AIMS Mathematics

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

AIMS Mathematics, 7(1): 606–616.
DOI: 10.3934/math.2022038
Received: 14 July 2021
Accepted: 29 September 2021
Published: 14 October 2021

Research article

Modified BAS iteration method for absolute value equation

Cui-Xia Li[1,*] **and Long-Quan Yong**[2]

[1] School of Mathematics, Yunnan Normal University, Kunming, Yunnan 650500, China

[2] Shaanxi Key Laboratory of Industrial Automation, Shaanxi University of Technology, Hanzhong, Shaanxi 723001, China

* **Correspondence:** Email: lixiatk@126.com; Tel: +8618211679797.

Abstract: In this paper, to improve the convergence speed of the block-diagonal and anti-block-diagonal splitting (BAS) iteration method, we design a modified BAS (MBAS) method to obtain the numerical solution of the absolute value equation. Theoretical analysis shows that under certain conditions the MBAS method is convergent. Numerical experiments show that the MBAS method is feasible.

Keywords: block-diagonal and anti-block-diagonal splitting; iteration method; absolute value equation; convergence
Mathematics Subject Classification: 65F10, 90C05, 90C30

1. Introduction

To establish the efficient iteration method to obtain the numerical solution of the absolute value equation (AVE)

$$Ax - |x| = b \text{ with } A \in \mathbb{R}^{n\times n} \text{ and } b \in \mathbb{R}^n, \tag{1.1}$$

the following equivalent two-by-two block nonlinear equation of the AVE (1.1) is considered in [1]

$$\begin{cases} Ax - y = b, \\ -|x| + y = 0, \end{cases}$$

i.e.,

$$\bar{A}z = \begin{bmatrix} A & -I \\ -\hat{D} & I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} = \bar{b}, \tag{1.2}$$

where $\hat{D} = D(x) = diag(sign(x)), x \in \mathbb{R}^n$. Afterwards, using the following matrix splitting of matrix $\bar{A}$, that is,

$$\bar{A} = \begin{bmatrix} A & -I \\ -\hat{D} & I \end{bmatrix} = \begin{bmatrix} \alpha I + A & 0 \\ 0 & \alpha I + I \end{bmatrix} - \begin{bmatrix} \alpha I & I \\ \hat{D} & \alpha I \end{bmatrix},$$

where α is a given appropriate constant, the block-diagonal and anti-block-diagonal splitting (BAS) method for the nonlinear equation (1.2) was designed and described as follows:

The BAS method: Let initial vectors $x^{(0)} \in \mathbb{R}^n$ and $y^{(0)} \in \mathbb{R}^n$. For $k = 0, 1, \cdots$ until the iteration sequence $\{x^{(k)}, y^{(k)}\}$ is convergent, calculate

$$\begin{cases} x^{(k+1)} = (\alpha I + A)^{-1}(\alpha x^{(k)} + y^{(k)} + b), \\ y^{(k+1)} = \dfrac{1}{1+\alpha}(\hat{D}x^{(k)} + \alpha y^{(k)}), \end{cases} \tag{1.3}$$

or

$$\begin{bmatrix} \alpha I + A & 0 \\ 0 & \alpha I + I \end{bmatrix} \begin{bmatrix} x^{(k+1)} \\ y^{(k+1)} \end{bmatrix} = \begin{bmatrix} \alpha I & I \\ \hat{D} & \alpha I \end{bmatrix} \begin{bmatrix} x^{(k)} \\ y^{(k)} \end{bmatrix} + \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{1.4}$$

where α is a given appropriate constant.

In [1], some conditions were given to guarantee the convergence of the BAS method. Numerical experiment results showed that the convergence behavior of the BAS method is better than the generalized Newton (GN) method in [2] and the nonlinear HSS-like (NHSS) method [3, 4].

As is known, the AVE (1.1) is widely concerned because it is viewed as a very useful tool in a number of practical problems, including linear programming, the quasi-complementarity problems, bimatrix games, see [5–9]. Recently, many authors have exploited some feasible iteration methods to obtain the numerical solution of the AVE (1.1), see [2, 6, 10–13, 15–19]. In addition, the AVE (1.1) also looks like the basis pursuit problem and generalized inverses computation, see [23–25].

In this paper, a new iteration method is designed to solve the AVE (1.1). Specifically, to improve the convergence speed of the BAS iteration method, a modified BAS (MBAS) iteration method is developed to solve the AVE (1.1). The convergence property of the MBAS method is studied under certain conditions. The feasibility of the MBAS method is verified by numerical experiments.

The remainder of the paper is structured below. In Section 2, we establish the modified BAS (MBAS) method to obtain the numerical solution of the AVE (1.1) and present some conditions to guarantee the convergence of the MBAS iteration method. In Section 3, the effectiveness of the MBAS method is verified by numerical experiments. In Section 4, we terminate the paper with some conclusions.

2. The MBAS method

In this section, we will establish the MBAS iteration method to acquire the numerical solution of the AVE (1.1). For this purpose, our way is to use the $x^{(k+1)}$ of the first equation in (1.3) instead of the $x^{(k)}$ of the second equation in (1.3). Clearly, the goal of our approach is that we not only can enhance the convergence behavior of the BAS iteration method (1.3), but also can reduce the storage requirements of the BAS iteration method (1.3). Based on this approach, we obtain the modified BAS (MBAS) iteration method and describe as follows.

The MBAS iteration method: Let initial vectors $x^{(0)} \in \mathbb{R}^n$ and $y^{(0)} \in \mathbb{R}^n$. For $k = 0, 1, \cdots$ until the iteration sequence $\{x^{(k)}, y^{(k)}\}$ is convergent, calculate

$$\begin{cases} x^{(k+1)} = (\alpha I + A)^{-1}(\alpha x^{(k)} + y^{(k)} + b), \\ y^{(k+1)} = \dfrac{1}{1+\alpha}(\hat{D}x^{(k+1)} + \alpha y^{(k)}), \end{cases} \tag{2.1}$$

where α is a given nonnegative constant.

The structure of the system (1.2) is two-by-two block and looks like the saddle point problem [26]. Further, from the form of the MBAS iteration method (2.1), in a way, it can be also seen as a special case of parameterized inexact Uzawa method. For this, one can see [27] for more details.

Lemma 2.1 is introduced to obtain some conditions to guarantee the convergence of the MBAS iteration method.

Lemma 2.1. [14] *Let $x^2 - ax + b = 0$ with $a, b \in \mathbb{R}$, and λ be any root of this equation. Then $|\lambda| < 1$ iff $|b| < 1$ and $|a| < 1 + b$.*

Let the iteration errors be

$$e_k^x = x^* - x^{(k)}, e_k^y = y^* - y^{(k)},$$

where x^* and y^* satisfy Eq (1.2). Then the following main result with respect to the MBAS iteration method (2.1) can be obtained. $\|\cdot\|$ denotes the Euclid norm.

Theorem 2.2. *Let $A \in \mathbb{R}^{n\times n}$ be nonsingular. Denote*

$$\beta = \|(\alpha I + A)^{-1}\|,$$

where α is a given nonnegative constant. When

$$\alpha^2\beta < (1+\alpha) < \frac{1}{\beta}, \tag{2.2}$$

the MBAS iteration method (2.1) *is convergent.*

Proof. Combining (1.2) with (2.1), we have

$$\begin{cases} e_{k+1}^x = \alpha(\alpha I + A)^{-1}e_k^x + (\alpha I + A)^{-1}e_k^y, \\ e_{k+1}^y = \dfrac{1}{1+\alpha}(\hat{D}e_{k+1}^x + \alpha e_k^y). \end{cases} \tag{2.3}$$

From (2.3), we can get

$$\begin{aligned} \|e_{k+1}^x\| &= \|\alpha(\alpha I + A)^{-1}e_k^x + (\alpha I + A)^{-1}e_k^y\| \\ &\le \alpha\|(\alpha I + A)^{-1}e_k^x\| + \|(\alpha I + A)^{-1}e_k^y\| \\ &\le \alpha\|(\alpha I + A)^{-1}\| \cdot \|e_k^x\| + \|(\alpha I + A)^{-1}\| \cdot \|e_k^y\| \\ &= \alpha\beta\|e_k^x\| + \beta\|e_k^y\| \end{aligned}$$

and

$$\begin{aligned}\|e_{k+1}^y\| &= \|\frac{1}{1+\alpha}(\hat{D}e_{k+1}^x + \alpha e_k^y)\| \\ &\leq \|\frac{1}{1+\alpha}\hat{D}e_{k+1}^x\| + \|\frac{\alpha}{1+\alpha}e_k^y\| \\ &= \frac{1}{1+\alpha}\|\hat{D}e_{k+1}^x\| + \frac{\alpha}{1+\alpha}\|e_k^y\| \\ &\leq \frac{1}{1+\alpha}\|\hat{D}\| \cdot \|e_{k+1}^x\| + \frac{\alpha}{1+\alpha}\|e_k^y\| \\ &\leq \frac{1}{1+\alpha}\|e_{k+1}^x\| + \frac{\alpha}{1+\alpha}\|e_k^y\|.\end{aligned}$$

Further, let $z^k = e^k = (e_k^x, e_k^y)^T$, then

$$\begin{aligned}z^{k+1} &\leq \begin{pmatrix} \alpha\beta & \beta \\ \frac{\alpha\beta}{1+\alpha} & \frac{\alpha+\beta}{1+\alpha} \end{pmatrix} z^k \\ &\leq \begin{pmatrix} \alpha\beta & \beta \\ \frac{\alpha\beta}{1+\alpha} & \frac{\alpha+\beta}{1+\alpha} \end{pmatrix}^2 z^{k-1} \\ &\cdots \\ &\leq \begin{pmatrix} \alpha\beta & \beta \\ \frac{\alpha\beta}{1+\alpha} & \frac{\alpha+\beta}{1+\alpha} \end{pmatrix}^k z^0.\end{aligned}$$

Let

$$T = \begin{pmatrix} \alpha\beta & \beta \\ \frac{\alpha\beta}{1+\alpha} & \frac{\alpha+\beta}{1+\alpha} \end{pmatrix}.$$

Clearly, if $\rho(T) < 1$, then $\lim_{k\to\infty} T^k = 0$. This implies

$$\lim_{k\to\infty} \|e_k^x\| = 0 \text{ and } \lim_{k\to\infty} \|e_k^y\| = 0.$$

In this way, the MBAS iteration method (2.1) can converge to the solution of the AVE (1.1).

Next, we just need to get the sufficient condition for $\rho(T) < 1$. Let λ be an eigenvalue of the matrix T. Then λ satisfies

$$(\lambda - \alpha\beta)(\lambda - \frac{\alpha+\beta}{1+\alpha}) - \frac{\alpha\beta^2}{1+\alpha} = 0,$$

equivalently,

$$\lambda^2 - (\alpha\beta + \frac{\alpha+\beta}{1+\alpha})\lambda + \frac{\alpha^2\beta}{1+\alpha} = 0. \tag{2.4}$$

Using Lemma 2.1 for Eq (2.4), $|\lambda| < 1$ if and only if

$$|\frac{\alpha^2\beta}{1+\alpha}| < 1$$

and

$$|\alpha\beta + \frac{\alpha+\beta}{1+\alpha}| < 1 + \frac{\alpha^2\beta}{1+\alpha}.$$

Therefore, $\rho(T) < 1$ when the condition (2.2) holds. □

Theorem 2.3. *Let $A \in \mathbb{R}^{n\times n}$ be symmetric positive definite, $\lambda_{\min}$ indicate the minimum eigenvalue of matrix A and $\alpha \geq 0$. When*

$$1 < \lambda_{\min},$$

the MBAS iteration method (2.1) *is convergent.*

Proof. By calculation, we have

$$\begin{aligned}\beta(1+\alpha) &= (1+\alpha)\|(\alpha I + A)^{-1}\| \\ &= (1+\alpha)\|(\alpha I + A)^{-1}\| \\ &= \frac{1+\alpha}{\alpha + \lambda_{\min}}\end{aligned}$$

and

$$\begin{aligned}\beta\alpha^2 &= \alpha^2\|(\alpha I + A)^{-1}\| \\ &= \alpha^2\|(\alpha I + A)^{-1}\| \\ &= \frac{\alpha^2}{\alpha + \lambda_{\min}} < 1 + \alpha.\end{aligned}$$

Obviously, when $1 < \lambda_{\min}$, we have $\beta(1+\alpha) < 1$. □

Corollary 2.4. *Let $A \in \mathbb{R}^{n\times n}$ be nonsingular and $\alpha \geq 0$. When*

$$\|A^{-1}\| \leq \frac{1}{1+2\alpha}, \tag{2.5}$$

the MABS iteration method (2.1) *is convergent.*

Proof. Utilizing the Banach perturbation lemma in [20], we obtain

$$\beta(1+\alpha) \leq \frac{(1+\alpha)\|A^{-1}\|}{1-\alpha\|A^{-1}\|} \text{ and } \beta\alpha^2 \leq \frac{\alpha^2\|A^{-1}\|}{1-\alpha\|A^{-1}\|}.$$

To make the condition (2.2) valid, we only need

$$\frac{(1+\alpha)\|A^{-1}\|}{1-\alpha\|A^{-1}\|} < 1 \text{ and } \frac{\alpha^2\|A^{-1}\|}{1-\alpha\|A^{-1}\|} < 1 + \alpha.$$

By the simple computations, it is easy to see that the results of Corollary 2.4 hold under the condition (2.5). □

Corollary 2.4 is the same as Corollary 1 in [1]. That is to say, the convergence conditions of Corollary 2.1 are suitable for the BAS method.

3. Numerical experiments

In this section, to detect the feasiblity of the MBAS method (2.1) to gain the numerical solution of the AVE (1.1), we give some numerical experiments. Here, by the iteration steps (IT) and the CPU time (CPU) in seconds, we contrast the MBAS method (2.1) with the SOR-like method in [21], the BAS method (1.3), and the following new iteration (NI) method in [22]

$$\begin{cases} x^{(k+1)} = A^{-1}(y^{(k)} + b), \\ y^{(k+1)} = \alpha|x^{(k+1)}| + (1-\alpha)y^{(k)}, \end{cases} \tag{3.1}$$

where $\alpha > 0$.

In these testing four methods, we choose zero vector as all initial vectors, and all iterations of these four methods are stopped when RES $\leq 10^{-6}$, where 'RES' indicates the relative residual error and is of form

$$\text{RES} = \frac{\|Ax^{(k)} - |x^{(k)}| - b\|}{\|b\|},$$

or the number of iteration outnumbers 500. The right-hand side vector b of the AVE (1.1) is taken in a way such that the vector $x = (x_1, x_2, \ldots,)^T$ with

$$x_i = (-1)^i, i = 1, 2, \ldots,$$

is the exact solution. The iteration parameters used in the above four iteration methods are the experimental optimal ones, which minimize the numbers of iteration steps. If the experimental optimal iteration parameters form an interval, then they are further optimized according to the least CPU time. Naturally, in the following tables, α_{exp} indicates the the experimentally optimal parameters for these testing four methods. Here, we use MATLAB R2016B for all the tests.

Example 1. Let the AVE in (1.1) be

$$A = \text{tridiag}(-1, 4, -1) \in \mathbb{R}^{n\times n}, x^* = (-1, 1, -1, 1, \ldots)^T \in \mathbb{R}^n,$$

and $b = Ax^* - |x^*|$ in [21].

For Example 1, the numerical results (including IT, CPU and α_{exp}) for these testing four methods are listed in Table 1. Clearly, these testing four methods can converge rapidly under the corresponding experimentally optimal parameters. Further, on the base of these numerical results in Table 1, we can find that the number of iterations of these testing four methods are nearly sensitivity when the mesh sizes are changed. According to the numerical results in Table 1, the MBAS method has better computational efficiency by the iteration steps and the CPU times, compared with the BAS method, the SOR-like method and the NI method.

Table 1. Numerical comparison of Example 1.

	n	3000	4000	5000	6000	7000
MBAS	IT	11	11	11	11	11
	CPU	0.0019	0.0029	0.0031	0.0056	0.0071
	α_{exp}	0.02	0.02	0.02	0.02	0.02
BAS	IT	21	21	21	21	21
	CPU	0.2632	0.4424	0.6071	0.8973	1.1976
	α_{exp}	0.02	0.02	0.02	0.01	0.02
SOR-like	IT	16	16	16	17	17
	CPU	0.0042	0.0066	0.0079	0.0099	0.0117
	α_{exp}	1	1	1	1.02	1.01
NI	IT	16	16	16	17	17
	CPU	0.0048	0.0066	0.0076	0.0088	0.0126
	α_{exp}	1	1	1	1.01	1.01

Example 2. Let the AVE in (1.1) be $A = \bar{A} + \mu I$, where

$$\bar{A} = \begin{bmatrix} W & -I & -I & 0 & \cdots & 0 & 0 \\ 0 & W & -I & -I & \cdots & 0 & 0 \\ 0 & 0 & W & -I & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & -I \\ \vdots & \vdots & \ddots & \ddots & \ddots & W & -I \\ 0 & 0 & \ddots & \ddots & \ddots & 0 & W \end{bmatrix} \in \mathbb{R}^{n\times n},$$

with

$$W = \text{tridiag}(-1, 4, -1) = \begin{bmatrix} 4 & -1 & 0 & \cdots & 0 & 0 \\ -1 & 4 & -1 & \cdots & 0 & 0 \\ 0 & -1 & 4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 4 & -1 \\ 0 & 0 & 0 & \cdots & -1 & 4 \end{bmatrix} \in \mathbb{R}^{m\times m},$$

and

$$x^* = (-1, 1, -1, 1, \ldots)^T \in \mathbb{R}^n \text{ and } b = Ax^* - |x^*|.$$

Similar to Table 1, for Example 2, Tables 2 and 3 for the different value of μ enumerate the numerical results (including IT, CPU and α_{exp}) of these testing four methods. In our numerical experiments, we take $\mu = 2, 4$. These numerical results in Tables 2 and 3 further verify the observed result from Table 1, i.e., the MBAS method precedes the BAS method [1], the SOR-like method [21] and the NI method [22] in the aspect of the computational efficiency under certain conditions.

Table 2. Numerical comparison of Example 2 with $\mu = 2$.

	m	60	70	80	90
MBAS	IT	11	11	11	11
	CPU	0.2154	0.3107	0.4328	0.5799
	α_{exp}	0.01	0.03	0.01	0.02
BAS	IT	21	21	21	21
	CPU	1.7489	3.2116	5.5408	8.9070
	α_{exp}	0.03	0.03	0.01	0.01
SOR-like	IT	16	16	16	17
	CPU	0.3434	0.4983	0.7140	0.8760
	α_{exp}	1	1	1	1.02
NI	IT	16	16	16	17
	CPU	0.3706	0.5125	0.8135	0.8948
	α_{exp}	1	1	1	1.01

Table 3. Numerical comparison of Example 2 with $\mu = 4$.

	m	60	70	80	90
MBAS	IT	8	8	8	8
	CPU	0.1630	0.2465	0.3345	0.4238
	α_{exp}	0.01	0.01	0.01	0.01
BAS	IT	15	15	15	15
	CPU	1.2469	2.2818	3.9165	6.3493
	α_{exp}	0.01	0.02	0.01	0.01
SOR-like	IT	12	12	12	12
	CPU	0.2430	0.4105	0.4834	0.7020
	α_{exp}	1.01	1.01	1.01	1
NI	IT	12	12	12	12
	CPU	0.2617	0.3308	0.5424	0.6945
	α_{exp}	1	1	1	1

Example 3. [22] Consider the AVE in (1.1), where the matrix $A = \hat{A} + \mu I_{m^2} (\mu \geq 0)$ with

$$\hat{A} = \text{Tridiag}(-I_m, S_m, -I_m) \in \mathbb{R}^{m^2 \times m^2}, S_m = \text{tridiag}(-1, 4, -1) \in \mathbb{R}^{m \times m},$$

and

$$x^* = (-1, 1, -1, 1, \ldots)^T \in \mathbb{R}^n \text{ and } b = Ax^* - |x^*|.$$

For Example 3, we still investigate the efficiency of the above four methods. The corresponding numerical results are listed in Tables 4 and 5. The numerical results in Table 4 correspond to $\mu = 4$, naturally, the numerical results in Table 5 correspond to $\mu = 8$.

These numerical results in Tables 4 and 5 still verify the observed result from Table 1. Under the corresponding experimentally optimal parameters, the four testing methods converge rapidly to the unique solution of Example 3. Among four testing methods, the computational efficiency of the MBAS

method is best, compared with the BAS method [1], the SOR-like method [21] with the NI method [22] under certain conditions.

Table 4. Numerical comparison of Example 3 with $\mu = 4$.

	m	30	50	70	90
MBAS	IT	8	8	8	8
	CPU	0.0082	0.0223	0.0576	0.0967
	α_{exp}	0.02	0.01	0.01	0.01
BAS	IT	15	15	15	15
	CPU	0.0156	0.0755	0.2196	0.5442
	α_{exp}	0.01	0.01	0.01	0.01
SOR-like	IT	11	12	12	12
	CPU	0.0121	0.0345	0.1028	0.1775
	α_{exp}	1	1.01	1.01	1
NI	IT	11	12	12	12
	CPU	0.0128	0.0340	0.1210	0.1736
	α_{exp}	1	1	1	1

Table 5. Numerical comparison of Example 3 with $\mu = 8$.

	m	30	50	70	90
MBAS	IT	7	7	7	7
	CPU	0.0061	0.0178	0.0477	0.0792
	α_{exp}	0.06	0.06	0.03	0.04
BAS	IT	13	13	13	13
	CPU	0.0129	0.0612	0.2009	0.5094
	α_{exp}	0.008	0.008	0.005	0.006
SOR-like	IT	9	9	9	9
	CPU	0.0087	0.0253	0.1080	0.1155
	α_{exp}	1.01	1	1	1.02
NI	IT	9	9	9	9
	CPU	0.0106	0.0265	0.0816	0.1277
	α_{exp}	1	1	1	1.01

4. Conclusions

In this paper, to accelerate the block-diagonal and anti-block-diagonal splitting (BAS) iteration method, we have presented a modified BAS (MBAS) iteration method to solve the absolute value equation (AVE). To guarantee the convergence of the MBAS method, we give some convergence conditions under certain conditions. Numerical experiments manifest that the MBAS method compared to some existing numerical methods (such as the SOR-like method [21] and the NI method [22]) is feasible for the AVE under certain conditions.

Acknowledgments

The authors would like to thank two anonymous referees for providing helpful suggestions, which greatly improved the paper. This research was supported by National Natural Science Foundation of China (No.11961082) and Key Project of Shaanxi Provincial Education Department under grant 20JS021.

Conflict of interest

The authors declare that they have no competing interests.

References

1. C. X. Li, S. L. Wu, Block-diagonal and anti-block-diagonal splitting iteration method for absolute value equation, In: *Simulation tools and techniques*, 12th EAI International Conference, SIMUtools 2020, Guiyang, China, **369** (2021), 572–581. doi: 0.1007/978-3-030-72792-5_45.
2. O. L. Mangasarian, A generalized Newton method for absolute value equations, *Optim. Lett.*, **3** (2009), 101–108. doi: 10.1007/s11590-008-0094-5.
3. Z. Z. Bai, X. Yang, On HSS-based iteration methods for weakly nonlinear systems, *Appl. Numer. Math.*, **59** (2009), 2923–2936. doi: 10.1016/j.apnum.2009.06.005.
4. M. Z. Zhu, Y. E. Qi, The nonlinear HSS-like iteration method for absolute value equations, *arXiv*. Available from: `https://arxiv.org/abs/1403.7013v4`.
5. J. Rohn, A theorem of the alternatives for the equation $Ax + B|x| = b$, *Linear Multilinear A.*, **52** (2004), 421–426. doi: 10.1080/0308108042000220686.
6. O. L. Mangasarian, Absolute value programming, *Comput. Optim. Applic.*, **36** (2007), 43–53. doi: 10.1007/s10589-006-0395-5.
7. O. L. Mangasarian, R. R. Meyer, Absolute value equations, *Linear Algebra Appl.*, **419** (2006), 359–367. doi 10.1016/j.laa.2006.05.004.
8. S. L. Wu, P. Guo, Modulus-based matrix splitting algorithms for the quasi-complementarity problems, *Appl. Numer. Math.*, **132** (2018), 127–137. doi: 10.1016/j.apnum.2018.05.017.
9. R. W. Cottle, J. S. Pang, R. E. Stone, *The linear complementarity problem*, Society for Industrial and Applied Mathematics, 2009. doi: 10.1137/1.9780898719000.
10. J. Rohn, An algorithm for solving the absolute value equations, *Electron. J. Linear Algebra*, **18** (2009), 589–599. doi: 10.13001/1081-3810.1332.
11. J. Rohn, V. Hooshyarbakhsh, R. Farhadsefat, An iterative method for solving absolute value equations and sufficient conditions for unique solvability, *Optim. Lett.*, **8** (2014), 35–44. doi: 10.1007/s11590-012-0560-y.
12. D. K. Salkuyeh, The Picard-HSS iteration method for absolute value equations, *Optim. Lett.*, **8** (2014), 2191–2202. doi: 10.1007/s11590-014-0727-9.
13. O. L. Mangasarian, A hybrid algorithm for solving the absolute value equation, *Optim. Lett.*, **9** (2015), 1469–1474. doi: 10.1007/s11590-015-0893-4.

14. S. L. Wu, T. Z. Huang, X. L. Zhao, A modified SSOR iterative method for augmented systems, *J. Comput. Appl. Math.*, **228** (2009), 424–433. doi: 10.1016/j.cam.2008.10.006.

15. C. X. Li, S. L. Wu, Modified SOR-like iteration method for absolute value equations, *Math. Probl. Eng.*, **2020** (2020), 9231639. doi: 10.1155/2020/9231639.

16. A. X. Wang, H. J. Wang, Y. K. Deng, Interval algorithm for absolute value equations, *Cent. Eur. J. Math.*, **9** (2011), 1171–1184. doi: 10.2478/s11533-011-0067-2.

17. S. Ketabchi, H. Moosaei, An efficient method for optimal correcting of absolute value equations by minimal changes in the right hand side, *Comput. Math. Appl.*, **64** (2012), 1882–1885. doi: 10.1016/j.camwa.2012.03.015.

18. C. Zhang, Q. J. Wei, Global and finite convergence of a generalized newton method for absolute value equations, *J. Optim. Theory. Appl.*, **143** (2009), 391–403. doi: 10.1007/s10957-009-9557-9.

19. C. X. Li, A preconditioned AOR iterative method for the absolute value equations, *Inter. J. Comput. Meth.*, **14** (2017), 1750016. doi: 10.1142/S0219876217500165.

20. J. M. Ortega, W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Academic Press, 1970. doi: 10.1016/C2013-0-11263-9.

21. Y. F. Ke, C. F. Ma, SOR-like iteration method for solving absolute value equations, *Appl. Math. Comput.*, **311** (2017), 195–202. doi: 10.1016/j.amc.2017.05.035.

22. Y. F. Ke, The new iteration algorithm for absolute value equation, *Appl. Math. Lett.*, **99** (2020), 105990. doi: 10.1016/j.aml.2019.07.021.

23. T. Saha, S. Srivastava, S. Khare, P. S. Stanimirovic, M. D. Petkovic, An improved algorithm for basis pursuit problem and its applications, *Appl. Math. Comput.*, **355** (2019), 385–398. doi: 10.1016/j.amc.2019.02.073.

24. M. D. Petkovic, Generalized Schultz iterative methods for the computation of outer inverses, *Comput. Math. Appl.*, **67** (2014), 1837–1847. doi: 10.1016/j.camwa.2014.03.019.

25. M. D. Petkovic, M. A. Krstic, K. P. Rajkovic, Rapid generalized Schultz iterative methods for the computation of outer inverses, *J. Comput. Appl. Math.*, **344** (2018), 572–584 doi: 10.1016/j.cam.2018.05.048.

26. M. Benzi, G. H. Golub, J. Liesen, Numerical solution of saddle point problems, *Acta Numer.*, **14** (2005), 1–137. doi: 10.1017/S0962492904000212.

27. Z. Z. Bai, Z. Q. Wang, On parameterized inexact Uzawa methods for generalized saddle point problems, *Linear Algebra Appl.*, **428** (2008), 2900–2932. doi: 10.1016/j.laa.2008.01.018.

AIMS Press