J. Appl. Math. & Informatics Vol. **33**(2015), No. 3 - 4, pp. 387 - 399
http://dx.doi.org/10.14317/jami.2015.387

SMOOTHING APPROXIMATION TO l_1 EXACT PENALTY FUNCTION FOR CONSTRAINED OPTIMIZATION PROBLEMS†

NGUYEN THANH BINH

Abstract. In this paper, a new smoothing approximation to the l_1 exact penalty function for constrained optimization problems (COP) is presented. It is shown that an optimal solution to the smoothing penalty optimization problem is an approximate optimal solution to the original optimization problem. Based on the smoothing penalty function, an algorithm is presented to solve COP, with its convergence under some conditions proved. Numerical examples illustrate that this algorithm is efficient in solving COP.

AMS Mathematics Subject Classification : 90C30, 65K05.
Key words and phrases : Constrained optimization, exact penalty function, smoothing method, approximate optimal solution.

1. Introduction

Consider the following COP:

$$\begin{aligned}(\mathrm{P})\ \ &\min\ f(x)\\ &\text{s.t. } g_i(x) \le 0,\ i = 1, 2, \ldots, m,\end{aligned}$$

where $f, g_i : R^n \to R,\ i \in I = \{1, 2, \ldots, m\}$ are continuously differentiable functions and $X_0 = \{x \in R^n \mid g_i(x) \le 0,\ i = 1, 2, \ldots, m\}$ is the feasible set to (P).

To solve (P), many exact penalty function methods have been introduced in literatures, see, [1, 3, 4, 5, 7, 13, 25]. In 1967, Zangwill [25] first the classical l_1 exact penalty function as follows:

$$F_1(x, \rho) = f(x) + \rho \sum_{i=1}^{m} \max\{g_i(x), 0\}, \tag{1}$$

Received October 30, 2014. Revised February 16, 2015. Accepted February 18, 2015.
†This work is supported by National Natural Science Foundation of China (Grant No.11371242).

where $\rho > 0$ is a penalty parameter. Obviously, it is not a smooth function. In many studies, another popular penalty function for (P) is defined as:

$$F_2(x, \rho) = f(x) + \rho \sum_{i=1}^{m} \max\{g_i(x), 0\}^2, \tag{2}$$

which is called l_2 penalty function. Although $F_2(x, \rho)$ is continuously differentiable, it is not an exact penalty function.

In recent years, the lower order penalty function

$$F^k(x, \rho) = f(x) + \rho \sum_{i=1}^{m} [\max\{g_i(x), 0\}]^k, \qquad k \in (0, 1) \tag{3}$$

has been introduced and investigated in [10, 11, 18]. Recently, Huang and Yang [6, 23] and Rubinov et al. [14, 15, 16] discussed a nonlinear Lagrangian penalty function,

$$F_k(x, \rho) = \left[f(x)^k + \rho \sum_{i=1}^{m} \max\{g_i(x), 0\}^k \right]^{\frac{1}{k}} \tag{4}$$

for some $k \in (0, +\infty)$.

It is noted that two penalty functions (3) and (4) ($0 < k \leq 1$) are exact, but not smooth, which makes certain efficient methods (e.g., Newton methods) not applicable. Therefore, the smoothing methods for these exact penalty functions (1), (3) or (4) ($0 < k \leq 1$) attracts much attention, see, [2, 8, 9, 10, 11, 12, 18, 19, 20, 21, 22, 24, 26]. Chen et al. [2] introduced a smooth function to approximate the classical l_1 penalty function by integrating the sigmoid function $1/(1 + e^{-\alpha t})$. Lian [8] and Wu et al. [19] proposed a smoothing approximation to l_1 exact penalty function for inequality constrained optimization. Pinar et al. [12] also proposed a smoothing approximation to l_1 exact penalty function and an ϵ-optimal minimum can be obtained by solving the smoothed penalty problem. Xu et al. [21] discussed a second-order differentiability smoothing to the classical l_1 exact penalty function for constrained optimization problems.

In this paper, we aim to smooth l_1 exact penalty function of the form (1). First, we define a function $p_\epsilon(t)$ as follows:

$$p_\epsilon(t) = \begin{cases} 0 & \text{if } t < 0, \\ \dfrac{t^2}{4\epsilon} & \text{if } 0 \leq t < \epsilon, \\ t + \dfrac{1}{2}\epsilon e^{-\frac{t}{\epsilon}+1} - \dfrac{5\epsilon}{4} & \text{if } t \geq \epsilon. \end{cases}$$

It is easy to prove that $p_\epsilon(t)$ is continuously differentiable on R. Using $p_\epsilon(t)$ as the smoothing function, a new smoothing approximation to l_1 exact penalty function is obtained, based on the smoothed penalty function obtained thereafter an algorithm for solving COP is given in this paper.

The rest of this paper is organized as follows. In Section 2, we introduce a smoothing function for the classical l_1 exact penalty function and some fundamental properties of the smoothing function. In Section 3, the algorithm based on the smoothed penalty function is proposed and its global convergence is presented, with some numerical examples given. Finally, conclusions are given in Section 4.

2. A smoothing penalty function

Let $p(t) = \max\{t, 0\}$. Then, the penalty function (1) is turned into

$$F(x,\rho) = f(x) + \rho \sum_{i=1}^{m} p\left(g_i(x)\right), \tag{5}$$

where $\rho > 0$. The corresponding penalty optimization problem to $F(x,\rho)$ is defined as

$$(P_\rho) \quad \min \ F(x,\rho), \quad \text{s.t.} \quad x \in R^n.$$

In order to $p(t)$, we define function $p_\epsilon(t) : R^1 \to R^1$ as

$$p_\epsilon(t) = \begin{cases} 0 & \text{if } t < 0, \\ \dfrac{t^2}{4\epsilon} & \text{if } 0 \le t < \epsilon, \\ t + \dfrac{1}{2}\epsilon e^{-\frac{t}{\epsilon}+1} - \dfrac{5\epsilon}{4} & \text{if } t \ge \epsilon, \end{cases}$$

where $\epsilon > 0$ is a smoothing parameter.

Remark 2.1. Obviously, $p_\epsilon(t)$ has the following attractive properties: $p_\epsilon(t)$ is continuously differentiable on R and $\lim_{\epsilon \to 0} p_\epsilon(t) = p(t)$.

Figure 1 shows the behavior of $p(t)$ (represented by the real line), $p_{0.5}(t)$ (represented by the dot line), $p_{0.1}(t)$ (represented by the broken and dot line), $p_{0.001}(t)$ (represented by the broken line).

Consider the penalty function for (P) given by

$$F_\epsilon(x,\rho) = f(x) + \rho \sum_{i=1}^{m} p_\epsilon\left(g_i(x)\right). \tag{6}$$

Clearly, $F_\epsilon(x,\rho)$ is continuously differentiable on R^n. Applying (6), the following penalty problem for (P) is obtained

$$(NP_{\rho,\epsilon}) \quad \min \ F_\epsilon(x,\rho), \qquad \text{s.t. } x \in R^n.$$

Now, the relationship between (P_ρ) and $(NP_{\rho,\epsilon})$ is studied.

Lemma 2.1. *For any given $x \in R^n$, $\epsilon > 0$ and $\rho > 0$, we have*

$$0 \le F(x,\rho) - F_\epsilon(x,\rho) \le \frac{5m\rho\epsilon}{4}. \tag{7}$$

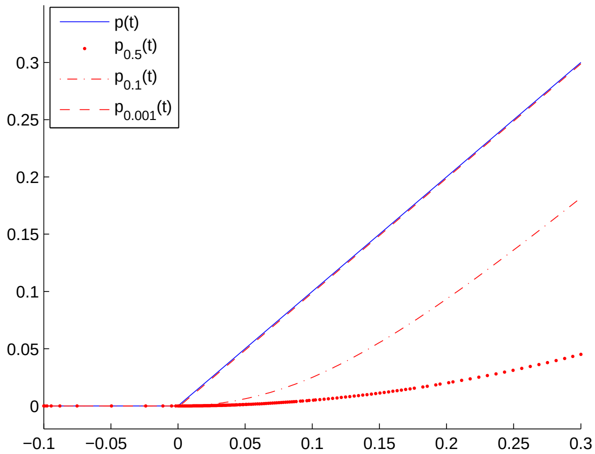

FIGURE 1. The behavior of $p(t)$ and $p_\epsilon(t)$.

Proof. For $x \in R^n$ and $i \in I$, by the definition of $p_\epsilon(t)$, we have

$$p\left(g_i(x)\right) - p_\epsilon\left(g_i(x)\right) = \begin{cases} 0 & \text{if } g_i(x) < 0, \\ g_i(x) - \dfrac{g_i(x)^2}{4\epsilon} & \text{if } 0 \le g_i(x) < \epsilon, \\ \dfrac{5\epsilon}{4} - \dfrac{1}{2}\epsilon e^{-\frac{g_i(x)}{\epsilon}+1} & \text{if } g_i(x) \ge \epsilon. \end{cases}$$

That is,

$$0 \le p\left(g_i(x)\right) - p_\epsilon\left(g_i(x)\right) \le \frac{5\epsilon}{4}, \qquad i = 1, 2, \ldots, m.$$

Thus,

$$0 \le \sum_{i=1}^{m} p\left(g_i(x)\right) - \sum_{i=1}^{m} p_\epsilon\left(g_i(x)\right) \le \frac{5m\epsilon}{4},$$

which implies

$$0 \le \rho \sum_{i=1}^{m} p\left(g_i(x)\right) - \rho \sum_{i=1}^{m} p_\epsilon\left(g_i(x)\right) \le \frac{5m\rho\epsilon}{4}.$$

Therefore,

$$0 \le \left\{ f(x) + \rho \sum_{i=1}^{m} p\left(g_i(x)\right) \right\} - \left\{ f(x) + \rho \sum_{i=1}^{m} p_\epsilon\left(g_i(x)\right) \right\} \le \frac{5m\rho\epsilon}{4},$$

that is,

$$0 \le F(x, \rho) - F_\epsilon(x, \rho) \le \frac{5m\rho\epsilon}{4}.$$

The proof completes. □

A direct result of Lemma 2.1 is given as follows.

Corollary 2.2. *Let $\{\epsilon_j\} \to 0$ be a sequence of positive numbers and assume x^j is a solution to $(NP_{\rho,\epsilon})$ for some given $\rho > 0$. Let x' be an accumulation point of the sequence $\{x^j\}$. Then x' is an optimal solution to (P_ρ).*

Definition 2.3. For $\epsilon > 0$, a point $x_\epsilon \in R^n$ is called ϵ-feasible solution to (P) if $g_i(x_\epsilon) \le \epsilon,\ \forall i \in I$.

Definition 2.4. For $\epsilon > 0$, a point $x_\epsilon \in X_0$ is called ϵ-approximate optimal solution to (P) if

$$|f^* - f(x_\epsilon)| \le \epsilon,$$

where f^* is the optimal objective value of (P).

Theorem 2.5. *Let x^* be an optimal solution of problem (P_ρ) and x' be an optimal solution to $(NP_{\rho,\epsilon})$ for some $\rho > 0$ and $\epsilon > 0$. Then,*

$$0 \le F(x^*, \rho) - F_\epsilon(x', \rho) \le \frac{5m\rho\epsilon}{4}. \tag{8}$$

Proof. From Lemma 2.1, for $\rho > 0$, we have that

$$0 \le F(x^*, \rho) - F_\epsilon(x^*, \rho) \le \frac{5m\rho\epsilon}{4},$$
$$0 \le F(x', \rho) - F_\epsilon(x', \rho) \le \frac{5m\rho\epsilon}{4}.$$

Under the assumption that x^* is an optimal solution to (P_ρ) and x' is an optimal solution to $(NP_{\rho,\epsilon})$, we get

$$F(x^*, \rho) \le F(x', \rho),$$
$$F_\epsilon(x', \rho) \le F_\epsilon(x^*, \rho).$$

Therefore, we obtain that

$$\begin{aligned} 0 &\le F(x^*, \rho) - F_\epsilon(x^*, \rho) \le F(x^*, \rho) - F_\epsilon(x', \rho) \\ &\le F(x', \rho) - F_\epsilon(x', \rho) \le \frac{5m\rho\epsilon}{4}. \end{aligned}$$

That is,

$$0 \le F(x^*, \rho) - F_\epsilon(x', \rho) \le \frac{5m\rho\epsilon}{4}.$$

This completes the proof. □

Theorem 2.5 show that an approximate solution to $(NP_{\rho,\epsilon})$ is also an approximate solution to (P_ρ) when the error ϵ is sufficiently small.

Lemma 2.6 ([20])**.** *Suppose that x^* is an optimal solution to (P_ρ). If x^* is feasible to (P), then it is an optimal solution to (P).*

Theorem 2.7. *Suppose that x^* satisfies the conditions in Lemma 2.6 and x' be an optimal solution to $(NP_{\rho,\epsilon})$ for some $\rho > 0$ and $\epsilon > 0$. If x' is ϵ-feasible to (P). Then,*

$$0 \leq f(x^*) - f(x') \leq \frac{3m\rho\epsilon}{2}, \tag{9}$$

that is, x' is an approximate optimal solution to (P).

Proof. Since x' is ϵ-feasible to (P), it follows that

$$\sum_{i=1}^{m} p_\epsilon\left(g_i(x')\right) \leq \frac{m\epsilon}{4}.$$

As x^* is a feasible solution to (P), we have

$$\sum_{i=1}^{m} p\left(g_i(x^*)\right) = 0.$$

By Theorem 2.5, we get

$$0 \leq \left\{ f(x^*) + \rho \sum_{i=1}^{m} p\left(g_i(x^*)\right) \right\} - \left\{ f(x') + \rho \sum_{i=1}^{m} p_\epsilon\left(g_i(x')\right) \right\} \leq \frac{5m\rho\epsilon}{4}.$$

Thus,

$$\rho \sum_{i=1}^{m} p_\epsilon\left(g_i(x')\right) \leq f(x^*) - f(x') \leq \rho \sum_{i=1}^{m} p_\epsilon\left(g_i(x')\right) + \frac{5m\rho\epsilon}{4}.$$

That is,

$$0 \leq f(x^*) - f(x') \leq \frac{3m\rho\epsilon}{2}.$$

By Lemma 2.6, x^* is actually an optimal solution to (P). Thus x' is an approximate optimal solution to (P). This completes the proof. □

By Theorem 2.7, an optimal solution to $(NP_{\rho,\epsilon})$ is an approximate optimal solution to (P) if it is ϵ-feasible to (P). Therefore, we can obtain an approximately optimal solution to (P) by solving $(NP_{\rho,\epsilon})$ under some mild conditions.

3. Algorithm and numerical examples

In this section, using the smoothed penalty function $F_\epsilon(x, \rho)$, we propose an algorithm to solve COP, defined as Algorithm 3.1.

Algorithm 3.1

Step 1: Choose $x^0,\ \epsilon > 0,\ \epsilon_0 > 0,\ \rho_0 > 0,\ 0 < \eta < 1$ and $N > 1$, let $j = 0$ and go to Step 2.

Step 2: Use x^j as the starting point to solve

$$(NP_{\rho_j,\epsilon_j}) \qquad \min_{x \in R^n} \ F_{\epsilon_j}(x, \rho_j) = f(x) + \rho_j \sum_{i=1}^{m} p_{\epsilon_j}\left(g_i(x)\right).$$

Let x^{j+1} be the optimal solution obtained (x^{j+1} is obtained by a quasi-Newton method).

Step 3: If x^{j+1} is ϵ-feasible to (P), then stop and we have obtained an approximate solution x^{j+1} of (P). Otherwise, let $\rho_{j+1} = N\rho_j,\ \epsilon_{j+1} = \eta\epsilon_j$ and $j = j+1$, then go to Step 2.

Remark 3.1. In this Algorithm 3.1, as $N > 1$ and $0 < \eta < 1$, the sequence $\{\epsilon_j\} \to 0$ $(j \to +\infty)$ and the sequence $\{\rho_j\} \to +\infty$ $(j \to +\infty)$.

In practice, it is difficult to compute $x^{j+1} \in arg \min\limits_{x\in R^n} F_{\epsilon_j}(x, \rho_j)$. We generally look for the local minimizer or stationary point of $F_{\epsilon_j}(x, \rho_j)$ by computing x^{j+1} such that $\nabla F_{\epsilon_j}(x, \rho_j) = 0$. For $x \in R^n$, we define

$$\begin{aligned} I^0(x) &= \{i \mid g_i(x) < 0,\ i \in I\}, \\ I_\epsilon^+(x) &= \{i \mid g_i(x) \geq \epsilon,\ i \in I\}, \\ I_\epsilon^-(x) &= \{i \mid 0 \leq g_i(x) < \epsilon,\ i \in I\}. \end{aligned}$$

Then, the following result is obtained.

Theorem 3.1. *Assume that* $\lim\limits_{\|x\|\to+\infty} f(x) = +\infty$. *Let* $\{x^j\}$ *be the sequence generated by Algorithm* 3.1. *Suppose that the sequence* $\{F_{\epsilon_j}(x^j, \rho_j)\}$ *is bounded. Then* $\{x^j\}$ *is bounded and any limit point* x^* *of* $\{x^j\}$ *is feasible to (P), and satisfies*

$$\lambda \nabla f(x^*) + \sum_{i\in I} \mu_i \nabla g_i(x^*) = 0, \tag{10}$$

where $\lambda \geq 0$ *and* $\mu_i \geq 0,\ i = 1, 2, \ldots, m$.

Proof. First, we will prove that $\{x^j\}$ is bounded. Note that

$$F_{\epsilon_j}(x^j, \rho_j) = f(x^j) + \rho_j \sum_{i=1}^m p_{\epsilon_j}\left(g_i(x^j)\right), \qquad j = 0, 1, 2, \ldots, \tag{11}$$

and by the definition of $p_\epsilon(t)$, we have

$$\sum_{i=1}^m p_{\epsilon_j}\left(g_i(x^j)\right) \geq 0. \tag{12}$$

Suppose to the contrary that $\{x^j\}$ is unbounded. Without loss of generality, we assume that $\|x^j\| \to +\infty$ as $j \to +\infty$. Then, $\lim\limits_{j\to+\infty} f(x^j) = +\infty$ and from (11) and (12), we have

$$F_{\epsilon_j}(x^j, \rho_j) \geq f(x^j) \to +\infty,\ \rho_j > 0,\ j = 0, 1, 2, \ldots,$$

which results in a contradiction since the sequence $\{F_{\epsilon_j}(x^j, \rho_j)\}$ is bounded. Thus $\{x^j\}$ is bounded.

We show next that any limit point x^* of $\{x^j\}$ is feasible to (P). Without loss of generality, we assume that $\lim_{j\to+\infty} x^j = x^*$. Suppose that x^* is not feasible to (P). Then there exits some $i \in I$ such that $g_i(x^*) \geq \alpha > 0$. Note that

$$F_{\epsilon_j}(x^j,\rho_j) = f(x^j) + \rho_j \sum_{i\in I^+_{\epsilon_j}(x^j)} \left(g_i(x^j) + \frac{1}{2}\epsilon_j e^{-\frac{g_i(x^j)}{\epsilon_j}+1} - \frac{5\epsilon_j}{4} \right) + \rho_j \sum_{i\in I^-_{\epsilon_j}(x^j)} \frac{g_i(x^j)^2}{4\epsilon_j}. \tag{13}$$

If $j \to +\infty$, then for any sufficiently large j, the set $\{i \mid g_i(x^j) \geq \alpha\}$ is not empty. Because I is finite, then there exists an $i_0 \in I$ that satisfies $g_{i_0}(x^j) \geq \alpha$. If $j \to +\infty, \rho_j \to +\infty, \epsilon_j \to 0$, it follows from (13) that $F_{\epsilon_j}(x^j,\rho_j) \to +\infty$, which contradicts the assumption that $\{F_{\epsilon_j}(x^j,\rho_j)\}$ is bounded. Therefore, x^* is feasible to (P).

Finally, we show that (10) holds. By Step 2 in Algorithm 3.1, $\nabla F_{\epsilon_j}(x^j,\rho_j) = 0$, that is

$$\nabla f(x^j) + \rho_j \sum_{i\in I^+_{\epsilon_j}(x^j)} \left(1 - \frac{1}{2} e^{-\frac{g_i(x^j)}{\epsilon_j}+1}\right) \nabla g_i(x^j) + \rho_j \sum_{i\in I^-_{\epsilon_j}(x^j)} \frac{1}{2\epsilon_j} g_i(x^j)\nabla g_i(x^j) = 0. \tag{14}$$

For $j = 1, 2, \dots,$ let

$$\gamma_j = 1 + \sum_{i\in I^+_{\epsilon_j}(x^j)} \rho_j \left(1 - \frac{1}{2} e^{-\frac{g_i(x^j)}{\epsilon_j}+1}\right) + \sum_{i\in I^-_{\epsilon_j}(x^j)} \frac{\rho_j}{2\epsilon_j} g_i(x^j). \tag{15}$$

Then $\gamma_j > 1$. From (14), we have

$$\frac{1}{\gamma_j}\nabla f(x^j) + \sum_{i\in I^+_{\epsilon_j}(x^j)} \frac{\rho_j \left(1 - \frac{1}{2} e^{-\frac{g_i(x^j)}{\epsilon_j}+1}\right)}{\gamma_j} \nabla g_i(x^j) + \sum_{i\in I^-_{\epsilon_j}(x^j)} \frac{\rho_j \epsilon_j^{-1}}{2\gamma_j} g_i(x^j)\nabla g_i(x^j) = 0. \tag{16}$$

Let

$$\lambda^j = \frac{1}{\gamma_j},$$

$$\mu_i^j = \frac{\rho_j \left(1 - \frac{1}{2} e^{-\frac{g_i(x^j)}{\epsilon_j}+1}\right)}{\gamma_j}, \qquad i \in I^+_{\epsilon_j}(x^j),$$

$$\mu_i^j = \frac{\rho_j \epsilon_j^{-1}}{2\gamma_j} g_i(x^j), \qquad i \in I^-_{\epsilon_j}(x^j),$$

$$\mu_i^j = 0, \qquad i \in I \setminus \left(I^+_{\epsilon_j}(x^j) \cup I^-_{\epsilon_j}(x^j) \right).$$

Then we have

$$\lambda^j + \sum_{i\in I} \mu_i^j = 1, \ \forall j, \tag{17}$$

$$\mu_i^j \geq 0, \ i \in I, \ \forall j.$$

When $j \to \infty$, we have that $\lambda^j \to \lambda \geq 0$, $\mu_i^j \to \mu_i \geq 0$, $\forall i \in I$. By (16) and (17), as $j \to +\infty$, we have

$$\lambda \nabla f(x^*) + \sum_{i\in I} \mu_i \nabla g_i(x^*) = 0,$$

$$\lambda + \sum_{i\in I} \mu_i = 1.$$

For $i \in I^0(x^*)$, as $j \to +\infty$, we get $\mu_i^j \to 0$. Therefore, $\mu_i = 0$, $\forall i \in I^0(x^*)$. So, (10) holds, and this completes the proof. □

Theorem 3.1 points out that the sequence $\{x^j\}$ generated by Algorithm 3.1 may converge to a K-T point to (P) under some conditions.

Now, we will solve some COP with Algorithm 3.1 on MATLAB. In each example, we let $\epsilon = 10^{-6}$, then it is expected to get an ϵ-solution to (P) with Algorithm 3.1 on MATLAB. Numerical results show that Algorithm 3.1 yield some approximate solutions that have a better objective function value in comparison with some other algorithms.

Example 3.2. Consider the example in [8],

$$\begin{aligned}
\text{(COP1)} \ \min \ & f(x) = x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4 \\
\text{s.t.} \ & g_1(x) = 2x_1^2 + x_2^2 + x_3^2 + 2x_1 + x_2 + x_4 - 5 \leq 0, \\
& g_2(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_1 - x_2 + x_3 - x_4 - 8 \leq 0, \\
& g_3(x) = x_1^2 + 2x_2^2 + x_3^2 + 2x_4^2 - x_1 - x_4 - 10 \leq 0.
\end{aligned}$$

Let $x^0 = (0,0,0,0)$, $\rho_0 = 4$, $N = 10$, $\epsilon_0 = 0.01$, $\eta = 0.05$ and $\epsilon = 10^{-6}$. Numerical results of Algorithm 3.1 for solving (COP1) are given in Table 1.

Therefore, we get an approximate solution

$$x^3 = (0.170768, 0.827977, 2.011779, -0.960639)$$

at the 3'th iteration. One can easily check that x^3 is a feasible solution since the constraints of (COP1) at x^3 are as follows:

$$\begin{aligned}
g_1(x^3) =& 2 * 0.170768^2 + 0.827977^2 + 2.011779^2 + 2 * 0.170768 \\
& +0.827977 - 0.960639 - 5 = -0.000001922981999,
\end{aligned}$$

TABLE 1. Numerical results of Algorithm 3.1 with $x^0 = (0,0,0,0),\ \rho_0 = 4,\ N = 10$

j	ρ_j	ϵ_j	$f(x^j)$	$g_1(x^j)$	$g_2(x^j)$	$g_3(x^j)$	x^j
1	4	0.01	-44.256310	0.003740	0.009916	-1.872051	(0.169769,0.835551, 2.009753,-0.966244)
2	40	0.0005	-44.233949	0.000019	0.000050	-1.883070	(0.169561,0.835531, 2.008640,-0.964883)
3	400	0.000025	-44.233515	-0.000002	-0.000001	-1.916964	(0.170768,0.827977, 2.011779,-0.960639)

$$\begin{aligned}g_2(x^3) =&0.170768^2 + 0.827977^2 + 2.011779^2 + (-0.960639)^2 + 0.170768\\ &-0.827977 + 2.011779 + 0.960639 - 8 = -0.000001344484998,\\ g_3(x^3) =&0.170768^2 + 2 * 0.827977^2 + 2.011779^2 + 2 * (-0.960639)^2\\ &-0.170768 + 0.960639 - 10 = -1.916966143634999.\end{aligned}$$

The objective function value is given by $f(x^3) = -44.233515$. The solution we obtained is slightly better than the solution obtained in the 4'th iteration by method in [8] (the objective function value $f(x^*) = -44.23040$) for this example.

Now we change the initial parameters. Let $x^0 = (0,0,0,0),\ \rho_0 = 8,\ N = 6,\ \epsilon_0 = 0.01,\ \eta = 0.03$ and $\epsilon = 10^{-6}$. Numerical results of Algorithm 3.1 for solving (COP1) are given in Table 2. Further, with the same parameters $\rho_0,\ N,\ \epsilon_0,\ \eta$ as above, we change the starting point to $x^0 = (8,8,8,8)$. New numerical results are given in Table 3.

It is easy to see from Tables 2 and 3 that the convergence of Algorithm 3.1 is the same and the objective function values are almost the same. That is to say, the efficiency of Algorithm 3.1 does not completely depend on how to choose a starting point in this example.

Note: j is the number of iteration in the Algorithm I.

ρ_j is constrain penalty parameter at the j'th iteration.

x^j is a solution at the j'th iteration in the Algorithm I.

$f(x^j)$ is an objective value at x^j.

$g_i(x^j)$ $(i = 1, \ldots, m)$ is a constrain value at x^j.

Example 3.3. Consider the example in [19],

$$\begin{aligned}\text{(COP2)}\ \min\ & f(x) = -2x_1 - 6x_2 + x_1^2 - 2x_1x_2 + 2x_2^2\\ \text{s.t. }& x_1 + x_2 \le 2,\\ & -x_1 + 2x_2 \le 2,\\ & x_1,\ x_2 \ge 0.\end{aligned}$$

Let

$$g_1(x) = x_1 + x_2 - 2,\ g_2(x) = -x_1 + 2x_2 - 2,$$

TABLE 2. Numerical results of Algorithm 3.1 with $x^0 = (0,0,0,0)$, $\rho_0 = 8$, $N = 6$

j	ρ_j	ϵ_j	$f(x^j)$	$g_1(x^j)$	$g_2(x^j)$	$g_3(x^j)$	x^j
1	8	0.01	-44.245082	0.001869	0.004961	-1.877585	(0.169665,0.835541, 2.009194,-0.965561)
2	48	0.0003	-44.233893	0.000009	0.000025	-1.883098	(0.169561,0.835531, 2.008637,-0.964880)
3	288	0.000009	-44.232243	-0.000052	-0.000174	-1.921571	(0.162363,0.825936, 2.017343,-0.955283)

TABLE 3. Numerical results of Algorithm 3.1 with $x^0 = (8,8,8,8)$, $\rho_0 = 8$, $N = 6$

j	ρ_j	ϵ_j	$f(x^j)$	$g_1(x^j)$	$g_2(x^j)$	$g_3(x^j)$	x^j
1	8	0.01	-44.245082	0.001869	0.004961	-1.877585	(0.169665,0.835541, 2.009194,-0.965561)
2	48	0.0003	-44.233893	0.000009	0.000025	-1.883098	(0.169561,0.835531, 2.008637,-0.964880)
3	288	0.000009	-44.233355	-0.000113	-0.000079	-1.900244	(0.166329,0.831255, 2.012529,-0.960615)

$$g_3(x) = -x_1,\ g_4(x) = -x_2.$$

Thus problem (COP2) is equivalent to the following problem:

$$\begin{aligned}
\text{(COP2')}\ \min\ & f(x) = -2x_1 - 6x_2 + x_1^2 - 2x_1x_2 + 2x_2^2 \\
\text{s.t. } & g_1(x) = x_1 + x_2 - 2 \le 0, \\
& g_2(x) = -x_1 + 2x_2 - 2 \le 0, \\
& g_3(x) = -x_1 \le 0, \\
& g_4(x) = -x_2 \le 0.
\end{aligned}$$

Let $x^0 = (1,1)$, $\rho_0 = 8$, $N = 10$, $\epsilon_0 = 0.5$, $\eta = 0.01$ and $\epsilon = 10^{-6}$. Numerical results of Algorithm 3.1 for solving (COP2') are given in Table 4.

By Table 4, an approximate optimal solution to (COP2') is obtained at the 3'th iteration, that is $x^* = (0.800000, 1.200000)$ with corresponding objective function value $f(x^*) = -7.200000$. The solution we obtained is similar with the solution obtained in the 4'th iteration by method in [19] (the objective function value $f(x^*) = -7.2000$) for this example.

4. Conclusion

This paper has presented a smoothing approximation to the l_1 exact penalty function and an algorithm based on this smoothed penalty problem. It is shown that the optimal solution to the $(NP_{\rho,\epsilon})$ is an approximate optimal solution

TABLE 4. Numerical results of Algorithm 3.1 with $x^0 = (1,1)$, $\rho_0 = 8$, $N = 10$

j	ρ_j	ϵ_j	$f(x^j)$	$g_1(x^j)$	$g_2(x^j)$	x^j
1	8	0.5	-8.111111	0.333333	-0.333333	(1.000000,1.333333)
2	80	0.005	-7.200980	0.000350	-0.399930	(0.800210,1.200140)
3	800	0.00005	-7.200000	0.000000	-0.400000	(0.800000,1.200000)

to the original optimization problem under some mild conditions. Numerical results show that the algorithm proposed here is efficient in solving some COP.

Acknowledgement

The author would like to thank the anonymous reviewers and the editor for their insightful suggestions and comments which led to improvements of this paper. This work is supported by National Natural Science Foundation of China (Grant No.11371242).

References

1. M.S. Bazaraa and J.J. Goode, *Sufficient conditions for a globally exact penalty function without convexity*, Mathematical Programming Study, **19** (1982), 1-15.
2. C.H. Chen and O.L. Mangasarian, *Smoothing methods for convex inequalities and linear complementarity problems*, Math. Program, **71** (1995), 51-69.
3. G. Di Pillo and L. Grippo, *An exact penalty function method with global conergence properties for nonlinear programming problems*, Math. Program, **36** (1986), 1-18.
4. G. Di Pillo and L. Grippo, *Exact penalty functions in contrained optimization*, SIAM J. Control. Optim., **27** (1989), 1333-1360.
5. S.P. Han and O.L. Mangasrian, *Exact penalty function in nonlinear programming*, Math. Program, **17** (1979), 257-269.
6. X.X. Huang and X.Q. Yang, *Convergence analysis of a class of nonlinear penalization methods for constrained optimization via first-order necessary optimality conditions*, J. Optim. Theory Appl., **116** (2003), 311-332.
7. J.B. Lasserre, *A globally convergent algorithm for exact penalty functions* Eur. J. Oper. Res., **7** (1981), 389-395.
8. S.J. Lian, *Smoothing approximation to l_1 exact penalty function for inequality constrained optimization*, Appl. Math. Comput., **219** (2012), 3113-3121.
9. B.Z. Liu, *On smoothing exact penalty functions for nonlinear constrained optimization problems*, J. Appl. Math. Comput., **30** (2009), 259-270.
10. Z.Q. Meng, C.Y. Dang and X.Q. Yang, *On the smoothing of the square-root exact penalty function for inequality constrained optimization*, Comput. Optim. Appl., **35** (2006), 375-398.
11. K.W. Meng, S.J. Li and X.Q. Yang, *A robust SQP method based on a smoothing lower order penalty function*, Optimization, **58** (2009), 22-38.
12. M.C. Pinar and S.A. Zenios, *On smoothing exact penalty function for convex constrained optimization*, SIAM J. Optim., **4** (1994), 486-511.
13. E. Rosenberg, *Exact penalty functions and stability in locally Lipschitz programming*, Math. Program **30** (1984), 340-356.

14. A.M. Rubinov, B.M. Glover and X.Q. Yang, *Extended Lagrange and penalty function in continuous optimization*, Optimization, **46** (1999), 327-351.
15. A.M. Rubinov, X.Q. Yang and A.M. Bagirov, *Penalty functions with a small penalty parameter*, Optim. Methods Softw., **17** (2002), 931-964.
16. A.M. Rubinov and X.Q. Yang, *Nonlinear Lagrangian and Penalty Functions in Optimization*, Kluwer Academic, Dordrecht, 2003.
17. X.L. Sun and D. Li, *Value-estimation function method for constrained global optimization*, J. Optim. Theory Appl., **24** (1999), 385-409.
18. Z.Y. Wu, F.S. Bai, X.Q. Yang and L.S. Zhang, *An exact lower order penalty function and its smoothing in nonlinear programming*, Optimization, **53** (2004), 57-68.
19. Z.Y. Wu, H.W.J. Lee, F.S. Bai, L.S. Zhang and X.M. Yang, *Quadratic smoothing approximation to l_1 exact penalty function in global optimization*, J. Ind. Manag. Optim., **1** (2005), 533-547.
20. X.S. Xu, Z.Q. Meng, J.W. Sun and R. Shen, *A penalty function method based on smoothing lower order penalty function*, J. Comput. Appl. Math., **235** (2011), 4047-4058.
21. X.S. Xu, Z.Q. Meng, J.W. Sun, L.Q. Huang and R. Shen, *A second-order smooth penalty function algorithm for constrained optimization problems*, Comput. Optim. Appl., **55** (2013), 155-172.
22. X.Q. Yang, *Smoothing approximations to nonsmooth optimization problems*, J. Aust. Math. Soc. B, **36** (1994), 274-285.
23. X.Q. Yang and X.X. Huang, *A nonlinear Lagrangian approach to constrained optimization problems*, SIAM J. Optim., **11** (2001), 1119-1144.
24. X.Q. Yang, Z.Q. Meng, X.X. Huang and G.T.Y. Pong, *Smoothing nonlinear penalty function for constrained optimization*, Numer. Funct. Anal. Optim., **24** (2003), 357-364.
25. W.I. Zangwill, *Nonlinear programming via penalty function*, Mgmt. Sci., **13** (1967), 334-358.
26. S.A. Zenios, M.C. Pinar and R.S. Dembo, *A smooth penalty function algorithm for network-structured problems*, Eur. J. Oper. Res., **64** (1993), 258-277.

Nguyen Thanh Binh received M.Sc. from Thai Nguyen University of Education. He is also studying in Ph.D degree at Department of Mathematics, Shanghai University, China. His research interests include nonlinear optimization and computational mathematics.

Department of Mathematics, Shanghai University, Shanghai, 200444, China.
e-mail: bingbongyb@gmail.com