Projected Barzilai-Borwein Gradient Algorithm Based on Complementarity Constrained Programming Model for Absolute Value Equations (Postprint)

Authors: Yan Tao

Date: 2019-01-28T00:00:00+00:00

Abstract

Under the solvability condition of the absolute value equation problem Ax-|x|=b, a new method for solving absolute value problems is presented. First, an equivalent complementarity-constrained programming model for solving absolute value problems is established; subsequently, by exploiting the non-negative constraints in the new model, a projected Barzilai-Borwein (BB) gradient algorithm for solving absolute value problems is proposed. Numerical experimental results demonstrate the effectiveness of the method.

Full Text

Preamble

Vol. 37 No. 4
Application Research of Computers

Projected Barzilai-Borwein Gradient Method for Absolute Value Equations Based on Mathematical Programs with Complementarity Constraints Model

Yan Tao
(Dept. of Mathematics, College of Science, Nanjing University of Science & Technology, Nanjing 210094, China)

Abstract: This paper presents a novel method for solving absolute value equations under the condition that a solution exists. First, an equivalent mathematical program with complementarity constraints (MPCC) model is established for solving absolute value equations. Then, utilizing the non-negative constraints in the new model, a projected Barzilai-Borwein (BB) gradient algorithm is proposed for solving absolute value equations. Numerical experimental results demonstrate the effectiveness of the method.

Keywords: absolute value equation; mathematical program with complementarity constraints; projected gradient method

---

0 Introduction

Consider the following absolute value equation problem:

$$Ax - |x| = b$$

where $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$, and $|x|$ denotes the component-wise absolute value of x. The absolute value equation problem represents a special case of the generalized absolute value problem $Ax+B|x| = b$, and under certain conditions, it is equivalent to the linear complementarity problem [2,3]. This problem has attracted extensive research on its theory and algorithms, yielding numerous relevant results. Reference [4] provides a detailed introduction to the related theory. Based on nonsmooth equations, references [5,6] propose corresponding semismooth Newton methods; by applying various smoothing functions to transform the original problem into smooth equations, references [7-9] develop corresponding smoothing methods; reference [10] converts the absolute value equation problem into an unconstrained optimization problem when A is symmetric, and subsequently proposes a Gauss-Seidel algorithm for solving the original problem.

On the other hand, mathematical programs with complementarity constraints (MPCC) constitute a class of optimization problems containing complementarity conditions in their constraints. They represent a special subclass of mathematical programs with equilibrium constraints (MPEC) and find important applications in practical problems such as engineering design and economic modeling [11]. In recent years, significant progress has been made in algorithms for MPCC, including penalty function approximation methods [12], smoothing approximation methods [13], and regularization methods [14]. Inspired by these results, this paper establishes an equivalent MPCC model for solving absolute value equations, constructs a penalty function subproblem, and employs a projected BB gradient method to solve it.

1 Complementarity Constraints Programming Model

First, we present a constrained equation system equivalent to the absolute value equation problem.

Theorem 1 If x^* is a solution of the absolute value equation problem (1), then w^* is a solution of problem (2); conversely, if w^* is a solution of problem (2), defining $x^* = W_1^* - W_2^*$, then x^* is a solution of absolute value equation problem (1).

Proof The first part follows directly by substituting x^* into problem (2). For the second part, we first decompose x^* as $x^* = x_+^* - x_-^*$, where $x_+^*, x_-^* \geq 0$ and $x_+^{*T} x_-^* = 0$. Therefore, we can construct $w^* = (x_+^{*T}, x_-^{*T})^T$, and the first equality in problem (2) becomes $A(W_1^* - W_2^*) + W_1^* + W_2^* = b$. From problem (2), we have $x^* = W_1^* - W_2^*$ and $|x^*| = W_1^* + W_2^*$. We now only need to prove $W_1^* = x_+^*$ and $W_2^* = x_-^*$. From the construction of w^*, we have $x^* = W_1^* - W_2^*$ and $|x^*| = W_1^* + W_2^*$. Solving these equations yields $W_1^* = \frac{|x^*|+x^*}{2} = x_+^*$ and $W_2^* = \frac{|x^*|-x^*}{2} = x_-^*$, which completes the proof of the second part.

Furthermore, this paper reformulates problem (2) as the following mathematical program with complementarity constraints:

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}\|Aw - b\|^2 \\ \text{s.t.} \quad & G(w) \geq 0,\ H(w) \geq 0,\ G(w)^T H(w) = 0 \end{aligned}$$

where $G(w) = (w_1, w_2, \dots, w_n)^T$ and $H(w) = (w_{n+1}, w_{n+2}, \dots, w_{2n})^T$. Moreover, w^* is an optimal solution of problem (3) if and only if w^* is a solution of problem (2). Therefore, solving absolute value equation problem (1) can be transformed into solving MPCC problem (3).

2 Algorithm

The algorithm consists of two iterative processes: an outer iteration for updating the penalty parameter ρ and an inner iteration for solving the subproblem at each step. To implement the solution algorithm, following reference [15], we first construct the following penalty function problem:

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}\|Aw - b\|^2 + \rho G(w)^T H(w) \\ \text{s.t.} \quad & G(w) \geq 0,\ H(w) \geq 0 \end{aligned}$$

where ρ is the penalty parameter. Model (4) has simple non-negative constraints like those in MPCC (3) and can conveniently apply projection algorithms. Moreover, we can prove that solving the penalty function problem (4) yields a stationary point of MPCC problem (3). Therefore, the algorithm construction will be based on model (4).

Below we present the outer iteration algorithm process and corresponding convergence results.

Outer Iteration Algorithm

a) Set initial point $w^{(1)} = (0, 0, \dots, 0)^T \geq 0$, $\rho_1 > 0$, $\beta > 1$, and $k = 1$.

b) Using $w^{(k)}$ as the initial point, solve for a stationary point $w^{(k+1)}$ of problem (4) with $\rho = \rho_k$.

c) If $\|Aw^{(k+1)} - b\| \leq$ tolerance, stop; otherwise, set $\rho_{k+1} = \beta\rho_k$, $k = k + 1$, and return to step b).

Theorem 2 For each ρ_k, let $w^{(k)}$ be a stationary point of problem (4). Let $\bar{w}$ be a limit point of the sequence $\{w^{(k)}\}$, and assume that $\bar{w}$ is a feasible point of problem (3). Then $\bar{w}$ is a C-stationary point of MPCC problem (3).

Proof From the definition of $w^{(k)}$, we have that $w^{(k)}$ is a stationary point of the penalty function problem (4). Let $I_G(\bar{w}) = \{i : G_i(\bar{w}) = 0\}$ and $I_H(\bar{w}) = \{i : H_i(\bar{w}) = 0\}$. By Lemma 3.2 in reference [12], $\bar{w}$ is a feasible point of MPCC problem (3). Furthermore, since the linear independence constraint qualification for MPCC holds at $\bar{w}$, i.e., $\{\nabla G_i(\bar{w}), \nabla H_j(\bar{w}) : i \in I_G(\bar{w}), j \in I_H(\bar{w})\}$ is linearly independent, by Theorem 2.1 in reference [12], $\bar{w}$ is a C-stationary point of MPCC problem (3).

In the outer iteration algorithm, we need to solve problem (4) in step c) at each iteration. Below we present the inner iteration algorithm process. Let $f(w) = \frac{1}{2}\|Aw - b\|^2 + \rho G(w)^T H(w)$. The penalty function problem (4) can be simplified as:

$$\begin{aligned} \min_{w} \quad & f(w) \\ \text{s.t.} \quad & w \geq 0 \end{aligned}$$

In the inner iteration algorithm, we use the projected Barzilai-Borwein gradient method [16] to solve problem (5). The search direction is set as:

$$d^{(l)} = P(w^{(l)} - \alpha_{BB} \nabla f(w^{(l)})) - w^{(l)}$$

where $P(\cdot)$ is the projection operator and α_{BB} is the step size obtained by the BB method. Moreover, to ensure global convergence, the algorithm employs a nonmonotone line search along the projection direction to obtain a line search step size λ_l, with the iteration formula:

$$w^{(l+1)} = w^{(l)} + \lambda_l d^{(l)}$$

To distinguish the variable notation between outer and inner iterations, we denote the inner iteration variable as $z^{(l)}$.

Inner Iteration Algorithm

a) Set initial point $z^{(1)} = w^{(k)}$, $l = 1$, and $\alpha_{BB}^{(1)} = 1$.

b) Compute the projected gradient direction $d^{(l)} = P(z^{(l)} - \alpha_{BB}^{(l)} \nabla f(z^{(l)})) - z^{(l)}$. If $\|d^{(l)}\| \leq$ tolerance, stop and output $w^{(k+1)} = z^{(l)}$.

c) Compute the line search step size λ_l using the nonmonotone line search.

d) Update $z^{(l+1)} = z^{(l)} + \lambda_l d^{(l)}$.

e) If $l = 200$, stop; otherwise, update $\alpha_{BB}^{(l+1)}$ using the BB formula, set $l = l+1$, and return to step b).

Theorem 3 Any accumulation point of the sequence $\{z^{(l)}\}$ generated by the inner iteration algorithm is a stationary point of problem (5).

3 Numerical Results

The stopping criterion for the outer iteration algorithm is set as $\|Aw-b\| \leq 10^{-6}$. Below we discuss the stopping criterion for the inner iteration algorithm. For the simplified penalty function model (5), we define the projected gradient as:

$$\nabla^P f(w) = \begin{cases} \nabla_i f(w) & \text{if } w_i > 0 \text{ or } \nabla_i f(w) \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

Then w is a stationary point of problem (5) if and only if $\nabla^P f(w) = 0$. From Lemma 2.1 in reference [16], this condition is equivalent to $\|P(w-\nabla f(w))-w\| = 0$. Moreover, numerical experiments in reference [17] show that using the former condition yields better results. Therefore, the inner iteration algorithm stopping criterion is set as $\|\nabla^P f(w)\| \leq 10^{-3}$. On the other hand, if the inner iteration algorithm terminates after only one iteration, we reduce the value of ρ by setting $\rho = \rho/4$.

We first generate 100 random absolute value equation problems as test problems. To ensure the problems have solutions, we construct matrix A such that its singular values are greater than 1. Specifically, we set $A = U\Sigma V^T$, where U and V are orthogonal matrices uniformly distributed in $[-10, 10]$, and the diagonal elements of Σ are randomly selected from the interval $[1, 50]$. We take $b = Ax^* - |x^*|$ where x^* is a vector uniformly distributed in $[-1, 1]$. The initial point for the outer iteration algorithm is set as $w^{(1)} = |\text{rand}(2n, 1) - \text{rand}(2n, 1)|$. Our algorithm successfully solved all test problems within the specified error tolerance. The detailed experimental results are shown in Table 1 .

Table 1 Results of 100 random test problems

Problem dimension n	Number of successful solves	Average outer iterations	Average inner iterations	Average CPU time (s)	Average $\|Ax-b\|$	Average $\|x-x^*\|$
200	100	3.2	28.4	2.2168E+003	2.7181E-007	1.8910E-007
500	100	3.8	35.6	3.3253E+003	6.2445E-007	4.7039E-008
1000	100	4.1	42.3	5.0775E+003	5.0775E-007	1.8032E-007

Additionally, we use the following two examples for numerical verification.

Example 2 The matrix A is defined as $A = I - 4B$, where $B_{ij} = \frac{a_i a_j}{a_{ii}+a_{jj}}$ with $a_i = i$ for $i, j = 1, 2, \ldots, n$ and $a_{ii} = n + 0.5$. Let $b = Ae - |e|$, where I is the n-dimensional identity matrix and e is a column vector of all ones. The unique solution of this problem is $x^* = (1, 1, \ldots, 1)^T$ [18]. The initial point is set to the zero vector. The algorithm was tested on problems with dimensions $n = 200, 500, 1000$ and obtained the unique solution in each case. Other numerical values such as iteration counts, penalty parameter values at termination, and $\|Ax - b\|$ are shown in Table 2 .

Table 2 Results of Example 2

Dimension n	Outer iterations	Inner iterations	Final ρ	$\|Ax - b\|$
200	3	25	1.8910E-007	6.2445E-007
500	4	32	2.7181E-007	1.8910E-007
1000	4	38	5.0775E-007	1.8032E-007

Example 3 [19] The matrix A is defined as:

$$A_{ij} = \begin{cases} 4 + 2(i-1) & \text{if } i = j \\ 0.5 + 0.25(i-1) & \text{if } i \neq j \end{cases}$$

Let $b = Ae - |e|$, where I is the n-dimensional identity matrix and e is a column vector of all ones. The unique solution of this problem is $x^* = (1, 1, \dots, 1)^T$. Our algorithm was tested on problems with dimensions $n = 200, 500, 1000$. The initial point is set to the zero vector. After a finite number of iterations, the algorithm obtained the unique solution. The numerical results including iteration counts, penalty parameter values at termination, and $\|Ax - b\|$ are shown in Table 3 .

Table 3 Results of Example 3

Dimension n	Outer iterations	Inner iterations	Final ρ	$\|Ax - b\|$
200	3	22	1.8910E-007	6.2445E-007
500	4	30	2.7181E-007	1.8910E-007
1000	4	36	5.0775E-007	1.8032E-007

4 Conclusion

This paper proposes a projected gradient-type algorithm for solving absolute value equations based on mathematical programs with equilibrium constraints. Numerical experiments verify the effectiveness of the algorithm and demonstrate that it provides a feasible method for solving absolute value equation problems.

References

[1] Rohn, J. A theorem of the alternatives for the equation $Ax + B|x| = b$ [J]. Linear and Multilinear Algebra, 2004, 52 (6): 421-426.

[2] Mangasarian O L. Absolute value programming [J]. Computational Optimization and Application, 2007, 36 (1): 43-53.

[3] Mangasarian O L. Absolute value equation solution via concave minimization [J]. Optimization Letters, 2007, 1 (1): 3-8.

[4] Mangasarian O L, Meyer R R. Absolute value equations [J]. Linear Algebra and its Application, 2006, 419 (5): 359-367.

[5] Mangasarian O L. A generalized Newton method for absolute value equations [J]. Optimization Letters, 2009, 3 (1): 101-108.

[6] Zhang Chao, Wei Qingju. Global and finite convergence of a generalized Newton method for absolute value equations [J]. Journal of Optimization Theory and Applications, 2009, 143 (2): 391-403.

[7] Louis C, Qu Biao, Zhou Guanglu. A globally and quadratically convergent method for absolute value equations [J]. Computation Optimization Apllication, 2011, 48 (1): 45-58.

[8] Yong Longquan, Liu Sangyang, Zhang Shemin. Smoothing Newton method for solving absolute value equations [J]. International Journal of the Physical Science, 2011, 6 (23): 5399-5405.

[9] Deng Yongkun, Wang Haijun, Zhang Pimg. Maximum entropy Newton method for absolute value equations [J]. Application Research of Computers, 2012, 29 (12): 4546-4548.

[10] Muhammad Aslam Noor, Javed Iqbal, Khalida Inayat Noor, et al. On an iterative method for solving absolute value equations [J]. Optimization Letters, 2012, 6 (5): 1027-1033.

[11] Luo Zhiquan, Pang Jongshi, Ralph D. Mathematical programs with equilibrium constraints [M]. Cambridge: Cambridge University Press, 1996.

[12] Hu Xinmin, Ralph D. Convergence of a penalty method for mathematical programming with complementarity constraints [J]. Journal of Optimization Theory and Applications, 2004, 123 (2): 365-390.

[13] Facchiner F, Jiang Houyuan, Qi Liqun. A smoothing method for mathematical programs with equilibrium constrains [J]. Mathematical Programming, 1999, 85 (1): 107-134.

[14] Scholtes S. Convergence properties of a regularization scheme for mathematical programs with complementarity constraints [J]. SIAM Journal on Optimization, 2001, 11 (4): 918-936.

[15] Tin-Loi, F. On the numerical solution of a class of unilateral contact structural optimization problems [J]. Structural Optimization, 1999, 17 (2-3): 155-161.

[16] Birgin E G, Martinez J M, Raydan M. Nonmonotone spectral projected gradient methods on convex sets [J]. SIAM Journal on Optimization, 2000, 10 (4): 1196-1211.

[17] Lin zhiren, More J J. Newton' s method for large bound-constrained optimization problems [J]. SIAM Journal on optimization, 1999, 9 (4): 1100-1127.

[18] Ujevic N. A new iterative method for solving linear systems [J]. Applied Mathematics and Computation, 2006, 179 (2): 725–730.

[19] Burden R L, Faires J D. Numerical analysis [M]. 7th ed. Boston: PWS Publishing Company, 2006.

Note: Figure translations are in progress. See original paper for figures.