Computational and Applied Mathematics (2023) 42:19
https://doi.org/10.1007/s40314-022-02157-x

Relaxed-based matrix splitting methods for solving absolute value equations

Juan Song[1] · **Yongzhong Song**[2]

Received: 4 June 2022 / Revised: 22 November 2022 / Accepted: 10 December 2022 /
Published online: 24 December 2022

Abstract

In this paper, we investigate the iterative methods for solving the absolute value equations (AVEs). Using matrix splitting and the relaxed technique, a relaxed-based matrix splitting (RMS) method is presented. As special cases, we propose a relaxed-based Picard (RP) method, relaxed-based AOR (RAOR) method, and relaxed-based SOR (RSOR) method. These methods include some known methods as special cases, such as the Newton-based matrix splitting iterative method, the modified Newton type iteration method, the Picard method, a new SOR-like method, the fixed point iteration method, the SOR-like method, the AOR method, the modified SOR-like method, etc. Some convergence conditions of the proposed method are presented. Numerical examples verify the theoretical results and the advantages of the new methods.

Keywords Absolute value equations · Relaxed-based matrix splitting method · Relaxed-based Picard method · Relaxed-based AOR method · Relaxed-based SOR method · Convergence

Mathematics Subject Classification 65F10 · 90C05 · 90C30

Communicated by Jinyun Yuan.

✉ Yongzhong Song
yzsong@njnu.edu.cn

Juan Song
song12368@163.com

1 School of Science, Wuxi University, Wuxi 214105, People's Republic of China

2 Jiangsu Key Lab for NSLSCS, School of Mathematical Sciences, Nanjing Normal University, Nanjing 210023, People's Republic of China

1 Introduction

In this paper, we investigate the iterative methods for solving the absolute value equations (AVEs)

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

where $A, B \in R^{n\times n}$ with $B \neq 0, b, x \in R^n$, and $|x|$ denotes the vector with absolute values of components of x. When $B = I$, the AVEs (1) reduce to a special form

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

If B is nonsingular, then the AVEs (1) can be rewritten to $B^{-1}Ax - |x| = B^{-1}b$, which has the form (2).

The system (1) is first introduced in Rohn (2004). For the existence and uniqueness of the solution of (1) and (2), one can see (Hu and Huang 2010; Mangasarian 2009b; Mangasarian and Meyer 2006; Rohn 2004, 2009; Wu 2021; Wu and Li 2018). In Mangasarian (2007), Mangasarian and Meyer (2006), Mezzadri (2020), Rohn (2004), it has been proved that an AVEs is equivalent to a linear complementarity problem (LCP), which is NP-hard. Hence, they have proved that solving the AVEs is NP-hard.

In this paper, we always assume that the AVEs is solvable.

In recent years, under the conditions of the existence and uniqueness of the solution of the AVEs (1) and (2), a variety of efficient iterative methods have been presented, such as for solving the AVEs (1), a Picard method (Rohn et al. 2014), the generalized Newton method (Hu et al. 2011; Miao et al. 2015; Wang et al. 2019), a smoothing Newton method (Miao et al. 2017), the Newton-based matrix splitting iterative method (Zhou et al. 2021), the AOR method and a preconditioned AOR method (Li 2017), and for solving the AVEs (2) the generalized Newton method (Li 2016a; Mangasarian 2009a; Moosaei et al. 2015; Zhang and Wei 2009), the generalized Traubs method (Haghani 2015), the fixed point iteration method (Ke 2020; Yu et al. 2020), the SOR-like methods (Dong et al. 2020; Ke and Ma 2017; Li and Wu 2020), the Picard-HSS iteration method (Salkuyeh 2014), the nonlinear MHSS-like method (Li 2016b), the Picard-CSCS iteration method and the nonlinear CSCS-like iteration method (Gu et al. 2017), the Picard-HSS-SOR iteration method (Zheng 2020), a generalized and a preconditioned generalized Gauss-Seidel methods (Edalatpour et al. 2017), and so on.

In this paper, to improve computing efficiency, we present a relaxed-based matrix splitting method (RMS) for solving the AVEs (1). As special cases, we propose a relaxed-based Picard (RP) method, a relaxed-based AOR (RAOR) method, and a relaxed-based SOR (RSOR) method. These methods include some known methods as special cases, such as the Newton-based matrix splitting iterative method (NMS method) (Zhou et al. 2021), the modified Newton type iteration method (MN method) (Wang et al. 2019), the Picard method (Rohn et al. 2014), the new SOR-like method (Dong et al. 2020), the fixed point iteration (FPI) method (Ke 2020; Yu et al. 2020), the SOR-like method (Ke and Ma 2017), the AOR method (Li 2017), the modified SOR-like (MSOR-like) method (Li and Wu 2020), etc. Moreover, we prove convergence theorems of the proposed methods. Finally, we use numerical examples to demonstrate our theoretical analysis and the superiority of the RMS method.

This paper is organized as follows. In Sect. 2, some notations and lemmas are reviewed. In Sect. 3, we propose the RMS, RP, RAOR, and RSOR methods for solving the AVEs (1). In Sect. 4, the convergence analysis of the proposed method is presented. Numerical examples and conclusions are given in Sects. 5 and 6, respectively.

2 Preliminaries

In this section, some notations and auxiliary results are presented.

Let $A \in R^{n\times n}$ be the set of $n \times n$ matrices with real entries and $R^n = R^{n\times 1}$. The ith component of a vector $x \in R^n$ is denoted by x_i. Denote $|x|$ the vector with ith component equal to $|x_i|$. For matrix A, an expression $A = Q - R$ is called a splitting of A when Q is nonsingular. $\|A\|$ denotes the spectral norm defined by $\|A\| = \max\{\|Ax\| : x \in R^n, \|x\| = 1\}$, where $\|x\|$ is the 2-norm. For matrix $A = (a_{ij}), B = (b_{ij}) \in R^{n\times n}$, we say $A \geq B$ if $a_{ij} \geq b_{ij}, i, j = 1, \cdots, n$.

Lemma 1 *(Young 1971, Lemma 6-2.1) If b and c are real, then both roots of the quadratic equation*

$$x^2 + bx + c = 0$$

are less than one in modulus if and only if

$$|c| < 1, |b| < 1 + c.$$

The following results are obvious.

Lemma 2 *For $x, y \in R^n$, the following results hold:*

(i) $\||x| - |y|\| \leq \|x - y\|$;
(ii) *if* $0 \leq x \leq y$, *then* $\|x\| \leq \|y\|$;
(iii) *if* $x \leq y$ *and* $Q \geq 0$, *then* $Qx \leq Qy$.

3 Relaxed-based matrix splitting method

In this section, some new methods for solving the AVEs (1) are presented.

Let $y = |x|$. Then, the AVEs (1) is equivalent to a new AVEs

$$\begin{cases} Ax - By = b, \\ y = |x|. \end{cases}$$

We split A into

$$A = Q - R, \tag{3}$$

where Q is nonsingular. Then, we define a relaxed-based matrix splitting (RMS) method for solving the AVEs (1) as

$$\begin{cases} x^{k+1} = Q^{-1}(Rx^k + By^k + b), \\ y^{k+1} = (1-\tau)y^k + \tau|x^{k+1}|, \end{cases} \quad k = 0, 1, 2, \cdots, \tag{4}$$

where τ is a positive constant.

The RMS method has a general form and it contains many existing iterative methods as its special cases.

Let $Q = \frac{1}{\omega}A$, $R = \frac{1-\omega}{\omega}A$ with $\omega \neq 0$. Then, the RMS method defined by (4) reduces to a relaxed-based Picard (RP) method defined as

$$\begin{cases} x^{k+1} = (1-\omega)x^k + \omega A^{-1}(By^k + b), \\ y^{k+1} = (1-\tau)y^k + \tau|x^{k+1}|. \end{cases} \tag{5}$$

The RP method is simple in format and more efficient when A^{-1} is easy to calculate. Especially, when A is a strictly diagonally dominant matrix, we usually choose it to solve AVE.

For $\tau = 1$, the RMS method (4) reduces to the Newton-based matrix splitting iterative method (NMS method) (Zhou et al. 2021)

$$x^{k+1} = Q^{-1}(Rx^k + B|x^k| + b), \tag{6}$$

the modified Newton type iteration method (MN method) whenever R is a positive semi-definite matrix (Wang et al. 2019), and a Picard method for $R = 0$ (Rohn et al. 2014)

$$x^{k+1} = A^{-1}(B|x^k| + b). \tag{7}$$

Comparing with the NMS method (6), the RMS method (4) has a relaxation factor, so it is more effective.

We decompose A into

$$A = D - L - U,$$

where D is a diagonal matrix, and L and U are strictly lower and upper triangular matrices of A, respectively, as usual.

In Wang et al. (2019); Zhou et al. (2021), from the NMS method, the authors propose some special iterative methods, which include a special MN method

$$x^{k+1} = (A + \Omega)^{-1}(\Omega x^k + B|x^k| + b), \tag{8}$$

the Newton-based Gauss-Seidel (NGS) method

$$x^{k+1} = (D + \Omega - L)^{-1}[(\Omega + U)x^k + B|x^k| + b], \tag{9}$$

and the Newton-based SOR (NSOR) method

$$x^{k+1} = (D + \alpha\Omega - \alpha L)^{-1}\{[\alpha\Omega + (1-\alpha)D + \alpha U]x^k + \alpha(B|x^k| + b)\}. \tag{10}$$

For $\omega \in \Re \setminus \{0\}$ and $\gamma \in \Re$, let

$$A = Q_{\gamma,\omega} - R_{\gamma,\omega},$$

where

$$Q_{\gamma,\omega} = \frac{1}{\omega}(D - \gamma L),\ \ R_{\gamma,\omega} = \frac{1}{\omega}\left[(1-\omega)D + (\omega-\gamma)L + \omega U\right].$$

Then, the RMS method defined by (4) is called the relaxed-based AOR (RAOR) method defined as

$$\begin{cases} x^{k+1} = (D - \gamma L)^{-1}\{[(1-\omega)D + (\omega-\gamma)L + \omega U]x^k + \omega(By^k + b)\}, \\ y^{k+1} = (1-\tau)y^k + \tau|x^{k+1}|. \end{cases} \tag{11}$$

For $\tau = 1$, the RAOR method (11) turns into the AOR method (Li 2017):

$$x^{k+1} = (D - \gamma L)^{-1}\{[(1-\omega)D + (\omega-\gamma)L + \omega U]x^k + \omega(B|x^k| + b)\}. \tag{12}$$

Comparing with the AOR method (12), the RAOR method (11) has a relaxation factor, so it is more effective.

For the case when $\omega = \gamma$, the RAOR method can be called the relaxed-based SOR (RSOR) method

$$\begin{cases} x^{k+1} = (D - \omega L)^{-1}\{[(1-\omega)D + \omega U]x^k + \omega(By^k + b)\}, \\ y^{k+1} = (1-\tau)y^k + \tau|x^{k+1}|. \end{cases} \tag{13}$$

For the AVEs (2), some known methods can be derived as follows.

In this case, the RP method (5) is equivalent to the new SOR-like method (Dong et al. 2020), where $\tau = \omega/\sigma$ with $\sigma > 0$.

The RP method (5) reduces to the fixed point iteration (FPI) method whenever $\omega = 1$ (Ke 2020; Yu et al. 2020)

$$\begin{cases} x^{k+1} = A^{-1}(y^k + b), \\ y^{k+1} = (1-\tau)y^k + \tau|x^{k+1}|, \end{cases} \tag{14}$$

and the SOR-like method whenever $\tau = \omega$ (Ke and Ma 2017)

$$\begin{cases} x^{k+1} = (1-\omega)x^k + \omega A^{-1}(y^k + b), \\ y^{k+1} = (1-\omega)y^k + \omega|x^{k+1}|. \end{cases} \tag{15}$$

Comparing with the FPI and SOR-like methods, the RP method has more degrees of freedom, so it may be more effective.

The RSOR method (13) reduces to the modified SOR-like (MSOR-like) method whenever $\tau = \omega$ (Li and Wu 2020)

$$\begin{cases} x^{k+1} = (D-\omega L)^{-1}\{[(1-\omega)D + \omega U]x^k + \omega(y^k + b)\}, \\ y^{k+1} = (1-\omega)y^k + \omega|x^{k+1}|. \end{cases} \tag{16}$$

Comparing with the MSOR-like method, the degree of freedom of the RSOR method is bigger, so it may be more effective.

4 Convergence analysis

In this section, we discuss the convergence of the methods proposed in Sect. 3.

For the AVEs (1) and the splitting (3), denote

$$\mu = ||Q^{-1}R||, \ \nu = ||Q^{-1}B||. \tag{17}$$

Lemma 3 *If*

$$\mu|1-\tau| < 1, \ \tau\nu < (1-\mu)(1-|1-\tau|), \tag{18}$$

then the RMS method (4) for solving the AVEs (1) is convergent.

Proof Let $\{x^*, y^*\}$ be solution of the AVEs (1). Then, it holds that

$$\begin{cases} x^* = Q^{-1}Rx^* + Q^{-1}By^* + Q^{-1}b, \\ y^* = (1-\tau)y^* + \tau|x^*|. \end{cases} \tag{19}$$

Denote

$$\begin{cases} \varepsilon_k^x = x^k - x^*, \\ \varepsilon_k^y = y^k - y^*, \end{cases} \ k = 0, 1, 2, \cdots$$

Subtracting (19) from (4), we have

$$\begin{cases} \varepsilon_{k+1}^x = Q^{-1}R\varepsilon_k^x + Q^{-1}B\varepsilon_k^y, \\ \varepsilon_{k+1}^y = (1-\tau)\varepsilon_k^y + \tau(|x^{k+1}| - |x^*|), \end{cases}$$

so that

$$|\varepsilon_{k+1}^x| \leq |Q^{-1}R||\varepsilon_k^x| + |Q^{-1}B||\varepsilon_k^y|,$$

$$|\varepsilon_{k+1}^{y}| \le |1-\tau||\varepsilon_{k}^{y}| + \tau||x^{k+1}| - |x^{*}|| \le |1-\tau||\varepsilon_{k}^{y}| + \tau|\varepsilon_{k+1}^{x}|.$$

By Lemma 2, we have

$$||\varepsilon_{k+1}^{x}|| \le \mu||\varepsilon_{k}^{x}|| + \nu||\varepsilon_{k}^{y}||, \ ||\varepsilon_{k+1}^{y}|| \le |1-\tau|||\varepsilon_{k}^{y}|| + \tau||\varepsilon_{k+1}^{x}||.$$

This can be rewritten as

$$\begin{bmatrix} -\tau & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} ||\varepsilon_{k+1}^{x}|| \\ ||\varepsilon_{k+1}^{y}|| \end{bmatrix} \le \begin{bmatrix} 0 & |1-\tau| \\ \mu & \nu \end{bmatrix} \begin{bmatrix} ||\varepsilon_{k}^{x}|| \\ ||\varepsilon_{k}^{y}|| \end{bmatrix},$$

which implies that

$$\begin{aligned} \begin{bmatrix} ||\varepsilon_{k+1}^{x}|| \\ ||\varepsilon_{k+1}^{y}|| \end{bmatrix} &\le \begin{bmatrix} 0 & 1 \\ 1 & \tau \end{bmatrix} \begin{bmatrix} 0 & |1-\tau| \\ \mu & \nu \end{bmatrix} \begin{bmatrix} ||\varepsilon_{k}^{x}|| \\ ||\varepsilon_{k}^{y}|| \end{bmatrix} \\ &\le \begin{bmatrix} \mu & \nu \\ \tau\mu & |1-\tau| + \tau\nu \end{bmatrix} \begin{bmatrix} ||\varepsilon_{k}^{x}|| \\ ||\varepsilon_{k}^{y}|| \end{bmatrix} \\ &\le \cdots \le \begin{bmatrix} \mu & \nu \\ \tau\mu & |1-\tau| + \tau\nu \end{bmatrix}^{k+1} \begin{bmatrix} ||\varepsilon_{0}^{x}|| \\ ||\varepsilon_{0}^{y}|| \end{bmatrix}. \end{aligned} \tag{20}$$

Let

$$W = \begin{bmatrix} \mu & \nu \\ \tau\mu & |1-\tau| + \tau\nu \end{bmatrix},$$

and let λ be an eigenvalue of W. Then, we have

$$\det(\lambda I - W) = \det \begin{bmatrix} \lambda - \mu & -\nu \\ -\tau\mu & \lambda - |1-\tau| - \tau\nu \end{bmatrix} = 0.$$

That is

$$\lambda^{2} - (|1-\tau| + \tau\nu + \mu)\lambda + \mu|1-\tau| = 0.$$

By (18), it holds that

$$\mu|1-\tau| < 1, \ |1-\tau| + \tau\nu + \mu < 1 + \mu|1-\tau|.$$

It follows by Lemma 1 that $|\lambda| < 1$, which implies $\rho(W) < 1$.

Hence, $W^{k} \to 0$ ($k \to \infty$), so that, by (20), $\varepsilon_{k}^{x} \to 0$, $\varepsilon_{k}^{y} \to 0$ ($k \to \infty$). This has proved that the RMS method (4) is convergent. □

Theorem 4 *Suppose that* $\mu + \nu < 1$. *If*

$$0 < \tau < \frac{2(1-\mu)}{1-\mu+\nu}, \tag{21}$$

then the RMS method (4) for solving the AVEs (1) is convergent.

Proof If $0 < \tau \le 1$, then $\mu|1-\tau| = \mu(1-\tau) < 1$ and $\tau\nu < \tau(1-\mu) = (1-\mu)(1-|1-\tau|)$. Hence, the inequality (18) holds.

For $1 \le \tau < 2(1-\mu)/(1-\mu+\nu)$, on the one hand, it gets that $\tau\nu < (1-\mu)(2-\tau) = (1-\mu)(1-|1-\tau|)$, which shows that the second inequality in (18) holds. On the other hand, since $2\mu(1-\mu)/(1-\mu+\nu) < 1+\mu$, it derives that $\tau\mu < 1+\mu$, so that $\mu|\tau-1| = \mu(\tau-1) < 1$, which shows that the first inequality in (18) holds.

In a word, we have proved that if (21) is satisfied, then the inequality (18) holds. It follows by Lemma 3 that the RMS method (4) for solving the AVEs (1) is convergent. □

Theorem 5 *Let A be nonsingular and $\|A^{-1}B\| < 1$. Then, the RP method (5) for solving the AVEs (1) is convergent, whenever either*

$$0 < \omega \leq 1,\ \ 0 < \tau < \frac{2}{1 + \|A^{-1}B\|}$$

or

$$1 < \omega < \frac{2}{1 + \|A^{-1}B\|},\ \ 0 < \tau < \frac{2(2-\omega)}{2 - \omega + \omega\|A^{-1}B\|}.$$

Proof For the RP method, μ and ν defined by (17) reduce to $\mu = |1-\omega|$ and $\nu = |\omega|\|A^{-1}B\|$.

When $\|A^{-1}B\| < 1$ and $0 < \omega < 2/(1 + \|A^{-1}B\|)$, it is easy to prove that $\mu + \nu = |1-\omega| + \omega\|A^{-1}B\| < 1$. And it gets that

$$\frac{2(1-\mu)}{1-\mu+\nu} = \frac{2(1-|1-\omega|)}{1-|1-\omega| + \omega\|A^{-1}B\|}.$$

Now, by Theorem 4, the required result follows directly. □

When $\omega = 1$, by Theorem 5, the following corollary is obvious.

Corollary 6 *Let A be nonsingular. If*

$$\|A^{-1}\| < 1,\ \ 0 < \tau < \frac{2}{1 + \|A^{-1}\|},$$

then the FPI method (14) for solving the AVEs (2) is convergent.

This convergence result is better than the corresponding one given by (Ke 2020, Theorem 2.1).

It is easy to see that if $\tau = \omega$, then the condition in Theorem 5 reduces to

$$\|A^{-1}B\| < 1,\ \ 0 < \omega < \frac{2}{1 + \sqrt{\|A^{-1}B\|}}.$$

Hence, by Theorem 5, the following corollary is direct.

Corollary 7 *Let A be nonsingular. If*

$$\|A^{-1}\| < 1,\ \ 0 < \omega < \frac{2}{1 + \sqrt{\|A^{-1}\|}},$$

then the SOR-like method (15) for solving the AVEs (2) is convergent.

This convergence condition is simpler than the corresponding one given by Ke and Ma (2017).

Lemma 8 *(Li 2017, Lemma 4.1) If $\rho(|Q^{-1}R| + |Q^{-1}B|) < 1$, then the NMS method (6) for solving the AVEs (1) is convergent.*

This lemma implies (Zhou et al. 2021, Theorem 4.1) and is better than (Wang et al. 2019, Theorem 3.1).

Since the Picard method (7) is a special case of the NMS method, from Lemma 8, the following corollary is obvious.

Corollary 9 *If A is nonsingular and $\rho(|A^{-1}B|) < 1$, then the Picard method (7) for solving the AVEs (1) is convergent.*

This corollary is consistent with (Moosaei et al. 2015, Theorem 4.3) and is better than (Wang et al. 2019, Corollary 3.2). And the convergence given in (Rohn et al. 2014, Theorem 2) can be derived from this corollary directly.

From Theorem 4, the following convergence for the RAOR method (11) is direct.

Theorem 10 *Let* $\tilde{\mu} = ||(D-\gamma L)^{-1}[(1-\omega)D+(\omega-\gamma)L+\omega U]||$, $\tilde{\nu} = ||(D-\gamma L)^{-1}B||$. *Assume that* $\tilde{\mu}+\omega\tilde{\nu} < 1$. *If*

$$0 < \tau < \frac{2(1-\tilde{\mu})}{1-\tilde{\mu}+\omega\tilde{\nu}},$$

then the RAOR method (11) for solving the AVEs (1) is convergent.

When $\omega = \gamma$, we can obtain the convergence of the RSOR method (13).

For the MSOR-like method (16), this theorem is consistent with (Li and Wu 2020, Theorem 1).

It is easy to prove that if $\tilde{\mu}+\omega\tilde{\nu} < 1$ then

$$\frac{2(1-\tilde{\mu})}{1-\tilde{\mu}+\omega\tilde{\nu}} > 1.$$

Hence, by Theorem 10, the following result is immediately.

Corollary 11 *Let* $\tilde{\mu} = ||(D-\gamma L)^{-1}[(1-\omega)D+(\omega-\gamma)L+\omega U]||$, $\tilde{\nu} = ||(D-\gamma L)^{-1}B||$. *Assume that* $\tilde{\mu}+\omega\tilde{\nu} < 1$. *Then, the AOR method (12) for solving the AVEs (1) is convergent.*

5 Numerical examples

In this section, three numerical examples are tested to demonstrate the effectiveness and feasibility of our methods for solving the AVEs (1) and (2) from aspects of the iteration times, the elapsed computation time, and the relative residual error.

For convenience, the iteration times, the elapsed computation time (unit: second), and the relative residual error are, respectively, denoted as ‘IT’, ‘CPU’(unit: s) and ‘RES’. In the computation, all initial vector are taken as zero vectors, and numerical methods are terminated once the relative error satisfies $||Ax_k - B|x_k| - b||_2/||b||_2 \leq 10^{-6}$ or the CPU is more than 500 s. All the tests are performed under Matlab 2016b on a personal computer with 2.30GHZ central processing unit (Inter(R) Core(TM) i5-8259U), 8GB memory, and Windows 10 operating system.

Example 1 (Guo et al. 2019) Let m be a prescribed positive integer and $n = m^2$. Consider the AVEs (2) with

$$A = tridiag(-I, S, -I) = \begin{bmatrix} S & -I & 0 & \cdots & 0 & 0 \\ -I & S & -I & \cdots & 0 & 0 \\ 0 & -I & S & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & S & -I \\ 0 & 0 & 0 & \cdots & -I & S \end{bmatrix} \in R^{n\times n},$$

$$S = tridiag(-1, 8, -1) = \begin{bmatrix} 8 & -1 & 0 & \cdots & 0 & 0 \\ -1 & 8 & -1 & \cdots & 0 & 0 \\ 0 & -1 & 8 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 8 & -1 \\ 0 & 0 & 0 & \cdots & -1 & 8 \end{bmatrix} \in R^{m\times m},$$
$$x^* = (-1, 1, -1, 1, \cdots, -1, 1)^\top \in R^n, \; b = Ax^* - |x^*|.$$

We test the RP method (5) with the experimental optimal parameters (ω, τ), the SOR-like method (15) with the optimal parameter given in Guo et al. (2019), the RAOR method (11) with the parameters $(\omega, \gamma, \tau) = (0.9, 0.85, 1.6)$, the AOR method (12) with the parameters $(\omega, \gamma) = (0.9, 0.85)$, the MN method (8), the NGS method (9), and the NSOR method (10) with the experimental optimal parameter, respectively. For the MN, NGS, and NSOR methods, Ω is taken as two common cases. The experimental results are listed in Table 1.

From Table 1, we can see that these seven tested methods can fast converge to the solution x^* for different sizes. When the matrix dimension increases, it is almost constant, which shows that these seven tested methods are stable. Furthermore, we can see that the RP, SOR-like, and RAOR methods have less CPU and IT than the other methods. In terms of the elapsed CPU and IT, the RP method requires the least computing time and iteration times.

In Mangasarian and Meyer (2006), the linear complementarity problem (LCP)

$$Mz + q \geq 0, z \geq 0, z^\top(Mz + q) = 0 \tag{22}$$

with $M \in R^{n\times n}$, $det(I - M) \neq 0$ and $q \in R^n$, can be reduced to the AVEs

$$(M + I)x - (M - I)|x| = q \tag{23}$$

with $x = \frac{1}{2}[(M - I)z + q]$.

Example 2 (Zhou et al. 2021) Consider the LCP (22), where $M = \hat{M} + 4I$ with

$$\hat{M} = tridiag(-I, S, -I) \in R^{n\times n}, \; S = tridiag(-1, 4, -1) \in R^{m\times m},$$

and $q = -Mz^*$ with $z^* = (1.2, 1.2, \cdots, 1.2)^\top \in R^n$ being its unique solution. In this case, the unique solution of the corresponding AVEs (23) is

$$x^* = (-0.6, -0.6, \cdots, -0.6)^\top \in R^n.$$

We test the RP method (5) with the parameters $(\omega, \tau) = (0.6, 0.998)$ and $(\omega, \tau) = (0.998, 0.598)$, the Picard method (7), the MN method (8), the NGS method (9), and the NSOR method (10) with the optimal parameter $\alpha = 0.9$. For the MN, NGS, and NSOR methods, set $\Omega = \hat{M}$.

Some iterative methods for solving the LCP (22) have been proposed. In Huang and Cui (2022), the authors given a class of RMMS method, in which the RMSOR method is the more effective one. Here, we test the RMSOR method, where $\Omega = 0.5I$, $\gamma = 1$, and the parameter in SOR splitting and relaxation parameter θ are, respectively, selected as the experimental optimal parameters 0.6 and 0.98.

The experimental results are listed in Table 2, where '-' denotes the CPU time larger than 500 s.

From Table 2, we can see that the Picard method does not converge within 500 s, and the other six tested methods can fast converge to the solution x^* for different sizes. When the matrix dimension increases, IT is constant, which shows that the other six tested methods

Table 1 Numerical results of Example 1

	m	8×8	16×16	32×32	64×64
RP	(ω, τ)	(0.98,1.03)	(0.98,1.02)	(0.98,1.02)	(0.98,1.02)
	IT	9	9	9	9
	CPU	0.0041	0.0176	0.3356	11.1864
	RES	$9.2021e^{-7}$	$8.5694e^{-7}$	$9.2389e^{-7}$	$9.5740e^{-7}$
SOR-like	(ω, ω)	(1.0671,1.0671)	(1.0704,1.0704)	(1.0714,1.0714)	(1.0717,1.0717)
	IT	13	13	13	13
	CPU	0.0053	0.0253	0.5124	17.1123
	RES	$5.0322e^{-7}$	$7.5853e^{-7}$	$8.7749e^{-7}$	$9.2821e^{-7}$
RAOR	IT	14	14	14	14
	CPU	0.0063	0.0241	0.5231	20.4936
	RES	$4.5378e^{-7}$	$7.2882e^{-7}$	$8.6364e^{-7}$	$9.3050e^{-7}$
AOR	IT	17	17	17	17
	CPU	0.0085	0.0305	0.6385	25.3445
	RES	$4.7171e^{-7}$	$7.4878e^{-7}$	$8.8460e^{-7}$	$9.3050e^{-7}$
MN	Ω	$0.5 * diag(A)$	$0.5 * diag(A)$	$0.5 * diag(A)$	$0.5 * diag(A)$
	IT	21	22	22	22
	CPU	0.0067	0.0389	0.9235	30.8884
	RES	$9.0245e^{-7}$	$6.4895e^{-7}$	$7.3282e^{-7}$	$9.5201e^{-7}$
	Ω	$diag(A)$	$diag(A)$	$diag(A)$	$diag(A)$
	IT	33	34	34	35
	CPU	0.0069	0.0630	1.4164	50.2641
	RES	$8.7680e^{-7}$	$8.3264e^{-7}$	$9.5132e^{-7}$	$7.0006e^{-7}$
NGS	Ω	$0.5 * diag(A)$	$0.5 * diag(A)$	$0.5 * diag(A)$	$0.5 * diag(A)$
	IT	26	27	27	27
	CPU	0.0067	0.0340	0.9642	37.6451
	RES	$6.9223e^{-7}$	$6.3687e^{-7}$	$7.4380e^{-7}$	$7.9705e^{-7}$
	Ω	$diag(A)$	$diag(A)$	$diag(A)$	$diag(A)$
	IT	38	39	39	39
	CPU	0.0094	0.0485	1.4684	56.4281
	RES	$7.4126e^{-7}$	$7.9015e^{-7}$	$9.1848e^{-7}$	$9.8249e^{-7}$
NSOR	Ω	$0.5 * diag(A)$	$0.5 * diag(A)$	$0.5 * diag(A)$	$0.5 * diag(A)$
	α	1.22	1.22	1.22	1.22
	IT	21	22	22	22
	CPU	0.0076	0.0367	0.8829	30.7969
	RES	$9.5619e^{-7}$	$7.8885e^{-7}$	$9.2134e^{-7}$	$9.8327e^{-7}$
	Ω	$diag(A)$	$diag(A)$	$diag(A)$	$diag(A)$
	α	1.22	1.22	1.22	1.22
	IT	33	34	35	35
	CPU	0.0103	0.0522	1.3352	48.5387
	RES	$9.4268e^{-7}$	$9.5474e^{-7}$	$7.7609e^{-7}$	$8.3095e^{-7}$

Table 2 Numerical results of Example 2

	n	3600	4900	6400	8100	10000
RP	IT	5	5	5	5	5
	(ω, τ)	(0.6,0.998)	(0.6,0.998)	(0.6,0.998)	(0.6,0.998)	(0.6,0.998)
	CPU	3.8662	9.0153	20.0840	38.8365	71.7079
	RES	$5.9447e^{-7}$	$5.6943e^{-7}$	$5.5102e^{-7}$	$5.3694e^{-7}$	$5.2585e^{-7}$
	(ω, τ)	(0.998,0.598)	(0.998,0.598)	(0.998,0.598)	(0.998,0.598)	(0.998,0.598)
	IT	6	6	6	6	6
	CPU	4.6897	10.7683	24.7143	47.6933	98.6913
	RES	$1.0528e^{-7}$	$1.0406e^{-7}$	$1.0332e^{-7}$	$1.0284e^{-7}$	$1.0252e^{-7}$
Picard	IT	–	–	–	–	–
	CPU	–	–	–	–	–
	RES	–	–	–	–	–
MN	IT	28	28	28	28	–
	CPU	26.2930	67.9872	152.5700	408.3453	–
	RES	$8.4789e^{-7}$	$8.7256e^{-7}$	$8.9115e^{-7}$	$9.0567e^{-7}$	–
NGS	IT	10	10	10	10	10
	CPU	9.2301	21.2518	45.6189	117.1443	226.7960
	RES	$4.4613e^{-7}$	$6.0898e^{-7}$	$3.7969e^{-7}$	$3.5586e^{-7}$	$3.3599e^{-7}$
NSOR	IT	8	8	8	8	8
	CPU	7.0860	16.4593	35.2885	71.4078	161.1710
	RES	$2.5137e^{-7}$	$2.2340e^{-7}$	$2.0211e^{-7}$	$1.8533e^{-7}$	$1.7174e^{-7}$
RMSOR	IT	13	13	13	13	13
	CPU	12.7717	28.1610	55.1909	112.0169	445.8347
	RES	$6.2102e^{-7}$	$6.3276e^{-7}$	$6.4167e^{-7}$	$6.4867e^{-7}$	$6.5431e^{-7}$

are all effective and stable except the MN method. Further, we can see that the RP, NGS, and NSOR methods have less CPU and IT than the other methods. In terms of the elapsed CPU and IT, the RP method requires the least computing time and iteration times.

Example 3 Consider the AVEs (1) with the matrix A and B are randomly generated by the following matlab procedure: $A = rand(n, n) + n * eye(n) \in R^{n\times n}$, $B = rand(n, n) + \frac{n}{2} * eye(n) \in R^{n\times n}$, $b = ones(n, 1) \in R^{n\times 1}$.

We test the RAOR method (11), the Picard method (7), and the MSOR-like method (16), where we choose five increasing sizes: $n = 4000, 5000, 6000, 7000, 8000$. The experimental results are listed in Table 3, where '–' denotes the CPU time larger than 500 s.

From Table 3, we can see that the MSOR-like method does not converge within 500 s, and the RAOR method has less CPU and IT than the Picard method. This shows that the RAOR method is superior to the Picard and MSOR-like methods under some conditions.

Table 3 Numerical results of Example 3

	n	4000	5000	6000	7000	8000
RAOR	IT	29	29	29	29	29
	CPU	38.2336	72.9921	125.9768	199.7599	341.1727
	RES	$7.8293e^{-7}$	$7.8299e^{-7}$	$7.8340e^{-7}$	$7.8343e^{-7}$	$7.8577e^{-7}$
Picard	IT	36	36	36	36	36
	CPU	64.2573	96.3039	169.8457	272.4404	448.7433
	RES	$6.8529e^{-7}$	$6.8775e^{-7}$	$6.8672e^{-7}$	$6.8729e^{-7}$	$6.8755e^{-7}$
MSOR-like	IT	–	–	–	–	–
	CPU	–	–	–	–	–
	RES	–	–	–	–	–

6 Conclusions

In this paper, we investigate the iterative methods for solving the AVEs (1). Using matrix splitting and the relaxed technique, a relaxed-based matrix splitting (RMS) method is presented. As special cases, we propose the RP, RAOR, and RSOR methods. These methods include some known methods as special cases, such as the Newton-based matrix splitting iterative method (NMS method) (Zhou et al. 2021), the modified Newton type iteration method (MN method) (Wang et al. 2019), the Picard method (Rohn et al. 2014), the NSOR method (Dong et al. 2020), the fixed point iteration (FPI) method (Ke 2020; Yu et al. 2020), the SOR-like method (Ke and Ma 2017), the AOR method (Li 2017), the modified SOR-like (MSOR-like) method (Li and Wu 2020), etc.

We prove convergence theorems of the proposed methods. Numerical examples show that our methods are superior to the Picard, SOR-like, MN, NGS, AOR, NSOR, MSOR-like methods and the RMSOR method given in Huang and Cui (2022) under some conditions.

For the RMS method, how to construct more effective splitting and make the method converge faster still needs further research.

With reference to Li (2017), we can construct a preconditioned relaxed-based matrix splitting method.

For the RMS, RP, RAOR, and RSOR methods, there are some parameters involved. For the selection of parameters, trial calculation is mainly used. How to select better parameters is an interesting and important problem, and of course, it is also a difficult problem, which deserves further study.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant Nos. 11771213 and 11971242), the Priority Academic Program Development of Jiangsu Higher Education Institutions, and Wuxi University Research Start-up Fund for Introduced Talents. The authors would like to thank the referees for their many valuable suggestions and comments which led us to improve this paper.

References

Dong X, Shao X-H, Shen H-L (2020) A new SOR-like method for solving absolute value equations. Appl Numer Math 156:410–421

Edalatpour V, Hezari D, Salkuyeh DK (2017) A generalization of the Gauss–Seidel iteration method for solving absolute value equations. Appl Math Comput 293:156–167

Guo P, Wu S-L, Li C-X (2019) On the SOR-like method for solving absolute value equations. Appl Math Lett 97:107–113
Gu X, Huang T, Li H, Wang S, Li L (2017) Two CSCS-based iteration methods for solving absolute value equations. J Appl Anal Comput 7:1336–1356
Haghani FK (2015) On generalized Traubs method for absolute value equations. J Optim Theory Appl 166:619–625
Hu S-L, Huang Z-H (2010) A note on absolute value equations. Optim Lett 4:417–424
Hu S-L, Huang Z-H, Zhang Q (2011) A generalized Newton method for absolute value equations associated with second order cones. J Comput Appl Math 235:1490–1501
Huang Z-G, Cui J-J (2022) The relaxation modulus-based matrix splitting iteration method for horizontal linear complementarity problems. Bull Iran Math Soc. https://doi.org/10.1007/s41980-022-00695-y
Ke Y-F (2020) The new iteration algorithm for absolute value equation. Appl Math Lett 99:105990
Ke Y-F, Ma C-F (2017) SOR-Like iteration method for solving absolute value equations. Appl Math Comput 311:195–202
Li C-X (2016) A modified generalized Newton method for absolute value equations. J Optim Theory Appl 170:1055–1059
Li C-X (2016) On the modified Hermitian and skew-Hermitian splitting iteration methods for a class of weakly absolute value equations. J Inequal Appl 260:1–10
Li C-X (2017) A preconditioned AOR iterative method for the absolute value equations. Int J Comput Methods 14:1–12
Li C-X, Wu S-L (2020) Modified SOR-Like method for absolute value equations. Math Probl Eng 2020:9231639
Mangasarian OL (2007) Absolute value programming. Comput Optim Appl 36:43–53
Mangasarian OL (2009) A generalized Newton method for absolute value equations. Optim Lett 3:101–108
Mangasarian OL (2009) On unique solvability of the absolute value equation. Optim Lett 3:603–606
Mangasarian OL, Meyer RR (2006) Absolute value equations. Linear Algebra Appl 419:359–367
Mezzadri F (2020) On the solution of general absolute value equations. Appl Math Lett 107:106462
Miao X-H, Yang J-T, Hu S (2015) A generalized Newton method for absolute value equations associated with circular cones. Appl Math Comput 269:155–168
Miao X-H, Yang J-T, Saheya B, Chen J-S (2017) A smoothing Newton method for absolute value equation associated with second-order cone. Appl Numer Math 120:82–96
Moosaei H, Ketabchi S, Noor MA, Iqbal J, Hooshyarbakhsh V (2015) Some techniques for solving absolute value equations. Appl Math Comput 268:696–705
Rohn J (2004) A theorem of alternatives for the equation $Ax + B|x| = b$. Linear Multilinear Algebra 52:421–426
Rohn J (2009) On unique solvability of the absolute value equation. Optim Lett 3:603–606
Rohn J, Hooshyarbakhsh V, Farhadsefat R (2014) An iterative method for solving absolute value equations and sufficient conditions for unique solvability. Optim Lett 8:35–44
Salkuyeh DK (2014) The Picard-HSS iteration method for absolute value equations. Optim Lett 8:2191–2202
Wang A, Cao Y, Chen J-X (2019) Modified Newton-type iteration methods for generalized absolute value equations. J Optim Theory Appl 181:216–230
Wu S-L (2021) The unique solution of a class of the new generalized absolute value equation. Appl Math Lett 116:107029
Wu S-L, Li C-X (2018) The unique solution of the absolute value equations. Appl Math Lett 76:195–200
Young DM (1971) Iterative solution of large linear systems. Academic Press, New York
Yu D, Chen C, Han D (2020) A modified fixed point iteration method for solving the system of absolute value equations. Optimization 71:449–461
Zhang C, Wei Q (2009) Global and finite convergence of a generalized Newton method for absolute value equations. J Optim Theory Appl 143:391–403
Zheng L (2020) The Picard-HSS-SOR iteration method for absolute value equations. J Inequal Appl 258:1–9
Zhou H-Y, Wu S-L, Li C-X (2021) Newton-based matrix splitting method for generalized absolute value equation. J Comput Appl Math 394:113578