Nonlinear Dynamics and Systems Theory, **23** (2) (2023) 119–128

A DC Algorithm for Solving non-Uniquely Solvable Absolute Value Equations

N. Anane *, Z. Kebaili and M. Achache

Fundamental and Numerical Mathematics Laboratory, Ferhat Abbas University, Setif 1, Setif 19000, Algeria.

Received: January 19, 2023; Revised: March 20, 2023

Abstract: In this paper, we deal with the solution of non-uniquely solvable absolute value equations (AVE) of the form $Ax - B|x| = b$, where A, $B \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$. To do so, a non-convex quadratic optimization is considered, where its first-order optimality conditions are reduced to AVEs. Therefore, solving the AVE is equivalent to computing the local minimum of the non-convex quadratic optimization. Next, by exploiting the technique of DC programming, a reformulation of the latter as a DC program is presented. The resulting DC algorithm (DCA) is simple and consists of solving a successive linear system of equations. Numerical experiments on some non-uniquely solvable AVE problems are given to illustrate the efficiency of this approach.

Keywords: *absolute value equations; DC programming; linear system; nonlinear modes; nonlinear systems in control theory.*

Mathematics Subject Classification (2010): 90C50, 90C33, 14C20, 70K75, 93C10.

1 Introduction

In this paper, we consider the absolute value equation (AVE) of the form

$$Ax - B|x| = b, \tag{1}$$

where A, $B \in \mathbb{R}^{n\times n}$, $b, x \in \mathbb{R}^n$ and $|x|$ denotes the component-wise absolute value of the vector x. When $B = I$, the AVE (1) reduces to a special form

$$Ax - |x| = b. \tag{2}$$

* Corresponding author: mailto:nasimaannan@gmail.com.

In the last years, the AVEs have become an interesting topic of research in the domain of mathematical programming and applied sciences. For instance, linear complementarity problems, bi-matrix games and equilibrium problems, and the hydrodynamic equation can be reformulated as AVE (1) [4, 7, 9]. For the existence and uniqueness of solutions of AVE (1) and (2), many results are stated based on different assumptions most of which are made on matrices A and B. Besides, various numerical methods have been developed for solving efficiently the uniquely solvable AVEs (see eg. [1, 2, 5, 8, 10, 13, 17] and the references therein).

The present work deals with AVE (1) that is not necessarily uniquely solvable, i.e., it has more than one solution. For that, a non-convex quadratic optimization is considered and its first-order optimality conditions are reduced to AVE (1). Therefore, finding a solution of AVE (1) is equivalent to computing a local minimum of the corresponding non-convex quadratic optimization. Next, by exploiting the idea of DC programming and DC Algorithm (DCA) for non-convex optimization [11, 14, 15], we propose a simple and efficient iterative method for solving the AVE (1) by its non-convex quadratic optimization. Hence, a suitable DC decomposition of the DC program is proposed for which the DC algorithm is applied. Numerical results are reported by some examples of solvable AVE (1) that can have either a unique solution or many solutions.

At the end of this section, some notations used in the paper are as follows. The scalar product of two vectors x and y in $\mathbb{R}^n$ is denoted by $\langle x, y\rangle = x^T y$. For $x \in \mathbb{R}^n$, the norm $\|x\|$ will denote the Euclidean norm $(x^T x)^{1/2}$ and $sign(x)$ will denote a vector with components equal to $+1, 0$ or -1, depending on whether the corresponding component of x is positive, zero or negative, respectively. In addition, $D := \partial|x|$=Diag($sign(x)$) (D is a diagonal matrix corresponding to $sign(x)$), where $\partial|x|$ represents the generalized Jacobian of $|x|$ based on the sub-gradient. $\lambda_{\max}(A)$ stands for the maximal eigenvalue of a matrix A. The vector of one is denoted by e and the matrix A is positive semi-definite if $x^T Ax \geq 0$ for any $x \in \mathbb{R}^n$. Finally, $\|A\| := \max\{\|Ax\| : x \in \mathbb{R}^n, \|x\| = 1\}$ denotes the induced norm of A.

The paper is organized as follows. In Section 2, a quadratic formulation of the AVE (1) is presented. The equivalence of its first optimality conditions to AVE (1) is shown, where any local minimum of the latter is a solution of the AVE. In Section 3, a brief outline of DC programming and the DCA is given. The DCA for this formulation is discussed. In Section 4, some numerical results are reported. A conclusion and future work outlook end Section 5.

2 Quadratic Formulation of AVE

In this section, we present a quadratic formulation of the AVE (1). It states that when A and B are given arbitrary matrices, the AVE (1) is equivalent to the first-order optimality conditions of the following unconstrained quadratic optimization problem:

$$\min_{x\in\mathbb{R}^n} \; q(x) = \frac{1}{2}\langle Ax - B|x|, x\rangle - \langle b, x\rangle \tag{3}$$

or, equivalently,

$$\min_{x\in\mathbb{R}^n} \; q(x) = \frac{1}{2}\langle Cx - F|x|, x\rangle - \langle b, x\rangle,$$

where $C = A^T + A$, $F = B^T + B$ are symmetric matrices, and $q(x)$ is the quadratic objective function of (3). Indeed, if x satisfies the first-order optimality conditions of

problem (3), then we have $\nabla q(x) = Ax - B|x| - b = 0$. It follows that any local minimum of (3) is a solution of the AVE (1). In the case where q is convex, any local minimum is global. Consequently, any unique solution of AVE (1) is a global minimum of (3).

3 Outline of DC Programming and DCA (Algorithm)

In general, a DC program takes the form

$$\alpha = \inf_{x\in\mathbb{R}^n} \left(q\left(x\right) = g\left(x\right) - h\left(x\right)\right) \left(P_{dc}\right),$$

where g, h are proper lower semi-continuous and convex functions on $\mathbb{R}^n$. The function q is called a DC function, and $g - h$ is a DC decomposition of q, while g and h are the DC components of q.
A point x^* is called a critical point of $g-h$ or a generalized Karush-Kuhn-Tucker (KKT) point of P_{dc} (3) if

$$\partial h(x^*) \cap \partial g(x^*) = \emptyset,$$

where $\partial\phi(x)$ denotes the sub-differential of $\phi(x)$ at the point x. Based on local optimality conditions and duality in DC programming, the DCA generates two sequences $\left\{x^k\right\}$ and $\left\{y^k\right\}$ in the primal and its dual, respectively. Each iteration k of DCA approximates the concave part of $-g$ by its affine majorization (that corresponds to taking $y^k \in \partial h(x^k)$ and minimizing the resulting convex function (that is equivalent to determining a point $x^{k+1} \in \partial g^*(y^k)$ (or $y^k \in \partial g(x^{k+1})$) with g^* being the conjugate function of g. The generic form of a DC algorithm is stated as follows.

3.1 Generic DCA scheme

Initialization: Let $x^0 \in \mathbb{R}^n$ be a starting point, $k := 0$;
Repeat.
Calculate $y^k \in \partial h(x^k)$;
Calculate $x^{k+1} \in \partial g^*(y^k) \Rightarrow y^k \in \partial g(x^{k+1})$;
$k := k + 1$;
Until convergence of $\left\{x^k\right\}$.

We note that the convergence properties of DCA (Algorithm) can be found in details in [14].

4 Proposed DC Decompositions

Let $\rho > 0$ be such that g and h are convex. In this paper, we adopt the following DC decomposition of $q(x)$:

$$q\left(x\right) = g\left(x\right) - h\left(x\right). \tag{4}$$

4.1 DCA for AVE

The DC decomposition of the objective function $q(x)$ is given by

$$g(x) = \frac{1}{2}x^T(A + \rho I)x \text{ and } h\left(x\right) = \frac{1}{2}(x^T(BD + \rho I)x) + x^T b$$

with $D(x)x = |x|$. Then the problem (4) is a DC program in the standard form

$$\min_{x\in\mathbb{R}^n} \{g(x) - h(x)\}.$$

Following the generic DCA scheme and its properties, we detail the ingredients of the DC algorithm for solving AVE (1).

• An initial point $x^0 \in \mathbb{R}^n$.

• Computation of y^k. We have

$$y^k \in \partial h(x^k) = \{\bigtriangledown h(x^k)\} = \{(\rho I + BD(x^k))x^k + b\}.$$

Then

$$y^k = (\rho I + BD(x^k))x^k + b. \tag{5}$$

• Computation of x^{k+1}. We have

$$x^{k+1} \in \partial g^*(y^k) \Rightarrow y^k \in \partial g(x^{k+1}) = \{\bigtriangledown g(x^{k+1})\} = \{(A + \rho I)x^{k+1}\}.$$

Hence

$$y^k = (A + \rho I)x^{k+1}. \tag{6}$$

Consequently, due to (5) and (6), we deduce that the DC algorithm is based only on solving the following linear system to obtain at each iteration k, x^{k+1}:

$$(A + \rho I)x^{k+1} = (\rho I + BD(x^k))x^k + b. \tag{7}$$

• Choice of ρ. The choice of the parameter ρ is based on the fact that g and h in (4) are convex functions. This is equivalent to obtaining for what suitable values of ρ, the Hessian matrices

$$\nabla^2 g(x) = A + \rho I \text{ and } \nabla^2 h(x) = \rho I + BD$$

are positive semi-definite (PSD) for any matrix D whose elements are ± 1 or 0. The matrix $\nabla^2 h(x)$ is a generalized Hessian caused by the non-differentiability of the absolute value function $|x|$. We have $\nabla^2 g(x)$ is PSD if $v^T(A + \rho I)v \geq 0$ for any vector $v \in \mathbb{R}^n$. By the Cauchy-Schwartz inequality, it follows that

$$v^T(A + \rho I)v \geq \rho v^T v - \|A\|\|v\|^2 = (\rho - \|A\|)\|v\|^2.$$

Hence $(A + \rho I)$ is PSD if $(\rho - \|A\|) \geq 0$. Therefore, it suffices to take $\rho \geq \|A\|$ such that $\nabla^2 g(x)$ is PSD and so g is convex. Now, according to the linear system (7), the matrix $(A + \rho I)$ must be invertible to ensure the uniqueness of solution of the latter. Therefore, we require only the values of ρ which provide the positive definiteness of this matrix, i.e., $\rho > \|A\|$.

In a similar way, $\nabla^2 h(x)$ is PSD for any diagonal matrix D whose elements are ± 1 or 0 if $v^T(\rho I + BD)v \geq 0$ for any $v \in \mathbb{R}^n$. Also, by the Cauchy-Schwartz inequality, we get

$$v^T(\rho I + BD)v \geq \rho v^T v - \|B\|\|D\|\|v\|^2 \geq (\rho - \|B\|)\|v\|^2, \ \forall v \in \mathbb{R}^n.$$

Hence, $(\rho I + BD)$ is PSD for all diagonal matrix D whose elements are ± 1 or 0 if $(\rho - \|B\|) \geq 0$. So, it suffices to take $\rho \geq \|B\|$ such that $(\rho I + BD)$ is PSD. Finally, to guarantee that g and h are convex, we take ρ as follows:

$$\rho \geq \rho_{\min} = \max(\|A\|, \|B\|).$$

Remark 4.1 When A and B are symmetric matrices, ρ is taken as follows:

$$\rho \geq \rho_{\min} = \max(|\lambda_{\max}(A)|, |\lambda_{\max}(B)|).$$

Now, according to (7), the DCA for solving AVE (1) is presented in Figure 1 as follows.

Step 0.
A precision $\epsilon > 0$;
a starting point $x^0 \in \mathbb{R}^n$, a parameter $\rho \geq \rho_{\min}$, set $k := 0$;
for $k = 0, 1, ...$
Step 1. Compute x^{k+1} the unique solution of the system (7);
If the relative residue RSD $:= \dfrac{\|x^{k+1} - x^k\|}{1 + \|b\|} \leq \epsilon$,
then stop and x^{k+1} is an approximated solution;
If not, set $k := k + 1$ and go to **Step 1**.

Figure 1. DC Algorithm for the AVE (1).

4.2 Numerical experiments

In this section, we implement the DC algorithm on **MATLAB** and run it on three examples of solvable AVE (1). We denote by x^0 the initial point in the algorithm and x^* is the true solution of the AVE (1). In the tables of the obtained numerical results, (Iter) represents the number of iterations produced by the algorithm and CPU(s) is the elapsed time. In all our implementation, we set $\epsilon = 10^{-6}$. However, the value of $\rho > 0$ is taken such that $\rho \geq \rho_{\min}$, which ensures the convexity of functions g and h as well the uniqueness of solution of system (7). Our stopping criterion is the residual relative error RSD:= $\dfrac{\|x^{k+1} - x^k\|}{1 + \|b\|}$.

Problem 1. Consider the AVE, where A and B are symmetric matrices:

$$A = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 3 & 1 \\ 2 & 1 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 2 & -1 & -2 \\ -1 & -1 & -1 \\ -2 & -1 & 2 \end{bmatrix}, \; b = [-1, \, 2, \, -1]^T.$$

In this example, two initial points are taken as $x_1^0 = [0, \, 0, \, 0]^T$ and $x_2^0 = [0.8, \, 0.8, \, 0.8]^T$. The iterations number, the CPU(s) times and the RSD for our obtained numerical results are stated in Table 1.

	x_1^0			x_2^0		
$\rho \downarrow$	Iter	CPU(s)	RSD	Iter	CPU(s)	RSD
0.8	18	0.006220	$6.2374e-007$	18	0.005908	$7.4684e-007$
2.5	20	0.006584	$4.5457e-007$	20	0.005955	$8.3400e-007$
3	22	0.005967	$6.5456e-007$	23	0.005950	$5.5641e-007$
$\rho_{\min}$	25	0.010668	$5.6823e-007$	26	0.010063	$4.6759e-007$
10	45	0.006927	$8.1837e-007$	55	0.008193	$8.7384e-007$

Table 1.

This example of the AVE has at least two solutions, namely,

$$x_1^* = [-1, 0.5, -1]^T \text{ and } x_2^* = \left[\frac{2}{3}, \frac{1}{6}, -2\right]^T.$$

Problem 2. In this example of AVEs, the matrices A and B are not symmetric and sparse, where

$$A = \begin{bmatrix} -5 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 10 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 10 & \cdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 10 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 10 \end{bmatrix}$$

and

$$B = \begin{bmatrix} 10 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 10 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 10 & \cdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 10 & 0 \\ 0 & 0 & \cdots & 0 & 12 & 3 \end{bmatrix}.$$

For $b = [-15, -20, \cdots, -20, -26]^T$, this example of the AVE admits at least two solutions, namely,

$$x_1^* = [1, -1, \cdots, -1]^T \text{ and } x_2^* = [-3, -1, \cdots, -1]^T.$$

The initial point is taken as

$$x^0 = [0, -0.5, \cdots, -0.5]^T.$$

Then the obtained numerical results with different size of n are shown in Table 2.

Size n		$\rho_{\min}$	20	100
100	iter	8	22	125
	CPU(s)	0.033798	0.075531	0.188725
	RSD	$7.4215e-007$	$9.9621e-007$	$9.6155e-007$
1500	iter	7	19	100
	CPU(s)	5.151696	13.954670	72.055416
	RSD	$5.5016e-007$	$8.7237e-007$	$9.6398e-007$
3000	iter	7	18	93
	CPU(s)	37.860998	95.470970	495.417176
	RSD	$3.8919e-007$	$9.2569e-007$	$9.9564e-007$
4000	iter	6	18	91
	CPU(s)	93.784429	258.326851	1381.492980
	RSD	$9.6312e-007$	$8.0178e-007$	$9.6084e-007$

Table 2.

For $b = [-15, 0, \cdots, 0, 0]^T$, this example of the AVE admits at least two solutions, namely,

$$x_1^* = [1, 0.5, \cdots, 0.5, 0.7857]^T \text{ and } x_2^* = [-3, 0, \cdots, 0]^T.$$

Our starting point in the algorithm for this example is taken as

$$x^0 = [0, -0.5, \cdots, -0.5, -0.5]^T.$$

The obtained numerical results with different size of n are shown in Table 3.

Size n		$\rho_{\min}$	20	100
100	iter	21	30	172
	CPU(s)	0.096385	0.075766	0.220155
	RSD	$7.6245e-007$	$9.9112e-007$	$9.5277e-007$
1500	iter	21	30	172
	CPU(s)	18.841956	21.227324	123.081425
	RSD	$7.6245e-007$	$9.9112e-007$	$9.5277e-007$
3000	iter	21	30	172
	CPU(s)	116.801005	154.224982	882.412729
	RSD	$7.6245e-007$	$9.9112e-007$	$9.5277e-007$
4000	iter	21	30	172
	CPU(s)	294.215091	415.582267	2381.140220
	RSD	$7.6245e-007$	$9.9112e-007$	$9.5277e-007$

Table 3.

Next, we deal with two examples of the AVEs which have a unique solution (see [2, 3, 5]).

Problem 3. Consider the AVE, where

$$A = \begin{bmatrix} -100 & 10 & 0 & \cdots & 0 & 0 \\ 10 & -100 & 10 & \cdots & 0 & 0 \\ 0 & 10 & -100 & \cdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & 10 & 0 \\ 0 & 0 & 0 & \cdots & -100 & 10 \\ 0 & 0 & \cdots & 0 & 10 & -100 \end{bmatrix},$$

and

$$B = \begin{bmatrix} -1 & 0.1 & 0 & \cdots & 0 & 0 \\ 0.1 & -1 & 0.1 & \cdots & 0 & 0 \\ 0 & 0.1 & -1 & \cdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0.1 & 0 \\ 0 & 0 & 0 & \cdots & -1 & 0.1 \\ 0 & 0 & \cdots & 0 & 0.1 & -1 \end{bmatrix}, \; b = (A - I)e.$$

The numerical results with different size of n and with the initial point

$$x^0 = [0.1, \cdots, 0.1]^T$$

are shown in Table 4.

Size n		$\rho_{\min}$	10	35
100	iter	3	6	37
	CPU(s)	0.013635	0.009922	0.044977
	RSD	$3.1876e-009$	$4.9863e-007$	$9.8002e-007$
1500	iter	2	6	38
	CPU(s)	4.078783	5.753656	34.767854
	RSD	$4.2391e-007$	$5.0486e-007$	$7.7765e-007$
3000	iter	2	6	38
	CPU(s)	30.693572	39.979472	259.846648
	RSD	$2.9980e-007$	$5.0511e-007$	$7.7887e-007$
4000	iter	2	6	38
	CPU(s)	70.940271	95.628900	605.623886
	RSD	$2.5965e-007$	$5.0518e-007$	$7.7918e-007$

Table 4.

Now, with and without spacing other initial point $x^0 = [1, 2, \cdots, n]^T$, the numerical results are shown in Table 5.

Size n		$\rho_{\min}$	10	35
100	iter	3	8	53
	CPU(s)	0.022255	0.011792	0.055950
	RSD	$2.4503e-007$	$6.1149e-007$	$9.6970e-007$
1500	iter	3	10	64
	CPU(s)	5.803212	8.905991	58.258065
	RSD	$9.5316e-007$	$1.8358e-007$	$9.9229e-007$
3000	iter	4	10	67
	CPU(s)	44.731209	71.126180	458.376902
	RSD	$3.3368e-009$	$3.7840e-007$	$9.3035e-007$
4000	iter	4	10	68
	CPU(s)	113.000257	156.209290	1065.611029
	RSD	$3.8532e-009$	$5.0839e-007$	$9.6915e-007$

Table 5.

This example is uniquely solvable and for $b = (A - I)e$, the solution is

$$x^* = [1.0215,\ 1.0226,\ 1.0227, \cdots,\ 1.0227,\ 1.0226,\ 1.0215]^T .$$

Problem 4. Consider the AVE, where

$$A = \begin{bmatrix} -25,5 & -2,5 & 0 & \cdots & 0 & 0 \\ -2,5 & -25,5 & -2,5 & \cdots & 0 & 0 \\ 0 & -2,5 & -25,5 & \cdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & -2,5 & 0 \\ 0 & 0 & 0 & \cdots & -25,5 & -2,5 \\ 0 & 0 & \cdots & 0 & -2,5 & -25,5 \end{bmatrix},$$

and

$$B = \begin{bmatrix} 0,6 & -0,01 & 0 & \cdots & 0 & 0 \\ -0,01 & 0,6 & -0,01 & \cdots & 0 & 0 \\ 0 & -0,01 & 0,6 & \cdots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & -0,01 & 0 \\ 0 & 0 & 0 & \cdots & 0,6 & -0,01 \\ 0 & 0 & \cdots & 0 & -0,01 & 0,6 \end{bmatrix}, \; b = (A - I)\, e.$$

The numerical results with different size of n and with the starting point

$$x^0 = [1, 2, \cdots, n]^T,$$

are summarized in Table 6.

Size n		$\rho_{\min}$	0.1	9
100	iter	6	5	50
	CPU(s)	0.025191	0.011119	0.056285
	RSD	$8.3305e-008$	$2.2208e-007$	$8.6286e-007$
1500	iter	6	6	55
	CPU(s)	7.978099	5.531777	51.736639
	RSD	$1.7720e-007$	$9.9349e-008$	$9.0413e-007$
3000	iter	6	6	56
	CPU(s)	57.503325	39.353990	370.091181
	RSD	$2.7508e-007$	$2.0998e-007$	$9.3224e-007$
4000	iter	6	6	57
	CPU(s)	132.305173	93.222723	942.398052
	RSD	$3.4085e-007$	$2.4286e-007$	$8.3854e-007$

Table 6.

This example has a unique solution if $\sigma_{\min}(A) > \sigma_{\max}(B)$ in $[2, 3, 5]$ given by

$$x^* = [1.0144, 1.0134, 1.0135, \cdots, 1.0135, 1.0134, 1.0144]^T.$$

5 Concluding Remarks

In this paper, we have used the technique of DC programming for solving absolute value equations. For that, a quadratic optimization is considered, where its first-order optimality conditions are equivalent to the AVE (1) and where any local minimum of the quadratic problem is a solution of the AVE. Further, based on a suitable decomposition of the objective function $q(x)$, we have designed a simple DC algorithm for solving the AVE (1). Numerical results illustrate that the DC algorithm is efficient for solving some solvable AVE problems that can have either one unique solution or many solutions. A good topic of research in the future is suggesting other DC decompositions of the objective $q(x)$ in order to design other DC algorithms for solving the AVE (1).

Our results have a great importance in application such as the solution of a linear complementarity problem including the linear and convex quadratic optimization, bimatrix games, interval matrix, hydrodynamic equation.

Acknowledgment

This work has been supported by: La Direction Générale de la Recherche Scientifique et du Développement Technologique (DGRSDT-MESRS), under project PRFU number C00L03UN190120190004. Algérie.

References

[1] L. Abdellah, M. Haddou and T. Migot. Solving absolute value equations using complementarity and smoothing functions. *Journal of Computational and Applied Mathematics* **327** (2018) 196–207.

[2] M. Achache. On the unique solvability and numerical study of absolute value equations. *J. Numer. Anal. Approx. Theory* **48** (2) (2019) 112–121.

[3] M. Achache and N. Anane. On unique solvability and Picard's iterative method for absolute value equations. *Bulletin of the Transilvani aUniversity of Brasov* **1(63)** (1) (2021) 13–26.

[4] M. Achache. Complexity analysis and numerical implementation of a short-step primal-dual algorithm for linear complementarity problems. *Applied Mathematics and computation.* **216** (2010) 1889–1895.

[5] M. Achache and N. Hazzam. Solving absolute value equations via complementarity and interior-point methods. *Journal of Nonlinear Functional Analysis.* **2018** (1) (2018) 1–10.

[6] N. Anane and M. Achache. Preconditioned conjugate gradient methods for absolute value equations. *J. Numer. Anal. Approx. Theory* **48** (1) (2020) 3–14.

[7] A. Yu. Aleksandrov. Delay-Independent Stability Conditions for a Class of Nonlinear Mechanical Systems. *Nonlinear Dynamics and Systems Theory* **21** (5) (2021) 447–456.

[8] L. Caccetta, B. Qu and G. Zhou. A globally and quadratically convergent method for absolute value equations. *Computational Optimization and Applications* **48** (1) (2011) 45–58.

[9] R.W. Cottle, J.S. Pang and R.E. Stone. *The linear Complementarity Problem.* Academic Press, New-York, 1992.

[10] M. Hladik. Bounds for the solution of absolute value equations. *Computational Optimization and Applications* **69** (1) (2018) 243–266.

[11] Z. Kebaili and M. Achache. Solving non monotone affine variational inequalities problem by DC programming and DCA. *Asian-European Journal of Mathematics* **13** (1) (2020) 2050067 (8 pages).

[12] O.L. Mangasarian and R.R. Meyer. Absolute value equations. *Linear Algebra and Applications.* **419** (2-3) (2006) 359–367.

[13] O.L. Mangasarian. A generalized Newton method for absolute value equation. *Optimization Letters* **3** (1) (2009) 101–108.

[14] T. Pham Dinh and H.A. Le Thi. Convex analysis approach to dc programming. Theory, algorithms and applications. *Acta Math. Vietnam* **22** (1) (1997) 289–355.

[15] T. Pham Dinh and H.A. Le Thi. A DC optimization algorithm for solving the trust-region subproblem. *SIAM Journal on Optimization* **8** (2) (1998) 476–505.

[16] J. Rohn. On unique solvability of the absolute value equations. *Optimization Letters* (3) (2009) 603–606.

[17] J. Rohn. An algorithm for computing all solutions of an absolute value equation. *Optimization Letters* (6) (2012) 851–856.