquardt method [6, 15, 16] usually used to solve the following subproblems:

$$\min \frac{1}{2}\|F(x^k) + F'(x^k)d\|^2 + \sigma\|d\|^2 \quad s.t.\ x^k + d \in \Omega, \tag{1.4}$$

where σ is a positive constant.

Along with constrained versions of the methods in question, one can also consider their projected variants. The projected Levenberg-Marquardt method has been proposed in [15] and its iteration consists of finding the solution d_{LM}^k of the unconstrained subproblem

$$\min \frac{1}{2}\|F(x^k) + F'(x^k)d\|^2 + \sigma\|d\|^2, \tag{1.5}$$

and then defines the next iterate x^{k+1} as the projection of $x^k + d_{LM}^k$ onto Ω.

The Newton iteration can be costly, since partial derivatives must be computed and the linear system (1.3) must be solved at every iteration. This fact motivates the development of quasi-Newton methods [10, 14, 17] which are defined as the generalizations of (1.3) given by

$$F(x^k) + B_k d = 0. \tag{1.6}$$

In quasi-Newton methods, the matrices B_k are intended to be approximations of $F'(x_k)$ and be updated by some quasi-Newton formulas. Another well known algorithm is the trust region type algorithm, for example, [3, 4, 9, 18–21].

Whether Newton method or quasi-Newton, one has to solve a linear system with full dimension, which will be expensive for large scale problems. To overcome this drawback, the active set methods are developed by many authors [7, 12, 13, 22]. Since only a reduced dimension linear system to be dealt with at each iteration, the active set Newton methods are more efficient than the full Newton method especially for large scale problems.

To prove global convergence of the method outlined above, one often assumes that the iteration sequence is contained in a bounded set. If l_i and u_i are bounded and the algorithm generates a feasible sequence, the assumption holds naturally. Otherwise, one often makes an assumption that the level set is bounded. For unconstrained nonlinear equations system, M.Solodov designed a Newton method with projection technique, the method can generate a bounded iteration sequence without additional assumption and the global convergence is obtained. Motivated by the idea of M.Solodov [23], in this paper, we extend the method to constrained equations (1.1). By using this active set strategy, we only need to solve a linear system with reduced dimension at each iteration. The algorithm generates a bounded sequence automatically even if l_i and u_i are infinite. We obtain the global convergence and give numerical tests to show the efficiency of the proposed algorithm.

The paper is organized as follows: In section 2, we describe our algorithm in detail. In section 3, we prove the global convergence of the proposed algorithm. Some numerical tests are shown in Section 4 and a conclusion is given in section 5. Throughout this paper, we use $\|\cdot\|$ to denote the 2−norm and E denotes the identity matrix.

2. Algorithm

We now describe our active set quasi-Newton method with projection technique in detail. To describe our algorithm, we introduce the definition of projection operator which is defined as a mapping form $\mathbb{R}^n$ to a nonempty closed convex subset Ω :

$$P_{\Omega}(x) = argmin\{\|y - x\| \mid y \in \Omega\}, \ \forall \ x \in \mathbb{R}^n. \tag{2.1}$$

A well-known property of the operator is that it is nonexpensive, namely,

$$\|P_{\Omega}(x) - P_{\Omega}(y)\| \leq \|x - y\|, \ \forall x, y \in \mathbb{R}^n. \tag{2.2}$$

Given a current iterate x^k, let

$$\delta_k := min\left\{\delta, c\sqrt{\|F(x^k)\|}\right\},$$

where δ and c are positive constants such that

$$\delta \leq \frac{1}{2}\min_{i=1,2,\cdots,n} |u_i - l_i|,$$

and define the index sets

$$\mathcal{A}_k := \left\{i \in \{1, 2, ..., n\} | x_i^k - l_k \leq \delta_k \ or \ u_i - x_i^k \leq \delta_k\right\},$$

$$\mathcal{I}_k := \{1, 2, ..., n\} \setminus \mathcal{A}_k = \{i | l_k + \delta_k < x_i^k < u_i - \delta_k\}.$$

The precise statement of our algorithm is as follows:

Algorithm 2.1: (Active Set-type Quasi-Newton Method)

(S.0) Choose a positive definite matrix B_k, $x^0 \in [l, u]$, choose parameters $\beta \in (0, 1)$, $\lambda \in (0, 1)$, $\delta > 0$, $c > 0$, $\varepsilon > 0$, $\mu_k > 0$, and $\rho_k \in [0, 1)$, and set $k := 0$.

(S.1) If $\|F(x^k)\| \leq \varepsilon$, stop.

(S.2) Try to compute a vector $d^k \in \mathbb{R}^n$ in the following way:
For $i \in \mathcal{A}_k$, set

$$d_i^k = -F_i(x^k)/(1-\rho_k)\mu_k. \tag{2.3}$$

For $i \in \mathcal{I}_k$, set solve the linear system

$$(B_k + \mu_k E_{\mathcal{I}_k})d_i^k = -F_i(x^k) + e_k, \tag{2.4}$$

where

$$\|e_k\| \leq \mu_k\rho_k\|d_i^k\|.$$

(S.3) Find $z^k = x^k + \alpha_k d^k$, where $\alpha_k = \beta^{m_k}$ with m_k being the smallest nonnegative integer m such that

$$-\langle F(x^k + \beta^m d^k), d^k\rangle \geq \lambda(1-\rho_k)\mu_k\|d^k\|^2. \tag{2.5}$$

(S.4) Compute

$$x^{k+1} = P_\Omega[x^k - \frac{\langle F(z^k), x^k - z^k\rangle}{\|F(z^k)\|^2}F(z^k)]. \tag{2.6}$$

(S.5) Update B_{k+1}, set $k := k+1$, go to (S.1).

Just as mentioned in [13], throughout this paper, we assume that the parameter $\delta > 0$ is chosen sufficiently small such that

$$\delta \leq \frac{1}{2}\min_{i=1,2,\cdots,n}|u_i - l_i|.$$

This implies that we cannot have $x_i^k - l_i \leq \delta_k$ and $u_i - x_i^k \leq \delta_k$ for the same index $i \in \mathcal{A}_k$.

Our algorithm is somewhat different from the traditional active set Newton method as described in [13], where the search step d^k in (S.2) is computed in the following formulas:

For $i \in \mathcal{A}_k$, set

$$d_i^k = \begin{cases} l_i - x_i^k & if \quad x_i^k - l_i \leq \delta_k, \\ u_i - x_i^k & if \quad u_i - x_i^k \leq \delta_k. \end{cases} \tag{2.7}$$

For $i \in \mathcal{I}_k$, solve the linear system

$$F'(x^k)_{\mathcal{I}_k\mathcal{I}_k} d_{\mathcal{I}_k} = -F(x^k)_{\mathcal{I}_k} - F'(x^k)_{\mathcal{I}_k\mathcal{A}_k} d_{\mathcal{A}_k}. \tag{2.8}$$

As described in [13], in order to understand the formula for the computation of the components d_i^k for $i \in \mathcal{I}_k$, note that, after a possible permutation of the rows and columns, [13] rewrite the standard (unconstrained) Newton equation $F'(x^k)d = -F(x^k)$ as

$$\begin{pmatrix} F'(x^k)_{\mathcal{I}_k\mathcal{I}_k} & F'(x^k)_{\mathcal{I}_k\mathcal{A}_k} \\ F'(x^k)_{\mathcal{A}_k\mathcal{I}_k} & F'(x^k)_{\mathcal{A}_k\mathcal{A}_k} \end{pmatrix} \begin{pmatrix} d_{\mathcal{I}_k} \\ d_{\mathcal{A}_k} \end{pmatrix} = -\begin{pmatrix} F(x^k)_{\mathcal{I}_k} \\ F(x^k)_{\mathcal{A}_k} \end{pmatrix} \tag{2.9}$$

Here we replace (2.7) by (2.3) and (2.8) by (2.4), the main proposal is to guarantee that the inequality (2.5) holds. On the other hand, we compute $d_{\mathcal{I}_k}$ by (2.4) instead of (2.8) which can be seen as an inexact Newton method.

The matrix B_k is updated by the well known rank two secant type formula updated by the well known BFGS formula

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, \tag{2.10}$$

where $y_k = F(x^{k+1}) - F(x^k)$ and $s_k = x^{k+1} - x^k$.

3. Convergence

In the section, we prove the global convergence of Algorithm 2.1, we make the following assumption.

Assumption:

(A1) The function $F(x)$ is Lipschitz continuous and monotone, i.e., there exists a positive constant L such that

$$\|F(x) - F(y)\| \le L\|x - y\| \tag{3.1}$$

and

$$\langle F(x) - F(y), (x - y)\rangle \ge 0,\ \forall\ x, y \in \Omega. \tag{3.2}$$

(A2) The sequence of matrices $\{B_k\}$ is positive definite and bounded, i.e., there exists a positive constant κ such that $\|B_k\| \le \kappa$ for all k.

We first show that the algorithm is feasible, i.e., there exists a positive m such that (2.5) holds.

Lemma 3.1. *The Algorithm 2.1 is well defined.*

Proof. We prove that the inequality (2.5) will hold with a nonnegative integer m. Suppose that for some index k this is not the case, which means, for all integer m, we have

$$-\langle F(x^k + \beta^m d^k), d^k\rangle < \lambda(1 - \rho_k)\mu_k\|d^k\|^2. \tag{3.3}$$

We further get

$$\begin{aligned} -\lim_{m\to\infty}\langle F(x^k + \beta^m d^k), d^k\rangle &= -\langle F(x^k), d^k\rangle \\ &= -\langle F_{\mathcal{A}_k}, d_{\mathcal{A}_k}\rangle - \langle F_{\mathcal{I}_k}, d_{\mathcal{I}_k}\rangle \\ &= \|F_{\mathcal{A}_k}\|^2/(1-\rho_k)\mu_k + \langle (B_k + \mu_k E_{\mathcal{I}_k})d_{\mathcal{I}_k} - e_k, d_{\mathcal{I}_k}\rangle \\ &\ge (1-\rho_k)\mu_k\|d_{\mathcal{A}_k}\|^2 + \mu_k\|d_{\mathcal{I}_k}\|^2 - \|e_k\|\|d_{\mathcal{I}_k}\| \\ &\ge (1-\rho_k)\mu_k\|d_{\mathcal{A}_k}\|^2 + (1-\rho_k)\mu_k\|d_{\mathcal{I}_k}\|^2 \\ &\ge (1-\rho_k)\mu_k\|d^k\|^2. \end{aligned} \tag{3.4}$$

Now we take the limit of both sides of (3.4) as $m \to \infty$, when (3.4) holds which implies that $\lambda \ge 1$, which contradicts the choice of $\lambda \in (0, 1)$. Hence we have that the inequality holds for some integer m, and the whole algorithm is well defined. □

In what follows, we assume that the algorithm generates an infinite iteration sequence. The following result shows that the algorithm generates a bounded sequence automatically and the proof is similar to Lemma 3.2 in [24] and we omit it here.

Theorem 3.2. *Suppose assumptions (A1) and (A2) hold, sequences* $\{x^k\}$ *and* $\{z^k\}$ *are generated by Algorithm 2.1, then* $\{x^k\}$ *and* $\{z^k\}$ *are both bounded. Furthermore, for any* $\overline{x}$ *such that* $F(\overline{x}) = 0$, *it holds that*

$$\|x^{k+1} - \overline{x}\|^2 \leq \|x^k - \overline{x}\|^2 - \|x^{k+1} - x^k\|^2. \tag{3.5}$$

$$\lim_{k\to\infty} \|x^k - z^k\| = 0. \tag{3.6}$$

and

$$\lim_{k\to\infty} \|x^{k+1} - x^k\| = 0. \tag{3.7}$$

Now we give the global convergence result of the Algorithm 2.1.

Lemma 3.3. *Let* $\{x^k\}$ *be generated by Algorithm 2.1, assume Assumption (A1) and (A2) hold, and there exists constants* $0 < \underline{\rho} < \overline{\rho} < 1$, *and* $\underline{\mu} < \overline{\mu}$ *such that* $\underline{\rho} \leq \rho_k \leq \overline{\rho}$, *and* $\underline{\mu} \leq \mu_k \leq \overline{\mu}$. *Then* $\{x^k\}$ *converges to some* x^* *such that* $F(x^*) = 0$.

Proof. By the inequality (2.5), we have

$$\langle F(z^k), x^k - z^k\rangle = -\alpha_k\langle F(z^k), d^k\rangle \geq \lambda(1 - \rho_k)\mu_k\alpha_k\|d^k\|^2. \tag{3.8}$$

By the definition of d^k, we have that

$$\|d_{\mathcal{A}_k}\| = \|F_{\mathcal{A}_k}/(1 - \rho_k)\mu_k\| \leq \|F_{\mathcal{A}_k}\|/(1 - \overline{\rho})\underline{\mu}. \tag{3.9}$$

and

$$\begin{aligned} \|F_{\mathcal{I}_k}\| &\geq \|(B_k + \mu_k E_{\mathcal{I}_k})d_{\mathcal{I}_k}\| - \|e_k\| \\ &\geq (1 - \rho_k)\mu_k\|d_{\mathcal{I}_k}\| \\ &\geq (1 - \overline{\rho})\underline{\mu}\|d_{\mathcal{I}_k}\|. \end{aligned} \tag{3.10}$$

Combining (3.9) and (3.10), we can assume that there exists a positive constant c_1 such that

$$\|F(x^k)\| \geq c_1\|d^k\|. \tag{3.11}$$

On the other hand, the definition of d^k also gives that

$$\|F_{\mathcal{A}_k}\| = \|(1 - \rho_k)\mu_k d_{\mathcal{A}_k}\| \leq (1 - \underline{\rho})\overline{\mu}\|d_{\mathcal{A}_k}\|. \tag{3.12}$$

From (2.4) and Assumption (A2), we have

$$\begin{aligned} \|F_{\mathcal{I}_k}\| &\leq \|(B_k + \mu_k E_{\mathcal{I}_k})d_{\mathcal{I}_k}\| + \|e_k\| \\ &\leq (\kappa + \mu_k + \rho_k\mu_k)\|d_{\mathcal{I}_k}\| \\ &\leq [\kappa + (1 + \overline{\rho})\overline{\mu}]\|d_{\mathcal{I}_k}\|. \end{aligned} \tag{3.13}$$

Combining (3.12) and (3.13), we can assume that there exists a positive constant c_2 such that

$$\|F(x^k)\| \le c_2\|d^k\|. \tag{3.14}$$

Now by (3.8), we obtain

$$\|F(z^k)\|\|x^k - z^k\| \ge \langle F(z^k), x^k - z^k\rangle \ge \lambda(1-\overline{\rho})\underline{\mu}\alpha_k\|d^k\|^2. \tag{3.15}$$

By the continuity of $F(x)$, the bound of sequence $\{z^k\}$ and (3.6), we have

$$\lim_{k\to\infty} \alpha_k\|d^k\|^2 = 0. \tag{3.16}$$

We consider the two possible cases:

$$\liminf_{k\to\infty} \|F(x^k)\| = 0 \ \ and \ \ \liminf_{k\to\infty} \|F(x^k)\| > 0. \tag{3.17}$$

In the first case, the continuity of F and the boundness of $\{x^k\}$ imply that the sequence $\{x^k\}$ has some accumulation point x^* such that $F(x^*) = 0$. Since $\overline{x}$ was an arbitrary solution, we can choose $\overline{x} = x^*$ in (3.5). The sequence $\{\|x^k - x^*\|\}$ converges and since x^* is an accumulation point of $\{x^k\}$, it must be the case that $\{x^k\}$ converges to x^*.

Now consider the second case. From (3.14), we have

$$\liminf_{k\to\infty} \|d^k\| > 0.$$

Hence by (3.16), we have

$$\liminf_{k\to\infty} \alpha_k = 0.$$

(The following proof is very similar to the last part in Theorem 2.1 [23], for complement, we list it here.) By the step rule, we have the inequality (2.5) is not valid for the value β^{m_k-1}, i.e.,

$$-\langle F(x^k + \beta^{m_k-1}d^k), d^k\rangle < \lambda(1-\rho_k)\mu_k\|d^k\|^2 \tag{3.18}$$

Let $k \to \infty$, we get

$$-\langle F(x^*), d^*\rangle < \lambda(1-\rho^*)\mu^*\|d^*\|^2, \tag{3.19}$$

Here x^*, d^*, ρ^*, μ^* denote the limits of the corresponding sequence respectively. On the other hand, by (3.4), we get

$$-\langle F(x^*), d^*\rangle \ge (1-\rho^*)\mu^*\|d^*\|^2, \tag{3.20}$$

that contradicts the choice for $\lambda \in (0,1)$. Hence the case $\liminf_{k\to\infty} \|F(x^k)\|$ is impossible.

This completes the proof. □

4. Numerical experiments

In this section, we demonstrate the numerical performance of Algorithm 2.1 (AQN) and its computational advantage by comparing with the modified Kanzow [13] ACTN method (denoted as AKP) and the classical Quasi-Newton method with project (denoted as CQN). All presented codes are written in MATLAB2019 and run on a PC with 3.30GHz CPU processor, 4.0GB memory and Windows 8 operation system.

We consider ten problems with dimension n=1000,5000,10000. We use six different starting points, that is:

$$\begin{aligned} x_1 &= (0.1, 0.1, ..., 0.1)^T, \\ x_2 &= (\frac{1}{2}, \frac{1}{2^2}, ..., \frac{1}{2^n})^T, \\ x_3 &= (2, 2, ..., 2)^T, \\ x_4 &= (1, \frac{1}{2}, ..., \frac{1}{n})^T, \\ x_5 &= (1, 1 - \frac{1}{2}, ..., 1 - \frac{1}{n})^T, \\ x_6 &= rand(0, 1). \end{aligned}$$

After several parameter selection experiments, we select the initial parameters that can make the three algorithms have better performance :

$$\beta = 0.5,\ \lambda = 0.6,\ \delta = 0.001,\ c = 1,\ \mu_k = 0.5,\ \varepsilon = 10^{-6},\ \rho_k = 0.3.$$

Set the terminating criterion for the iteration process as $\|F(x_k)\| \leq 10^{-6}$. The problems are listed as follows.

Problem 1. [25]

$$F_i(x) = e^{x_i} - 1,\ \ i = 1, 2, ..., n, \tag{4.1}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 2. [25]

$$\begin{aligned} F_1(x) &= e^{x_1} - 1, \\ F_i(x) &= e^{x_i} + x_{i-1} - 1,\ \ i = 2, ..., n, \end{aligned} \tag{4.2}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 3. [25]

$$\begin{aligned} F_1(x) &= 2x_1 - x_2 + e^{x_1} - 1, \\ F_i(x) &= -x_{i-1} + 2x_i - x_{i+1} + e^{x_i} - 1,\ \ i = 2, ..., n-1, \\ F_n(x) &= -x_{n-1} + 2x_n + e^{x_n} - 1, \end{aligned} \tag{4.3}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 4. [25]

$$\begin{aligned} F_1(x) &= \frac{5}{2}x_1 + x_2 - 1, \\ F_i(x) &= x_{i-1} + \frac{5}{2}x_i + x_{i+1} - 1,\ \ i = 2, ..., n-1, \\ F_n(x) &= x_{n-1} + \frac{5}{2}x_n - 1, \end{aligned} \tag{4.4}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 5. [25]

$$F_i(x) = e^{x_i} + \frac{3}{2}sin(2x_i) - 1, \quad i = 1, 2, ..., n, \tag{4.5}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 6. [25]

$$\begin{aligned} F_1(x) &= x_1 - e^{cos(h(x_1+x_2))}, \\ F_i(x) &= x_i - e^{cos(h(x_{i-1}+x_i+x_{i+1}))}, \quad i = 2, ..., n-1, \\ F_n(x) &= x_n - e^{cos(h(x_{n-1}+x_n))}, \end{aligned} \tag{4.6}$$

where $h = \frac{1}{n+1}$ and $\Omega = \mathbb{R}^n_+$.

Problem 7. [25]

$$F_i(x) = 2x_i - sin|x_i|, \quad i = 1, 2, ..., n, \tag{4.7}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 8. [26]

$$F_i(x) = 2\sqrt{2}x_i - 1, \quad i = 1, 2, ..., n, \tag{4.8}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 9. [26]

$$F_i(x) = e^{x_i^2} + 3sinx_icosx_i - 1, \quad i = 1, 2, ..., n, \tag{4.9}$$

where $\Omega = \mathbb{R}^n_+$.

Problem 10. [24]

$$F_i(x) = x_i - sin(|x_i - 1|), \quad i = 1, 2, ..., n, \tag{4.10}$$

where $\Omega = \mathbb{R}^n_+$.

Comprehensive results of our numerical experiment are presented in Tables 1–10. The columns of the presented tables have the following definitions:

IP: the initial points.
DIM: the dimension of the problem.
NI: the iterative number.
NF: the iterative number of function evaluation.
CPU: the CPU time in seconds when the algorithm terminate.
NORM: the final norm equation.

We denote result by '−' whenever the number of iterations exceeds 500 or the terminating criterion has not been satisfied. Among these results, none of the three methods were able to solve Problem 9 when initial point is $x_3 = (2, 2, ..., 2)^T$. Therefore, Table 9 does not include the case when the initial point is x_3. Meanwhile, in the drawing process, when the result was denoted by '−', its NI, NF, CPU and NORM are counted as ∞.

The performance of the three methods was evaluated using the performance profile which is presented by Dolan and Moré [27]. We comparing three methods with the same problem, dimension and initial point in an experiment, and recoding information of interest such as NI, NF, CPU and NORM.

We denote the set of problems as $\mathcal{P}$ and the set of methods as $\mathcal{M}$. For example, for each problem p and method m, we define

$$t_{p,m} = CPU\ time\ required\ to\ solve\ problem\ p\ by\ method\ m. \tag{4.11}$$

Compare the performance on problem p by method m with the best performance by any method on this problem, that is, we use the performance ratio

$$r_{p,m} = \frac{t_{p,m}}{min\{t_{p,m} : m \in \mathcal{M}\}}. \tag{4.12}$$

We assume that a parameter $R \geq r_{p,m}$ for all p, m is chosen, and $r_{p,m} = R$ if and only if method m does not solve problem p. If method m can solve problem p successfully, we obtain an overall assessment of the performance between these methods. It can be described as follows:

$$\rho_m(\tau) := \frac{1}{n_p} size\{p \in \mathcal{P} : r_{p,m} \leq \tau\}$$

where n_p represents the number of elements in set $\mathcal{P}$, then $\rho_m(\tau)$ is the probability for method $m \in \mathcal{M}$ that a performance ratio $r_{p,m}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio. The function ρ_m is the (cumulative) distribution function for the performance ratio.

The performance profile $\rho_m : \mathbb{R} \longmapsto [0, 1]$ for a method is a nondecreasing, piecewise constant function, continuous from the right at each breakpoint. We are interested in methods with a high probability of solve success, then we need only to compare the values of $\rho_m(\tau)$ for all of the methods and choose the method with the largest, there means that we need to find which method's function ρ_m first rearch the line $\rho_m(\tau) = 1$. In the same way, we can obtain the performance profile with respect to NI, NF and NORM.

As can be seen from the information in the Table 1–10, AQN has more stable solving performance and can solve more problems, such as what AKP cannot solve: x_3 and x_5 of Problem 3 when n = 10000; x_2 of Problem 3 when $n = 5000, 10000$; x_2 of Problem 5 when $n = 1000, 5000, 10000$; x_3 of Problem 5 when $n = 5000, 10000$; x_2 of Problem 8 when $n = 5000, 10000$; x_2 of Problem 10 when $n = 5000, 10000$. Compared with CQN, from Figure 1 and Figure 2, we can see that AQN reaches the line that $\rho_m(1) = 1$ before CQN, which demonstrates AQN has a faster solution time(CPU) and the solution results of the final norm equation(NORM) are more accurate. Although from Figure 3 and Figure 4, there shows the iterative number of CQN is less than AQN, in practice, we pay more attention to the advantage of solution time. To sum up, AQN has a more stable and faster solving performance.

Table 1. Numerical results for Problem 1.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	30	61	3.656	8.63E-07	22	45	9.419	6.82E-07	22	45	1.365	7.86E-07
	5000	32	65	123.832	7.97E-07	23	47	252.343	7.62E-07	23	47	94.233	8.78E-07
	10000	33	67	707.123	7.25E-07	24	49	1204.500	5.39E-07	24	49	801.382	6.21E-07
X2	1000	28	57	5.164	9.12E-07	19	39	8.252	7.03E-07	18	37	1.256	8.22E-07
	5000	28	57	143.052	9.12E-07	19	39	199.614	7.03E-07	18	37	83.818	8.22E-07
	10000	28	57	779.363	9.12E-07	19	39	913.557	7.03E-07	18	37	609.232	8.22E-07
X3	1000	34	69	5.087	7.49E-07	26	53	10.410	5.92E-07	27	55	2.115	5.01E-07
	5000	36	73	161.879	6.92E-07	27	55	288.600	6.62E-07	28	57	128.120	5.60E-07
	10000	36	73	853.569	9.79E-07	27	55	1328.800	9.36E-07	28	57	920.577	7.92E-07
X4	1000	34	69	7.029	8.51E-07	20	41	7.855	9.44E-07	22	45	2.102	5.97E-07
	5000	35	71	208.186	6.74E-07	20	41	210.414	9.44E-07	22	45	100.805	5.97E-07
	10000	35	71	1078.500	9.74E-07	20	41	966.249	9.44E-07	22	45	721.224	5.97E-07
X5	1000	34	69	5.434	9.47E-07	24	49	9.660	8.28E-07	25	51	1.683	8.35E-07
	5000	36	73	169.626	8.76E-07	25	51	264.494	9.26E-07	26	53	118.489	9.34E-07
	10000	37	75	907.287	7.96E-07	26	53	1266.100	6.55E-07	27	55	910.632	6.61E-07
X6	1000	34	69	5.624	9.32E-07	24	49	9.475	8.07E-07	25	51	1.702	8.48E-07
	5000	36	73	169.128	8.70E-07	25	51	264.354	9.25E-07	26	53	119.666	9.25E-07
	10000	37	75	912.983	7.91E-07	26	53	1283.200	6.54E-07	27	55	916.142	6.61E-07

Table 2. Numerical results for Problem 2.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	60	121	10.393	9.69E-07	40	81	16.089	9.12E-07	45	91	8.107	7.83E-07
	5000	59	119	304.457	9.61E-07	40	81	427.665	8.75E-07	44	89	198.837	9.90E-07
	10000	59	119	1494.000	8.76E-07	40	81	2015.480	8.62E-07	44	89	1590.250	9.57E-07
X2	1000	72	145	15.477	8.99E-07	46	93	18.481	9.61E-07	47	95	3.669	9.48E-07
	5000	72	145	439.560	8.99E-07	46	93	503.471	9.61E-07	47	95	216.065	9.48E-07
	10000	72	145	2181.400	8.99E-07	46	93	2308.030	9.61E-07	47	95	1613.963	9.48E-07
X3	1000	76	153	16.627	8.33E-07	48	97	19.281	7.93E-07	40	81	2.786	8.10E-07
	5000	74	149	451.509	9.91E-07	48	97	519.860	7.65E-07	38	77	176.960	9.97E-07
	10000	74	149	2251.200	9.34E-07	48	97	2434.370	7.55E-07	38	77	1279.398	9.46E-07
X4	1000	75	151	16.529	9.82E-07	48	97	19.329	8.67E-07	50	101	3.162	9.43E-07
	5000	75	151	472.387	9.78E-07	48	97	519.555	8.67E-07	50	101	229.550	9.43E-07
	10000	75	151	2346.100	9.76E-07	48	97	2408.623	8.67E-07	50	101	1657.715	9.43E-07
X5	1000	74	149	15.728	9.03E-07	47	95	18.880	8.61E-07	50	101	3.553	8.40E-07
	5000	73	147	448.860	9.98E-07	47	95	509.295	8.29E-07	48	97	218.293	7.60E-07
	10000	73	147	2209.300	8.51E-07	47	95	2358.238	8.17E-07	48	97	1607.768	9.27E-07
X6	1000	90	181	19.039	8.23E-07	56	113	22.527	8.90E-07	57	115	3.537	9.40E-07
	5000	94	189	563.484	8.70E-07	59	119	636.860	8.24E-07	62	125	285.941	7.93E-07
	10000	95	191	2835.600	1.00E-06	60	121	3044.774	8.69E-07	62	125	2105.600	9.51E-07

Table 3. Numerical results for Problem 3.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	93	187	19.750	9.30E-07	81	163	33.507	8.00E-07	44	89	3.338	8.25E-07
	5000	92	185	562.208	9.53E-07	84	169	918.799	8.97E-07	56	93	214.676	8.47E-07
	10000	99	199	3150.498	9.36E-07	87	175	4584.800	7.85e-07	48	97	1611.711	9.30E-07
X2	1000	76	153	16.189	9.89E-07	61	123	25.470	8.65E-07	36	73	2.281	8.46E-07
	5000	76	153	463.761	9.89E-07	61	123	667.653	8.65E-07	36	73	164.605	8.46E-07
	10000	76	153	2325.900	9.89E-07	61	123	3055.546	8.65E-07	36	73	1190.508	8.46E-07
X3	1000	121	243	31.734	9.47E-07	94	189	38.858	9.03E-07	57	115	3.676	6.93E-07
	5000	106	213	697.003	8.87E-07	100	201	1093.100	8.52E-07	63	127	288.233	6.82E-07
	10000	112	225	3761.469	9.95E-07	101	203	5055.167	8.92E-07	-	-	-	-
X4	1000	87	175	19.633	9.94E-07	72	145	29.727	8.19E-07	41	83	3.892	7.88E-07
	5000	88	177	569.847	9.97E-07	72	145	791.423	8.20E-07	41	83	186.841	7.94E-07
	10000	88	177	2817.300	9.81E-07	72	145	3597.463	8.20E-07	41	83	1368.882	7.94E-07
X5	1000	109	219	25.967	9.32E-07	89	179	37.222	8.30E-07	53	107	3.714	9.84E-07
	5000	116	233	788.855	8.80E-07	93	187	1035.100	8.54E-07	57	115	263.580	8.93E-07
	10000	117	235	3934.500	9.97E-07	95	191	4746.173	7.82E-07	-	-	-	-
X6	1000	107	215	25.975	9.97E-07	95	191	39.617	7.52E-07	53	107	3.527	9.08E-07
	5000	116	233	783.719	8.42E-07	98	197	1090.700	9.45E-07	55	111	255.271	9.85E-07
	10000	120	241	4148.100	9.15E-07	101	203	5045.600	7.46E-07	58	117	1955.841	7.26E-07

Table 4. Numerical results for Problem 4.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	96	193	39.937	9.05E-07	96	193	43.074	9.05E-07	62	125	6.113	8.47E-07
	5000	97	195	1080.500	7.73E-07	97	195	1085.200	7.73E-07	64	129	292.586	8.46E-07
	10000	96	193	4915.800	7.99E-07	96	193	4809.600	7.99E-07	66	133	2193.345	9.51E-07
X2	1000	96	193	39.898	7.99E-07	94	189	38.846	8.52E-07	76	153	4.878	8.00E-07
	5000	94	189	1049.800	8.95E-07	93	187	1015.018	9.44E-07	-	-	-	-
	10000	94	189	4742.816	9.23E-07	92	185	4602.733	9.80E-07	-	-	-	-
X3	1000	75	151	31.182	8.40E-07	75	151	30.961	8.40E-07	66	133	4.018	9.87E-07
	5000	73	147	802.568	9.18E-07	73	147	796.533	9.18E-07	73	147	337.588	6.90E-07
	10000	75	151	3808.900	7.44E-07	75	151	3748.947	7.44E-07	75	151	2478.365	8.36E-07
X4	1000	90	181	37.247	9.77E-07	93	187	38.923	8.42E-07	66	133	4.171	9.87E-07
	5000	93	187	1025.700	8.81E-07	92	185	1005.456	9.24E-07	76	153	348.842	9.56E-07
	10000	93	187	4705.600	9.01E-07	92	185	4622.813	9.17E-07	79	159	2634.719	8.62E-07
X5	1000	91	183	37.749	8.51E-07	93	187	38.567	8.93E-07	70	141	4.794	9.67E-07
	5000	91	183	1002.900	8.29E-07	92	185	1004.607	9.91E-07	75	151	348.323	9.69E-07
	10000	110	221	5628.200	7.64E-07	94	189	4710.946	7.83E-07	75	151	2492.402	9.22E-07
X6	1000	142	285	58.851	8.81E-07	143	287	60.753	9.66E-07	79	159	7.068	9.92E-07
	5000	151	303	1669.800	8.98E-07	151	303	1653.700	8.47E-07	85	171	393.990	7.41E-07
	10000	154	309	7856.500	9.98E-07	154	309	7653.971	9.64E-07	86	173	2869.170	8.31E-07

Table 5. Numerical results for Problem 5.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	11	23	2.351	7.06E-07	11	23	5.609	7.06E-07	30	61	7.252	7.27E-07
	5000	12	25	69.706	1.55E-07	12	25	67.909	1.55E-07	30	61	138.354	9.03E-07
	10000	12	25	352.637	2.19E-07	12	25	348.586	2.19E-07	35	71	1172.793	6.11E-07
X2	1000	12	25	2.984	9.73E-07	12	25	2.678	6.04E-07	61	123	4.507	7.65E-07
	5000	13	27	80.610	2.13E-07	13	27	79.346	1.33E-07	-	-	-	-
	10000	13	27	405.754	3.02E-07	13	27	398.015	1.88E-07	-	-	-	-
X3	1000	12	25	2.017	4.61E-07	12	25	1.936	4.61E-07	-	-	-	-
	5000	13	27	61.161	1.82E-07	13	27	60.556	1.82E-07	-	-	-	-
	10000	13	27	323.124	2.58E-07	13	27	319.935	2.58E-07	-	-	-	-
X4	1000	16	33	4.417	1.21E-07	13	27	3.303	3.32E-07	49	99	3.172	9.84E-07
	5000	14	29	93.207	1.19E-07	13	27	88.261	7.42E-07	65	131	298.947	9.85E-07
	10000	14	29	454.358	1.67E-07	13	27	397.148	1.87E-07	63	127	2097.393	7.91E-07
X5	1000	17	35	4.812	2.09E-07	13	27	2.720	1.52E-07	59	119	4.057	7.74E-07
	5000	17	35	133.735	4.68E-07	13	27	79.443	3.40E-07	60	121	277.731	9.35E-07
	10000	17	35	659.338	6.62E-07	13	27	396.954	4.81E-07	62	125	2093.674	7.82E-07
X6	1000	13	27	2.824	1.63E-07	13	27	2.754	1.51E-07	59	119	4.104	8.52E-07
	5000	17	35	134.016	3.29E-07	13	27	79.329	3.39E-07	60	121	297.163	8.30E-07
	10000	17	35	651.481	6.60E-07	13	27	396.469	4.82E-07	59	119	1976.682	8.91E-07

Table 6. Numerical results for Problem 6.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	49	99	20.419	7.13E-07	42	85	19.977	8.74E-07	28	57	7.443	9.06E-07
	5000	50	101	549.836	9.05E-07	43	87	484.930	9.65E-07	30	61	136.093	6.79E-07
	10000	51	103	2623.200	8.85E-07	44	89	2252.977	8.71E-07	30	61	1008.218	9.29E-07
X2	1000	48	97	20.223	6.99E-07	42	85	17.454	9.11E-07	29	59	1.842	9.19E-07
	5000	51	103	560.601	7.23E-07	44	89	491.333	6.51E-07	32	65	149.412	5.94E-07
	10000	51	103	2607.600	8.64E-07	44	89	2251.287	9.07E-07	35	71	1172.464	5.23E-07
X3	1000	39	79	16.445	7.27E-07	39	79	16.669	7.27E-07	27	55	1.868	7.22E-07
	5000	40	81	440.412	7.64E-07	40	81	439.550	7.64E-07	27	55	123.541	8.95E-07
	10000	41	83	2082.900	7.31E-07	41	83	2039.955	7.31E-07	28	57	934.110	5.24E-07
X4	1000	48	97	20.053	7.21E-07	42	85	17.886	9.11E-07	28	57	3.648	9.14E-07
	5000	49	99	540.145	7.74E-07	44	89	490.692	8.50E-07	31	63	148.501	6.77E-07
	10000	51	103	2582.600	7.23E-07	44	89	2264.235	6.61E-07	31	63	1213.002	7.78E-07
X5	1000	43	87	18.038	7.87E-07	42	85	17.649	9.23E-07	28	57	3.153	5.72E-07
	5000	44	89	483.664	9.58E-07	43	87	489.753	9.44E-07	29	59	151.122	8.58E-07
	10000	45	91	2279.000	9.44E-07	45	91	2273.992	5.95E-07	30	61	1025.754	5.67E-07
X6	1000	44	89	18.394	9.10E-07	42	85	17.362	9.04E-07	28	57	2.144	9.01E-07
	5000	50	101	550.710	8.75E-07	43	87	491.334	9.04E-07	29	59	132.839	8.87E-07
	10000	51	103	2599.800	8.75E-07	44	89	2265.719	9.95E-07	30	61	1022.843	7.92E-07

Table 7. Numerical results for Problem 7.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	30	61	3.984	9.52E-07	22	45	9.309	7.52E-07	22	45	10.004	7.55E-07
	5000	32	65	124.936	8.79E-07	23	47	257.722	8.41E-07	23	47	103.523	8.44E-07
	10000	33	67	696.347	7.99E-07	24	49	1205.700	5.95E-07	24	49	850.861	5.97E-07
X2	1000	30	61	8.726	8.29E-07	20	41	10.117	5.27E-07	20	41	4.586	5.67E-07
	5000	30	61	180.592	8.29E-07	20	41	225.843	5.27E-07	20	41	95.335	5.67E-07
	10000	30	61	886.356	8.29E-07	20	41	1006.400	5.27E-07	20	41	663.478	5.67E-07
X3	1000	35	71	5.263	6.87E-07	26	53	10.435	8.45E-07	25	51	1.791	9.09E-07
	5000	36	73	160.033	9.88E-07	27	55	286.293	9.45E-07	27	55	123.108	5.08E-07
	10000	37	75	862.174	8.98E-07	28	57	1366.000	6.68E-07	27	55	894.793	7.19E-07
X4	1000	34	69	7.383	8.53E-07	21	43	8.777	5.35E-07	21	43	1.328	6.68E-07
	5000	36	73	224.577	7.58E-07	21	43	230.923	5.35E-07	21	43	96.340	6.68E-07
	10000	35	71	886.356	8.29E-07	21	43	1051.442	5.35E-07	21	43	693.062	6.68E-07
X5	1000	35	71	5.908	7.47E-07	24	49	10.020	9.58E-07	25	51	1.524	5.81E-07
	5000	37	75	180.603	6.89E-07	26	53	284.448	5.36E-07	26	53	119.156	6.50E-07
	10000	37	75	944.657	9.75E-07	26	53	1318.300	7.58E-07	26	53	861.209	9.19E-07
X6	1000	35	71	5.919	7.63E-07	24	49	10.713	9.75E-07	25	51	6.592	5.83E-07
	5000	37	75	180.124	6.98E-07	26	53	284.553	5.40E-07	26	53	119.424	6.50E-07
	10000	37	75	934.966	9.82E-07	26	53	1310.800	7.60E-07	26	53	913.455	9.17E-07

Table 8. Numerical results for Problem 8.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	14	29	5.802	7.75E-07	14	29	5.782	7.75E-07	24	49	1.915	7.92E-07
	5000	15	31	165.321	5.08E-07	15	31	164.148	5.08E-07	27	55	124.365	8.29E-07
	10000	15	31	756.748	7.18E-07	15	31	751.927	7.18E-07	28	57	933.741	6.67E-07
X2	1000	15	31	7.203	8.97E-07	15	31	6.145	3.16E-07	40	81	2.560	7.75E-07
	5000	16	33	176.418	5.87E-07	15	31	164.099	7.08E-07	-	-	-	-
	10000	16	33	816.974	8.31E-07	16	33	802.254	2.93E-07	-	-	-	-
X3	1000	16	33	6.622	4.32E-07	16	33	6.548	4.32E-07	27	55	1.773	8.73E-07
	5000	16	33	177.685	9.66E-07	16	33	174.795	9.66E-07	31	63	143.292	6.08E-07
	10000	17	35	862.303	4.00E-07	17	35	851.777	4.00E-07	31	63	1043.391	8.67E-07
X4	1000	18	37	7.461	3.10E-07	15	31	6.472	3.12E-07	28	57	1.923	8.04E-07
	5000	16	33	178.233	6.05E-07	15	31	163.715	7.05E-07	41	83	189.831	9.66E-07
	10000	16	33	814.770	8.43E-07	15	31	750.481	9.99E-07	42	85	1412.292	7.38E-07
X5	1000	18	37	7.470	2.97E-07	14	29	5.710	9.89E-07	28	57	2.999	7.11E-07
	5000	18	37	197.939	6.65E-07	15	31	163.855	6.48E-07	29	59	132.820	7.97E-07
	10000	18	37	913.455	9.40E-07	15	31	751.494	9.17E-07	30	61	1030.916	5.65E-07
X6	1000	18	37	7.583	2.97E-07	15	31	6.453	2.96E-07	28	57	1.812	7.13E-07
	5000	18	37	199.224	4.30E-07	15	31	163.758	6.42E-07	29	59	132.158	8.65E-07
	10000	19	39	962.107	2.93E-07	15	31	756.324	9.11E-07	29	59	966.698	6.08E-07

Table 9. Numerical results for Problem 9.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	38	77	3.055	8.24E-07	12	25	4.959	5.05E-07	23	47	3.828	5.40E-07
	5000	40	81	113.409	9.88E-07	13	27	143.222	2.82E-07	24	49	109.381	6.04E-07
	10000	42	85	712.896	7.49E-07	13	27	651.289	3.99E-07	24	49	802.195	8.54E-07
X2	1000	37	75	4.855	8.16E-07	11	23	4.708	3.37E-07	21	43	1.338	7.99E-07
	5000	37	75	153.719	8.16E-07	11	23	120.945	3.37E-07	21	43	94.816	7.99E-07
	10000	37	75	835.797	8.16E-07	11	23	555.333	3.37E-07	21	43	699.885	7.99E-07
X4	1000	44	89	6.384	8.49E-07	12	25	6.552	5.24E-07	25	51	1.755	5.99E-07
	5000	45	91	187.263	9.32E-07	12	25	132.641	5.24E-07	25	51	114.659	6.45E-07
	10000	47	95	1195.000	9.22E-07	12	25	607.4983	5.24E-07	25	51	834.238	6.56E-07
X5	1000	43	87	5.098	9.20E-07	14	29	6.816	3.27E-07	31	63	3.127	6.74E-07
	5000	46	93	171.163	8.08E-07	14	29	156.394	7.32E-07	32	65	146.725	7.64E-07
	10000	47	95	967.804	8.34E-07	15	31	764.984	2.59E-07	33	67	1127.503	5.43E-07
X6	1000	43	87	5.095	9.54E-07	14	29	7.562	3.29E-07	31	63	2.035	6.66E-07
	5000	46	93	170.719	8.49E-07	14	29	157.247	7.28E-07	32	65	152.282	7.68E-07
	10000	47	95	924.105	8.18E-07	15	31	763.091	2.57E-06	33	67	1103.404	5.36E-07

Table 10. Numerical results for Problem 10.

IP	DIM	AQN				CQN				AKP			
		NI	NF	CPU	NORM	NI	NF	CPU	NORM	NI	NF	CPU	NORM
X1	1000	28	57	11.625	5.43E-07	28	57	11.710	5.43E-07	25	51	2.226	6.68E-07
	5000	29	59	323.043	6.46E-07	29	59	316.952	6.46E-07	26	53	118.148	7.47E-07
	10000	29	59	1476.900	9.13E-07	29	59	1452.863	9.13E-07	27	55	895.844	5.28E-07
X2	1000	25	51	10.099	8.50E-07	25	51	10.236	8.50E-07	30	61	1.998	5.30E-07
	5000	27	55	290.044	5.38E-07	27	55	285.598	5.38E-07	-	-	-	-
	10000	27	55	1328.900	7.61E-07	27	55	1312.528	7.61E-07	-	-	-	-
X3	1000	28	57	11.242	7.69E-07	28	57	11.411	7.69E-07	26	53	1.740	9.50E-07
	5000	29	59	311.758	9.14E-07	29	59	310.210	9.14E-07	28	57	126.392	5.31E-07
	10000	30	61	1482.600	6.88E-07	30	61	1469.212	6.88E-07	28	57	931.480	7.51E-07
X4	1000	25	51	10.069	8.57E-07	25	51	10.064	8.57E-07	28	57	1.864	5.98E-07
	5000	27	55	290.624	5.38E-07	27	55	287.511	5.38E-07	29	59	131.109	7.84E-07
	10000	27	55	1331.600	7.61E-07	27	55	1316.705	7.61E-07	30	61	999.476	6.92E-07
X5	1000	27	55	11.246	6.81E-07	27	55	11.420	6.81E-07	27	55	1.838	9.05E-07
	5000	28	57	309.842	8.10E-07	28	57	309.207	8.10E-07	29	59	135.589	8.78E-07
	10000	29	59	1472.000	6.10E-07	29	59	1471.858	6.10E-07	30	61	1043.962	6.19E-07
X6	1000	27	55	11.266	6.71E-07	27	55	11.374	6.79E-07	27	55	2.818	6.93E-07
	5000	28	57	310.428	8.08E-07	28	57	308.040	8.07E-07	29	59	147.317	8.16E-07
	10000	29	59	1460.100	6.10E-07	29	59	1459.546	6.08E-07	30	61	1177.273	5.73E-07

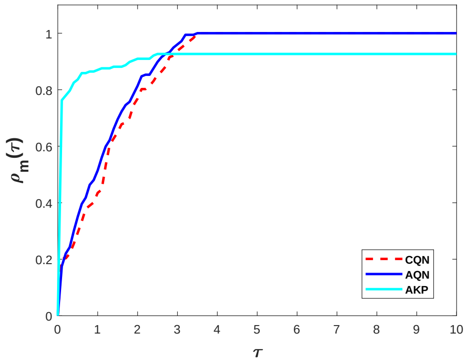

Figure 1. Performance profiles based on CPU time.

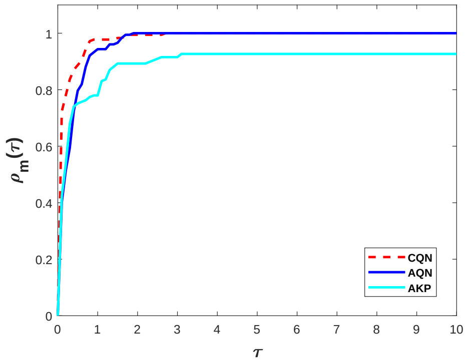

Figure 2. Performance profiles based on the final norm equation.

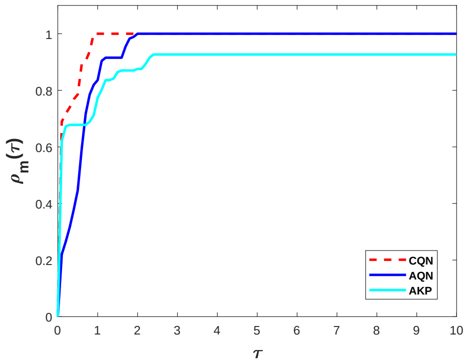

Figure 3. Performance profiles based on number of iterations.

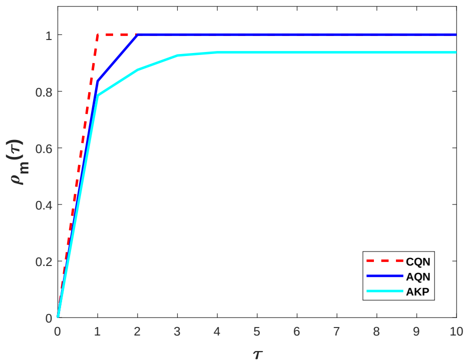

Figure 4. Performance profiles based on number of function evaluation.

5. Conclusions

In this paper, we propose an active set quasi-Newton method for the solution of optimization problem with bound constraints. The implementation of the method uses the quasi-Newton step as a trial step and the project step as the correction step. By using active set technique, we only need to solve a reduced dimension linear equation at each iteration to generate the search direction. We prove that the generated sequence is bounded automatically and obtain the global convergence of the proposed algorithm. Meanwhile, compared with other algorithms, our method has the most stable performance. There are some questions that need studying in the near future. Firstly, it is possible to get the global convergence of the proposed algorithm without the assumption of the positive definite of the matrix B_k. Secondly, how to get the local convergence of the proposed algorithm especially under some weak condition such as the local error bound condition needs further studying.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. R. Andreani, A. Friedlander, Bound constrained smooth optimization for solving variational inequalities and related problems, *Ann. Oper. Res.,* **116** (2002), 179–198.
2. D. P. Bertsekas, Projected Newton methods for optimization problems with simple constraints, *SIAM J. Control. Optim.,* **20** (1982), 221–246.
3. S. Bellavia, M. Macconi, B. Morini, An affine scaling trust-region approach to bound-constrained nonlinear systems, *Appl. Numer. Math.,* **44** (2003), 257–280.
4. S. Bellavia, M. Macconi, B. Morini, STRSCNE: A scaled trust-region solver for constrained nonlinear equations, *Comput. Optim. Appl.,* **28** (2004), 31–50.
5. S. Bellavia, B. Morini, S. Pieraccini, Constrained dogleg methods for nonlinear systems with simple bounds, *Comput. Optim. Appl.,* **53** (2012), 771–794.
6. F. Facchinei, A. Fischer, C. Kanzow, J. M. Peng, A simply constrained optimization reformulation of KKT systems arising from variational inequalities, *Appl. Math. Optim.,* **40** (1999), 19–37.
7. C. Kanzow, H. D. Qi, A QP-free constrained Newton-type method for variational inequality problems, *Math. Program.,* **85** (1997), 81–106.
8. C. Kanzow, Some equation-based methods for the nonlinear complementarity problem, *Optim. Methods Softw.,* **3** (2007), 327–340.
9. M. Ulbrich, Nonmonotone trust-region methods for bound-constrained semismooth equations with applications to nonlinear mixed complementarity problems, *SIAM J. Optim.,* **11** (2001), 889–917.
10. E. G. Birgina, N. Krejić, J. M. Martínez, Globally convergent inexact quasi-Newton methods for solving nonlinear systems, *Numer. Algorithms,* **32** (2003), 249–260.
11. A. Fischer, A. F. Izmailov, M. V. Solodov, Local attractors of Newton-type methods for constrained equations and complementarity problems with nonisolated solutions, *J. Optim. Theory. Appl.,* **180** (2019), 140–169.

12. F. Facchinei, J. Júdice, J. Soares, An active set Newton algorithm for large-scale nonlinear programs with box constraints, *SIAM J. Optim.,* **8** (1998), 158–186.

13. C. Kanzow, *An active set-type Newton method for constrained nonlinear systems*, M.C. Ferris, O.L. Mangasarian, (Eds.), Boston: Springer, 2001.

14. L. K. Schubert, Modification of a quasi-Newton method for nonlinear equations with a sparse Jacobian, *Math. Comp.,* **24** (1970), 27–30.

15. C. Kanzow, N. Yamashita, M. Fukushima, Levenberg-Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints, *J. Comput. Appl. Math.,* **172** (2004), 375–397.

16. J. L. Zhang, On the convergence properties of the Levenberg-Marquardt method, *Optimization,* **52** (2003), 739–756.

17. J. M. Martińez, Practical quasi-Newton methods for solving nonlinear systems, *J. Comput. Appl. Math.,* **124** (2000), 97–122.

18. S. Bellavia, S. Pieraccini, On affine-scaling inexact dogleg methods for bound-constrained nonlinear systems, *Optim. Methods Softw.,* **30** (2015), 276–300.

19. C. J. Lin, J. Moré, Newton's method for large bound-constrained optimization problems, *SIAM J. Optim.,* **9** (1999), 1100–1127.

20. R. H. Byrd, J. Nocedal, R. B. Schnabel, Representation of quasi-Newton matrices and their use in limited memory methods, *Math. Program.,* **63** (1994), 129–156.

21. L. Marini, B. Morini, M. Porcelli, Quasi-Newton methods for constrained nonlinear systems: complexity analysis and applications, *Comput. Optim. Appl.,* **71** (2018), 147–170.

22. H. D. Qi, L. Q. Qi, D. F. Sun, Solving KKT systems via the trust region and the conjugate gradient methods, *SIAM J. Optim.,* **14** (2004), 439–463.

23. M. V. Solodov, B. F. Svaiter, *A globally convergent inexact Newton method for systems of monotone equations*, M. Fukushima, L. Qi (Eds.), Boston: Springer, 1998.

24. Z. S. Yu, J. Lin, J. Sun, et.al. Spectral gradient projection method for mononote nonlinear equations with convex constraints, *Appl. Numer. Math.,* **59** (2009), 2416–2423.

25. A. M. Awwal, L. Wang, P. Kumam, H. Mohammad, W. Watthayu, A projection Hestenes-Stiefel method with spectral parameter for nonlinear monotone equations and signal processing, *Math. Comput. Appl.,* **25** (2020), 27.

26. A. H. Ibrahim, P. Kumam, A. B. Abubakar, W. Jirakitpuwapat, J. Abubakar, A hybrid conjugate gradient algorithm for constrained monotone equations with application in compressive sensing, *Heliyon,* **6** (2020), e03466.

27. E. D. Dolan, J. J.Moré, Benchmarking optimization software with performance profiles, *Math. Program.,* **91** (2002), 201–213.