Comp. Appl. Math. (2018) 37:221–233
https://doi.org/10.1007/s40314-016-0338-4

A relaxed two-step splitting iteration method for computing PageRank

Ya-Jun Xie[1] · **Chang-Feng Ma**[2]

Received: 18 November 2015 / Revised: 21 March 2016 / Accepted: 1 April 2016 /
Published online: 21 April 2016

Abstract In this paper, we extend the two-step matrix splitting iteration approach by introducing a new relaxation parameter. The main idea is based on the inner–outer iteration for solving the PageRank problems proposed by Gleich et al. (J Sci Comput 32(1): 349–371, 2010) and Bai (Numer Algebra Cont Optim 2(4): 855–862, 2012) and the two-step splitting iteration presented by Gu et al. (Appl Math Comput 271: 337–343, 2015). The theoretical analysis results show that the proposed method is efficient. Numerical experiments demonstrate that the convergence performances of the method are better than the existing methods.

Keywords Two-step splitting iteration · Inner–outer iteration · Relaxation parameter · PageRank · Numerical test

Mathematics Subject Classification 65F10 · 65F30

1 Introduction

Consider the solution of the eigenvalue problem with the following form:

$$Ax = x, \quad A = \alpha P + (1-\alpha)ve^T, \tag{1.1}$$

Communicated by Jinyun Yuan.

The Project was supported by the National Natural Science Foundation of China (Grant Nos. 11071041, 11201074), Fujian Natural Science Foundation (Grant Nos. 2016J01005, 2015J01578), and Outstanding Young Training Plan of Fujian Province universities (Grant No. 15kxtz13, 2015).

✉ Chang-Feng Ma
macf@fjnu.edu.cn

1 Department of Mathematics and Physics, Fujian Jiangxia University, Fuzhou 350108, China

2 School of Mathematics and Computer Science, Fujian Normal University, Fuzhou 350117, China

where P is the column-stochastic matrix with appropriate dimension, i.e., its entries are nonnegative real numbers and all its columns sum to one. $\alpha \in (0, 1)$ denotes the damping factor which determines the weight given to the Web link graph. e is a column vector with all the ones, and v is a personalization vector or a teleportation vector. x is the eigenvector to be determined.

The above linear system (1.1), the so-call PageRank problem, occurs when one determines the importance of the Web pages based on the graph of the Web. PageRank can be viewed as a stationary distribution of a Markov chain (Langville and Meyer 2005). The PageRank has been widely exploited by Google as part of its search engine technology with the rapid development of the Internet. Although the exact ranking techniques and calculation approaches used by Google have been improved gradually, the PageRank method still receives great attention and becomes a hot issue in the field of science and engineering calculation in the last years.

At present, the method for solving the model (1.1) can be classified as a direct method and iterative method. The direct method often involves a lot of storage of the filling elements; especially when the coefficient matrix and the condition number are very large, the stability of the direct method is weak. In view of the sparsity of the matrix P, one can take advantage of the sparse structure to reduce the storage space and computation. The iterative method has become a popular approach for solving the eigenvalue problem. Sauer developed the Power method to compute PageRank, as it converges for every choice of a nonnegative starting vector (Sauer 2005). But, when the damping factor α is large, the Power method easily causes a low efficiency in the convergence performance. An extrapolation method was proposed, which speeds up the convergence by calculating and then subtracting off estimates of the contributions of the second and third eigenvectors (Kamvar et al. 2003). Also, Kamvar et al. (2004) proposed the adaptive methods to improve the computation of PageRank, in which the PageRank of pages that have converged are not recomputed per iteration. PageRank-type algorithms are used in application areas other than Web search (Morrison et al. 2005). Capizzano considered the Jordan canonical form of the Google matrix, which is a potential contribution to the PageRank computation (Capizzano 2005). The Krylov subspace methods have also been considered recently. Jia presented the restarted refined Arnoldi method (Jia 1997) which can be regarded as the variant of the Arnoldi-type algorithm proposed by Golub et al. in (2006). A lot of approaches have been given; see the book (Langville and Meyer 2006), which provides a more comprehensive description of the PageRank problem, the papers (Bai et al. 2004, 1996; Bai 2012; Berkhin 2005; Gleich et al. 2010; Langville and Meyer 2005, 2006; Page et al. 1999; Yin et al. 2012; Golub and Van Loan 1996; Wu and Wei 2007; Kamvar et al. 2003; Huang and Ma 2015; Fan and Simpson 2015; Skardal et al. 2015), which contain many additional results and useful references, and the book (Stewart 1994), which includes overviews of the Markov chains.

Gu et al. (2015) developed a two-step matrix splitting iteration (TSS) for computing PageRank which is based on the inter–outer iteration (IOI) approach proposed by Gleich et al. in (2010) and Bai in (2012). Inspired by these works, in this paper, we construct a new relaxed two-step splitting iteration method (RTSS) for computing PageRank, which can be showed as a more efficient and flexible technique for this problem.

The remainder of the paper is organized as follows. In Sect. 2, we first give a brief review of the inter–outer iteration and two-step matrix splitting iteration methods. Also, consider a new relaxed two-step splitting iteration method for computing PageRank (1.1). In Sect. 3, we provide the convergence analysis in detail. Some numerical experiments are given to illustrate that the new method is efficient in Sect. 4. Finally, we end the paper with some conclusions in Sect. 5.

2 The RTSS method

In this section, we review the inner–outer iteration and two-step matrix splitting iteration approaches for computing the PageRank problem.

It is obvious that the eigenvector problem (1.1) is equivalent to solve the following linear system

$$(I - \alpha P)x = (1 - \alpha)v, \tag{2.1}$$

where I denotes the identity matrix with the corresponding dimension clear from the context.

By (2.1), it is known that the smaller the damping parameter α, the easier it is to solve the problem. In view of this consideration, Gleich et al. presented the inner–outer iteration approach for solving the system (2.1) (Gleich et al. 2010). Now, we give a brief review of this approach.

The coefficient matrix of (2.1) can be written as the following splitting:

$$\mathcal{A} := (I - \alpha P) = M_1 - N_1, \tag{2.2}$$

where $M_1 = I - \beta P,\ N_1 = (\alpha - \beta)P$. Then, the fixed point equation of (2.1) gives

$$(I - \beta P)x = (\alpha - \beta)Px + (1 - \alpha)v.$$

So the stationary outer iteration scheme

$$(I - \beta P)x^{k+1} = (\alpha - \beta)Px^k + (1 - \alpha)v, \quad k = 0, 1, 2 \ldots,$$

where $0 < \beta < \alpha$. Moreover, the inner Richardson iteration

$$y^{j+1} = \beta P y^j + f$$

is considered to speed up the outer iteration, where $f = (\alpha - \beta)Px^k + (1 - \alpha)v,\ j = 0, 1, 2 \ldots, l - 1,\ y^0 = x^k$. The l-th step inner solution y^l is assigned to be the next new x^{k+1}. The inner–outer iteration method is listed following Algorithm 2.1.

Algorithm 2.1 (*The inner-outer iteration* (IOI) *method*)

```
Step 1  Input $P$, $\alpha$, $\beta$, $v$, $\tau$, $\eta$. Compute $x^0 = v$, $z^0 = Px^0$. Set $k := 0$.
Step 2  while $\|\alpha z^k + (1-\alpha)v - x^k\|_1 \geq \tau$, update
            $f = (\alpha - \beta)z^k + (1-\alpha)v,$
        otherwise, go to Step 3.
Step 3  update
            $x^{k+1} = f + \beta z^k, \quad z^{(k+1)} = Px^{k+1},$
        until $\|f + \beta z^{k+1} - x^{k+1}\|_1 \geq \eta$, go to Step 4.
Step 4  Set $k := k + 1$, return to Step 2.
```

Based on the inner–outer iteration method, Gu et al. proposed a two-step splitting iteration (TSS) for PageRank which is closely related to the power method and Richardson iteration. It is shown in the following Algorithm 2.2

```
Algorithm 2.2 (The two-step splitting iteration (TSS) method)
    Step 1  Input $P$, $\alpha$, $\beta$, $v$, $\tau$, $\eta$. Compute $x^0 = v$, $z^0 = Px^0$. Set $k := 0$.
    Step 2  while $\|\alpha z^k + (1-\alpha)v - x^k\|_1 \geq \tau$, update
                $x^{k+1} = \alpha z^k + (1-\alpha)v, \quad z^{k+1} = Px^{k+1}$,      (2.3)
                $f = (\alpha - \beta)z^{k+1} + (1-\alpha)v$,      (2.4)
        otherwise, go to Step 3.
    Step 3  update
                $x^{k+2} = f + \beta z^{k+1}, \quad z^{k+2} = Px^{k+2}$,
          until $\|f + \beta z^{k+2} - x^{k+2}\|_1 \geq \eta$, go to Step 4.
    Step 4  Set $k := k+1$, return to Step 2.
```

In fact, an iteration updating (2.3) was considered before the vector f can be computed in step 2 of the IOI method in Algorithm 2.1. It has been shown that the TSS method derived better performances than the IOI method. Inspired by Gleich et al. (2010), Gu et al. (2015), we further consider these methods by introducing a new relaxed parameter γ, which can accelerate the TSS method to a certain extent. Specially, the TSS method only involves the stationary outer iteration scheme, namely, the inner iteration must be a precise iteration. In Algorithm 2.2, the inner loop is an inaccurate iteration. The convergence of the TSS method was shown by Theorem 2 in Gu et al. (2015); however, the inner loop that leads to Algorithm 2.2 is a non-stationary iteration, which has been ignored. So, in the paper, we will discuss the situation in detail.

Now, we consider the coefficient matrix $\mathcal{A}$ in the following splitting:

$$\mathcal{A} = (I - \alpha P) = M_2 - N_2, \tag{2.3}$$

where $M_2 = \gamma I - \beta P$, $N_2 = (\gamma - 1)I + (\alpha - \beta)P$, $\gamma > 0$. The relaxed two-step splitting iteration is described in the Algorithm 2.3.

Algorithm 2.3 (*The relaxed two-step splitting iteration* (RTSS) *method*)

```
Step 1  Input $P$, $\alpha$, $\beta$, $\gamma$, $v$, $\tau$, $\eta$. Compute $x^0 = v$, $z^0 = Px^0$. Set $k := 0$.
Step 2  while $\|\alpha z^k + (1-\alpha)v - x^k\|_1 \geq \tau$, update
            $x^{k+1} = \alpha z^k + (1-\alpha)v, \quad z^{k+1} = Px^{k+1},$
            $f = \frac{\gamma-1}{\gamma}x^{k+1} + \frac{\alpha-\beta}{\gamma}z^{k+1} + \frac{1-\alpha}{\gamma}v,$
        otherwise, go to Step 3.
Step 3  update
            $x^{k+2} = f + \frac{\beta}{\gamma}z^{k+1}, \quad z^{k+2} = Px^{k+2},$
        until $\|f + \beta z^{k+2} - x^{k+2}\|_1 \geq \eta$, go to Step 4.
Step 4  Set $k := k+1$, return to Step 2.
```

Remark 2.1 In fact, if we set $\gamma = 1$, the RTSS is reduced to the TSS, so the RTSS approach is a generalized version of the TSS method. The flexible choice of the relaxation parameter γ of the RTSS generates excellent convergence performances than the TSS method, which can be seen in our numerical test parts.

Remark 2.2 Algorithm 2.3 is generated by the stationary iteration scheme, so it can be extended to study the absolute value equation (AVE) $Ax + |x| = b$ (Mangasarian and Meyer 2006; Rohn 2012; Moosaei et al. 2015; Yong 2016; Yong et al. 2011), which can also be expressed in the form of a stationary equation. The importance of the AVE derives from the fact that quadratic programs, linear programs, bimatrix games and many other problems can all be reduced to a linear complementarity problem (LCP) (Cottle and Dantzig 1968; Cottle et al. 1992) which is equivalent to the AVE. Furthermore, we know that the AVE is NP-hard. It was testified in (Mangasarian 2007) by reducing the LCP corresponding to the NP-hard knapsack feasibility problem to an AVE. Under suitable hypotheses, such as when the singular values of the coefficient matrix A exceed 1, some efficient approaches can be utilized to solve the problem, for instance, generalized Newton method. In our future work, we will devote ourselves to investigate the AVE by the proposed method or its improved version.

3 The convergence properties

In this section, we discuss the convergence property of Algorithm 2.3 in detail. To this end, we rewrite Algorithm 2.3 as follows:

$$\begin{cases} x^{k+\frac{1}{2}} = \alpha P x^k + (1-\alpha)v, \quad x^{k,0} = x^{k+\frac{1}{2}}, \\ x^{k,j+1} = \frac{\beta}{\gamma} P x^{k,j} + \frac{\gamma-1}{\gamma} x^{k+\frac{1}{2}} + \frac{\alpha-\beta}{\gamma} P x^{k+\frac{1}{2}} + \frac{1-\alpha}{\gamma} v, \quad j = 0, 1, 2, \ldots, m_k - 1, \\ x^{k+1} = x^{k,m_k}, \quad k = 0, 1, 2, \ldots. \end{cases} \tag{3.1}$$

Theorem 3.1 *Let parameters* $0 < \beta < \alpha < \gamma \le 1$ *and* $1 - \alpha < \gamma$, m_k *be the inner loop numbers for the k-th outer iteration. Algorithm 2.3 generates the iteration scheme*

$$x^{k+1} = T^k x^k + (1-\alpha) Q^k v, \quad k = 0, 1, 2, \ldots, \tag{3.2}$$

where

$$\begin{cases} T^k = \frac{\alpha\gamma}{\beta}(\frac{\beta}{\gamma}P)^{m_k+1} + \frac{\alpha(\gamma-1)}{\beta}\Sigma_{j=1}^{m_k}(\frac{\beta}{\gamma}P)^j + \frac{\alpha(\alpha-\beta)}{\beta}\Sigma_{j=1}^{m_k}(\frac{\beta}{\gamma}P)^j P, \\ Q^k = (\frac{\beta}{\gamma}P)^{m_k} + \Sigma_{j=0}^{m_k-1}(\frac{\beta}{\gamma}P)^j + \frac{\alpha-\beta}{\beta}\Sigma_{j=1}^{m_k}(\frac{\beta}{\gamma}P)^j. \end{cases} \tag{3.3}$$

Then, the iteration solution x^k *generated by Algorithm 2.3 converges to the exact solution* x^* *of the PageRank problem (1.1).*

Proof From (3.1), we obtain

$$\begin{aligned} x^{k+1} &= x^{k,m_k} \\ &= \frac{\beta}{\gamma}Px^{k,m_k-1} + \frac{(\gamma-1)I + (\alpha-\beta)P}{\gamma}\big(\alpha P x^k + (1-\alpha)v\big) + \frac{1-\alpha}{\gamma}v \\ &= \frac{\beta}{\gamma}P\left[\frac{\beta}{\gamma}Px^{k,m_k-2} + \frac{(\gamma-1)I + (\alpha-\beta)P}{\gamma}\big(\alpha P x^k + (1-\alpha)v\big) + \frac{1-\alpha}{\gamma}v\right] \\ &\quad + \frac{(\gamma-1)I + (\alpha-\beta)P}{\gamma}\big(\alpha P x^k + (1-\alpha)v\big) + \frac{1-\alpha}{\gamma}v \\ &= \left(\frac{\beta}{\gamma}P\right)^2 x^{k,m_k-2} + \left[\frac{\gamma-1}{\gamma}\left(\frac{\beta}{\gamma}P + I\right) + \frac{\alpha-\beta}{\gamma}\left(\frac{\beta}{\gamma}P + I\right)P\right]\alpha P x^k \\ &\quad + [\frac{\gamma-1}{\gamma}\left(\frac{\beta}{\gamma}P + I\right) + \frac{(\alpha-\beta)P + I}{\gamma}(\frac{\beta}{\gamma}P + I)](1-\alpha)v \\ &= \left(\frac{\beta}{\gamma}P\right)^{m_k} x^{k,0} + \frac{\alpha(\gamma-1)}{\beta}\Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j x^k \quad + \frac{\alpha(\alpha-\beta)}{\beta}\Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j Px^k \\ &\quad + \frac{(1-\alpha)(\gamma-1)}{\gamma}\Sigma_{j=0}^{m_k-1}\left(\frac{\beta}{\gamma}P\right)^j v + \frac{(1-\alpha)(\alpha-\beta)}{\beta}\Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j v \\ &\quad + \frac{1-\alpha}{\gamma}\Sigma_{j=0}^{m_k-1}(\frac{\beta}{\gamma}P)^j v. \end{aligned} \tag{3.4}$$

Note that $x^{k,0} = \alpha P x^k + (1-\alpha)v$; moreover, we get

$$\begin{aligned} x^{k+1} &= \Big[\tfrac{\alpha\gamma}{\beta}(\tfrac{\beta}{\gamma}P)^{m_k+1} + \tfrac{\alpha(\gamma-1)}{\beta}\Sigma_{j=1}^{m_k}(\tfrac{\beta}{\gamma}P)^j + \tfrac{\alpha(\alpha-\beta)}{\beta}\Sigma_{j=1}^{m_k}(\tfrac{\beta}{\gamma}P)^j P\Big]x^k \\ &\quad + (1-\alpha)\Big[(\tfrac{\beta}{\gamma}P)^{m_k} + \Sigma_{j=0}^{m_k-1}(\tfrac{\beta}{\gamma}P)^j + \tfrac{\alpha-\beta}{\beta}\Sigma_{j=1}^{m_k}(\tfrac{\beta}{\gamma}P)^j\Big]v, \end{aligned}$$

and therefore (3.2) holds.

It follows from (3.2)–(3.3) and the exact solution x^* that

$$x^* = T^k x^* + (1-\alpha)Q^k v, \tag{3.5}$$

where T^k denotes the k-th step iteration matrix. Combined with (3.2) and (3.5), we have

$$x^{k+1} - x^* = T^k(x^k - x^*) = \cdots = T^k T^{k-1} \cdots T^0 (x^0 - x^*). \tag{3.6}$$

Then, one gives

$$\lim_{k\to+\infty} x^k = x^*$$

Table 1 Numerical results with $\alpha = 0.901$, $\beta = 0.9$ and $\gamma = 0.94$

Method		IOI	TSS	RTSS
	It	154	131	101
bfwa	CPU	0.0729	0.0101	0.0066
	RES	4.1885e−014	2.3019e−014	1.0993e−014
	It	164	151	88
rdb	CPU	2.5972	1.8607	1.0456
	RES	1.1032e−014	9.8976e−015	6.5403e−015
	It	38	33	28
cryg	CPU	1.8602	1.7259	1.2429
	RES	5.6366e−014	1.7114e−014	1.3795e−014

Table 2 Numerical results with $\alpha = 0.45$, $\beta = 0.4$ and $\gamma = 0.8$

Method		IOI	TSS	RTSS
	It	26	25	20
bfwa	CPU	0.0318	0.0023	0.0014
	RES	3.9429e−015	3.0842e−015	1.8088e−015
	It	32	28	23
rdb	CPU	0.6156	0.5215	0.4079
	RES	2.9758e−015	1.9350e−015	8.5085e−016
	It	14	12	11
cryg	CPU	0.8050	0.6540	0.4666
	RES	6.8805e−014	5.6382e−014	9.7613e−015

if and only if

$$\lim_{k\to+\infty} T^k T^{k-1} \cdots T^0 = 0.$$

If $\|T^k\|_1 < \eta$ $(0 < \eta < 1)$, then

$$\begin{aligned}\lim_{k\to+\infty} \|T^k T^{k-1} \cdots T^0\|_1 \\ &\le \lim_{k\to+\infty} \|T^k\|_1 \|T^{k-1}\|_1 \cdots \|T^0\|_1 \\ &= \prod_{k=0}^{+\infty} \|T^k\|_1 \le \prod_{k=0}^{+\infty} \eta = 0.\end{aligned} \tag{3.7}$$

Now, we will show the existence of the upper bound parameter η^k.

By (3.3), we get

$$T^k = \frac{\alpha\gamma}{\beta}\left[\left(\frac{\beta}{\gamma}P\right)^{m_k+1} + \frac{\gamma-1}{\gamma}\Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j + \frac{\alpha-\beta}{\gamma}\Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j P\right]$$

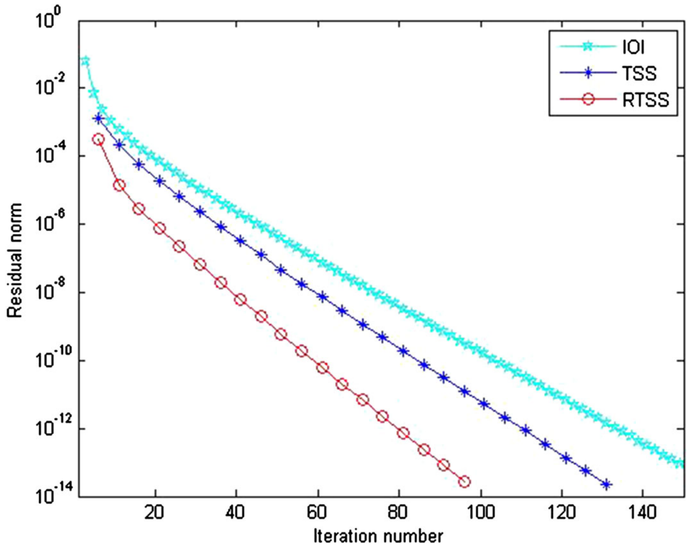

Fig. 1 The bfwa398 matrix with $\alpha = 0.901$, $\beta = 0.9$ and $\gamma = 0.94$

$$\begin{aligned}
&= \frac{\alpha\gamma}{\beta}\left[\left(\frac{\beta}{\gamma}P\right)^{m_k+1} + \Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j\left(\frac{\gamma-1}{\gamma}I + \frac{\alpha-\beta}{\gamma}P\right)\right]\\
&= \frac{\alpha\gamma}{\beta}\left[\left(\frac{\beta}{\gamma}P\right)^{m_k+1} + \Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j\left(\left(I-\frac{\beta}{\gamma}P\right) - \left(\frac{1}{\gamma}I - \frac{\alpha}{\gamma}P\right)\right)\right]\\
&= \frac{\alpha\gamma}{\beta}\left[\left(\frac{\beta}{\gamma}P\right)^{m_k+1} + \frac{\beta}{\gamma}P - \left(\frac{\beta}{\gamma}P\right)^{m_k+1} - \Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j\left(\frac{1}{\gamma}I - \frac{\alpha}{\gamma}P\right)\right]\\
&= \frac{\alpha\gamma}{\beta}\left[\frac{\beta}{\gamma}P - \Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j\left(\frac{1}{\gamma}I - \frac{\alpha}{\gamma}P\right)\right].
\end{aligned}$$

It follows from $e^T P = e^T$ that

$$\begin{aligned}
e^T T^k &= \frac{\alpha\gamma}{\beta}\Big[\frac{\beta}{\gamma}e^T - e^T\Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}P\right)^j(\frac{1}{\gamma}I - \frac{\alpha}{\gamma}P)\Big]\\
&= \frac{\alpha\gamma}{\beta}\Big[\frac{\beta}{\gamma}e^T - \frac{1-\alpha}{\gamma}\Sigma_{j=1}^{m_k}\left(\frac{\beta}{\gamma}\right)^j e^T\Big]\\
&= \frac{\alpha\gamma}{\beta}\Big[\frac{\beta}{\gamma} - \frac{\beta}{\gamma}\frac{(1-\alpha)[1-(\frac{\beta}{\gamma})^{m_k}]}{\gamma-\beta}\Big]e^T\\
&= \frac{\alpha\gamma}{\beta}\Big[\frac{\beta(\gamma-\beta) - \beta(1-\alpha)[1-(\frac{\beta}{\gamma})^{m_k}]}{\gamma(\gamma-\beta)}\Big]e^T\\
&= \eta^k e^T,
\end{aligned} \tag{3.8}$$

where

$$\eta^k := \frac{\alpha\gamma}{\beta}\Big[\frac{\beta(\gamma-\beta) - \beta(1-\alpha)[1-(\frac{\beta}{\gamma})^{m_k}]}{\gamma(\gamma-\beta)}\Big].$$

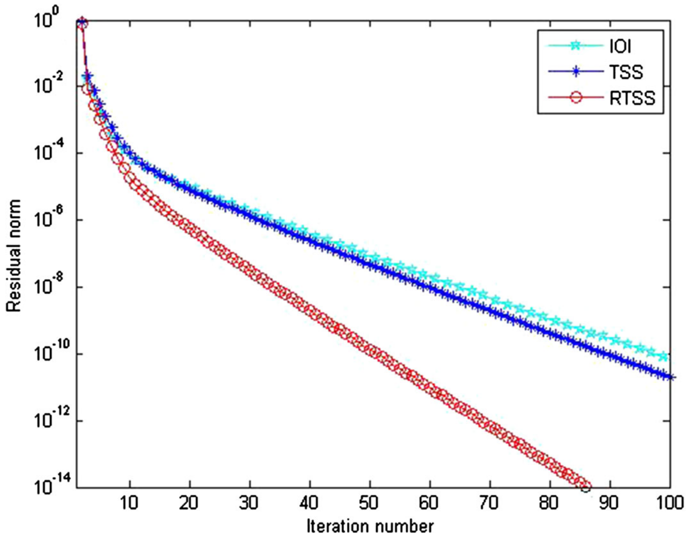

Fig. 2 The rdb5000 matrix with $\alpha = 0.901$, $\beta = 0.9$ and $\gamma = 0.94$

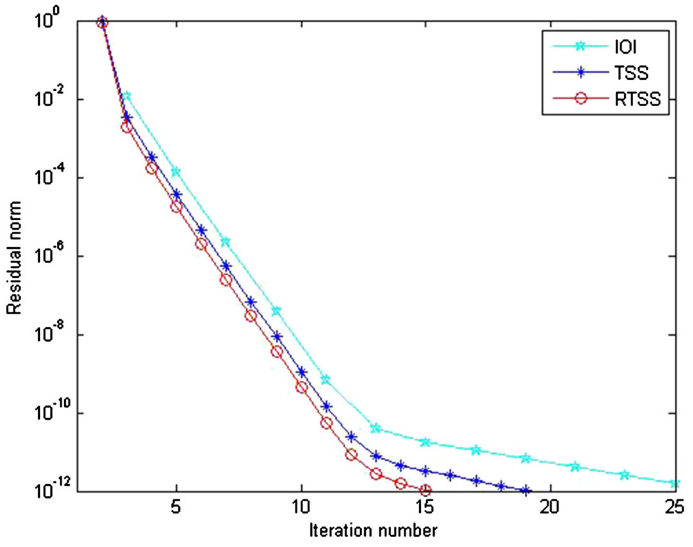

Fig. 3 The cryg10000 matrix with $\alpha = 0.901$, $\beta = 0.9$ and $\gamma = 0.94$

As

$$0 < \beta < \alpha < \gamma \leq 1 \text{ and } 0 < 1 - \alpha < \gamma, \tag{3.9}$$

it holds that

$$0 < \beta < \gamma, \;\; 0 < \frac{\beta(1-\alpha)}{\gamma^2} < 1. \tag{3.10}$$

Furthermore, we have

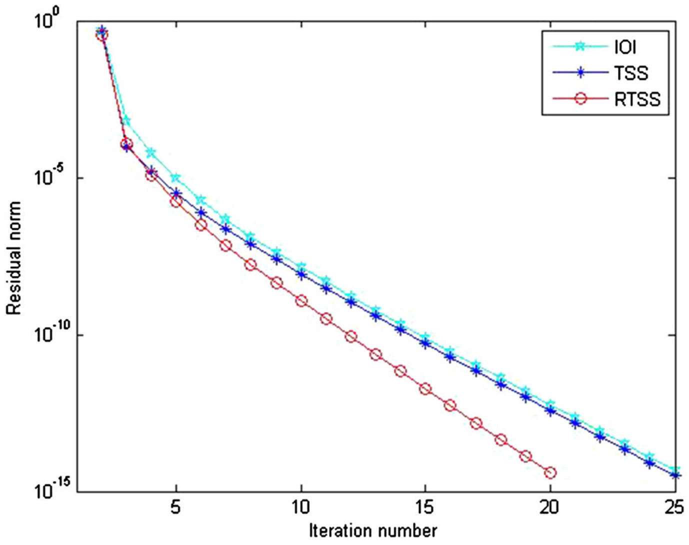

Fig. 4 The bfwa398 matrix with $\alpha = 0.45$, $\beta = 0.4$ and $\gamma = 0.8$

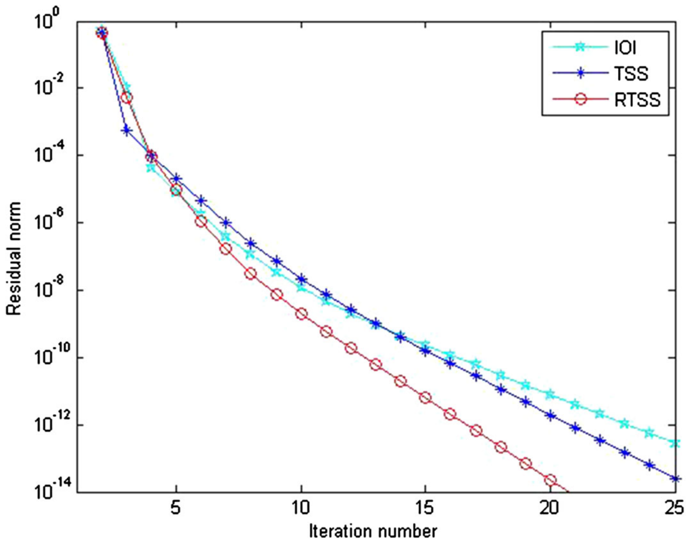

Fig. 5 The rdb5000 matrix with $\alpha = 0.45$, $\beta = 0.4$ and $\gamma = 0.8$

$$\begin{aligned}\frac{\beta(\gamma-\beta)-\beta(1-\alpha)[1-(\frac{\beta}{\gamma})^{m_k}]}{\gamma(\gamma-\beta)} &\le \frac{\beta(\gamma-\beta)-\beta(1-\alpha)[1-\frac{\beta}{\gamma})]}{\gamma(\gamma-\beta)}\\ &= \frac{\beta\gamma(\gamma-\beta)-\beta(1-\alpha)[\gamma-\beta]}{\gamma^2(\gamma-\beta)}\\ &= \frac{\beta}{\gamma}(1-\frac{1-\alpha}{\gamma}) < 1. \qquad (3.11)\end{aligned}$$

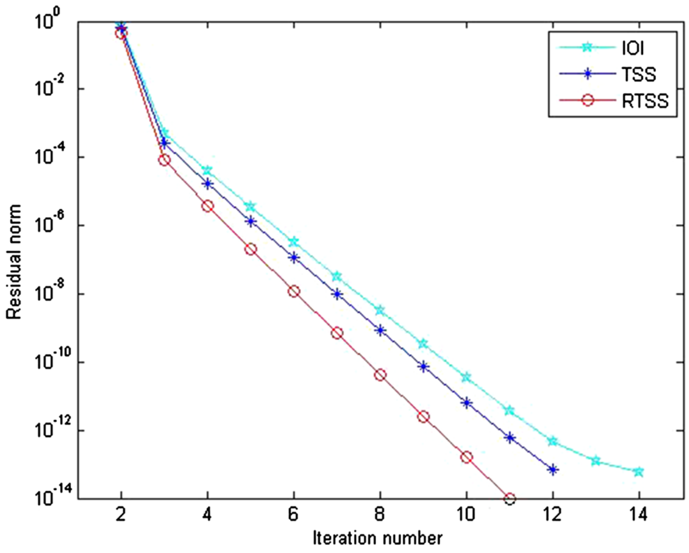

Fig. 6 The cryg10000 matrix with $\alpha = 0.45$, $\beta = 0.4$ and $\gamma = 0.8$

So,

$$\eta^k \leq \alpha(1 - \frac{1-\alpha}{\gamma}) =: \eta < 1.$$

By (3.8)–(3.11), it means that

$$\|T^k\|_1 \leq \eta < 1.$$

Combining with (3.7), we immediately get the conclusion, that is, the iteration solution x^k generated by Algorithm 2.3 converges to the exact solution x^* of the PageRank problem (1.1). This completes the proof. □

4 Numerical experiments

In this section, we report some numerical results to illustrate the effectiveness of the relaxed two-step iteration (RTSS) method for solving the linear system (1.1) arising from the Google PageRank problem. All examples have been carried out by MATLAB R2011b (7.13), Intel(R) Core(TM) i7-2670QM, CPU 2.20GHZ, RAM 8.GB PC Environment. We compare our algorithm to the IOI and TSS methods. We use the following examples to examine these approaches with different iteration performances from these aspects of the iteration numbers(denoted by 'IT'), elapsed CPU time in seconds (denoted by 'CPU') and the residual (denoted as 'RES') defined by

$$\text{RES} := \|\alpha z^k + (1-\alpha)v - x^k\|_1 < 10^{-14}.$$

We compare three approaches by choosing different parameters α, β, and γ in a reasonable interval. Three sparse matrices, bfwa398, rdb5000 and cryg10000, are selected from the University of Florida sparse matrix collection (available at http://www.cise.ufl.edu/research/sparse/mat/). All numerical tests are shown in Tables 1,2 and Figs. 1, 2, 3, 4, 5 and 6. These results illustrate that the convergence of the RTSS approach performs better than the TSS and IOI methods' both CPU time and the numbers of the iteration step, which demonstrates that the introduced approach is efficient.

5 Conclusion

In this paper, we have furthermore improved the two-step splitting iteration by introducing an additional relaxation parameter γ and presented a relaxed two-step splitting iteration for solving the PageRank problem, which can be regarded as a type of generalization for the TSS method. If we choose $\gamma = 1$, the RTSS approach is reduced to the TSS method. It is readily seen that the selection of the proper parameters γ may generate better convergence performances than the TSS and IOI methods. Numerical experiments show the effectiveness of the proposed approach.

References

Bai Z-Z, Golub GH, Lu L-Z, Yin J-F (2004) Block triangular and skew-Hermitian splitting methods for positive-definite linear systems. SIAM J Sci Comput 26(3):844–863
Bai Z-Z (2012) On convergence of the inner-outer iteration method for computing PageRank. Numer Algebra Cont Optim 2(4):855–862
Bai Z-Z, Sun J-C, Wang D-R (1996) A unified framework for the construction of various matrix multisplitting iterative methods for large sparse system of linear equations. Comput Math Appl 32(12):51–76
Berkhin P (2005) A survey on PageRank computing. Internet Math 2(1):73–120
Capizzano SS (2005) Jordan canonical form of the Google matrix: a potential contribution to the PageRank computation. SIAM J Matrix Anal Appl 27(2):305–312
Cottle RW, Dantzig G (1968) Complementary pivot theory of mathematical programming. Linear Algebra Appl 1:103–125
Cottle RW, Pang J-S, Stone RE (1992) The linear complementarity problem. Academic Press, New York
Fan C, Simpson O (2015) Distributed algorithms for finding local clusters using heat kernel pagerank, algorithms and models for the web graph. Springer International Publishing, pp 177–189
Gleich DF, Gray AP, Greif C, Lau T (2010) An inner-outer iteration for computing PageRank. SIAM J Sci Comput 32(1):349–371
Golub GH, Greif C (2006) An Arnoldi-type algorithm for computing PageRank. BIT 46(4):759–771
Golub GH, Van Loan C (1996) Matrix computations, 3rd edn. The Johns Hopkins University Press, Baltimore
Gu C-Q, Xie F, Zhang K (2015) A two-step matrix splitting iteration for computing PageRank. J Comput Appl Math 278(15):19–28
Huang N, Ma C-F (2015) Parallel multisplitting iteration methods based on M-splitting for the PageRank problem. Appl Math Comput 271:337–343
Jia Z-X (1997) Refined iterative algorithms based on Arnoldis process for large unsymmetric eigenproblems. Linear Algebra Appl 259(1):1–23
Kamvar S, Haveliwala T, Manning C, Golub GH (2003) Extrapolation methods for accelerating PageRank computations. In: Twelfth International World Wide Web Conference, Budapest, Hungary, pp 261–270
Kamvar S, Haveliwala T, Golub GH (2004) Adaptive methods for the computation of PageRank. Linear Algebra Appl 386(15):51–65
Kamvar S, Haveliwala T, Manning C, Golub GH (2003) Exploiting the block structure of the web for computing PageRank. Technical Report, Stanford
Langville AN, Meyer CD (2005) A survey of eigenvector methods for Web information retrieval. SIAM Rev 47(1):135–161
Langville A, Meyer C (2006) Googles PageRank and beyond: the science of search engine rankings. Princeton University Press, Princeton
Langville A, Meyer C (2006) A reordering for the PageRank problem. SIAM J Sci Comput 27(6):2112–2120
Mangasarian OL, Meyer RR (2006) Absolute value equations. Linear Algebra Appl 419(2):359–367
Mangasarian OL (2007) Absolute value programming. Comput Optim Appl 36(1):43–53

Morrison JL, Breitling R, Higham DJ, Gilbert DR (2005) GeneRank: using search engine technology for the analysis of microarray experiments. BMC Bioinform 6:233

Moosaei H, Ketabchi S, Noor MA (2015) Some techniques for solving absolute value equations. Appl Math Comput 268:696–705

Page L, Brin S, Motwani R, Winograd T (1999) The PageRank citation ranking: bringing order to the web, Stanford Digital Libraries SIDL-WP-1999-0120, Stanford

Rohn J (2012) An algorithm for computing all solutions of an absolute value equation. Optim Lett 6(5):851–856

Sauer T (2005) Numerical analysis. Addison Wesley Publication, USA

Skardal PS, Taylor D, Sun J, Arenas A (2015) Collective frequency variation in complex networks and Google's PageRank, arXiv preprint arXiv:1510.02018

Stewart W (1994) Introduction to the numerical solution of Markov chains. Princeton University Press, Princeton

Wu G, Wei Y-M (2007) A power-Arnoldi algorithm for computing PageRank. Numer Linear Algebra Appl 14(7):521–546

Yin J-F, Yin G-J, Ng MK (2012) On adaptively accelerated Arnoldi method for computing PageRank. Numer Linear Algebra Appl 19(1):73–85

Yong L-Q (2016) A smoothing newton method for absolute value equation. Int J Control Autom 9(2):119–132

Yong L-Q, Liu S-Y, Zhang S-M (2011) A new method for absolute value equations based on harmony search algorithm. ICIC Express Lett Part B Appl 2:1231–1236