J Appl Math Comput (2013) 42:479–490
DOI 10.1007/s12190-013-0645-5

JAMC

ORIGINAL RESEARCH

On the multi-splitting iteration method for computing PageRank

Chuanqing Gu · Lei Wang

Received: 26 August 2012 / Published online: 8 February 2013

Abstract A two-step iterative scheme based on the multiplicative splitting iteration is presented for PageRank computation. The new algorithm is applied to the linear system formulation of the problem. Our method is essentially a two-parameter iteration which can extend the possibility to optimize the iterative process. Theoretical analyses show that the iterative sequence produced by our method is convergent to the unique solution of the linear system, i.e., PageRank vector. An exact parameter region of convergence for the method is strictly proved. In each iteration, the proposed method requires solving two linear sub-systems with the splitting of the coefficient matrix of the problem. We consider using inner iterations to compute approximate solutions of these linear sub-systems. Numerical examples are presented to illustrate the efficiency of the new algorithm.

Keywords PageRank · Damping factor · Power method · Inner-outer iterations · Splitting iteration methods · Stationary schemes

Mathematics Subject Classification 65F15 · 65F10 · 65C40

1 Introduction

Many problems in scientific computing give rise to a system of linear equations

The work are supported by Shanghai Natural Science Foundation (10ZR1410900), Key Disciplines of Shanghai Municipality (S30104), Innovation Program of Shanghai Municipal Education Commission (13ZZ068), Natural Science Foundation of Universities of Anhui Province (KJ2011A248, KJ2012Z347) and Young Foundation of Huaibei Normal University (700583).

C. Gu (✉) · L. Wang
Department of Mathematics, Shanghai University, Shanghai 200444, China
e-mail: cqgu@staff.shu.edu.cn

L. Wang
e-mail: wanglei625@shu.edu.cn

$$Ax = b, \qquad A \in \mathcal{C}^{n\times n}, \quad \text{and} \quad x, b \in \mathcal{C}^n, \tag{1.1}$$

with A is a large sparse non-Hermitian matrix. Iterative methods play an important role for solving large sparse system of linear equations (1.1) as either solvers or preconditioners, in both theoretical studies and practical applications; for details, see [10, 31] and references therein.

Recently, with the booming development of the Internet, web search engines have become one of the most important Internet tools for information retrieval. One of the best-known algorithms in web search is Google's PageRank algorithm [23]. PageRank is related to a link analysis algorithm for ranking Web pages whereby a page's importance is determined according to the link structure of the Web. The core of the PageRank algorithm involves computing the principle eigenvector of the Google matrix representing the hyperlink structure of the web. The Google matrix is defined as a convex combination of matrices P and a certain rank-1 matrix:

$$A = \alpha P + (1-\alpha)ve^T \tag{1.2}$$

where P is a column-stochastic matrix (i.e., the dangling nodes are already replaced by columns with $1/n$), α $(0 \le \alpha < 1)$ is the damping factor, e is a column vector of all ones, and v is a personalization vector. It is common to set $v = e/n$ in the PageRank model, see [14] for the details.

A key parameter in the model is the damping factor, a scalar that determines the weight given to the Web link graph in the model. Usually, the damping factor α is a real positive number close to 1. The question of the optimal choice of α is an interesting issue, one can see [4] for a recent study. It was argued in [18] that the PageRank vector derived from larger α such as 0.99 perhaps gives a 'truer' PageRank than $\alpha = 0.85$ does. In practice, it may therefore make sense to consider a large range of values [9]. Moreover, PageRank-type algorithms are used in application areas other than web search [22, 28], where the value of α often has a concrete meaning.

Usually, the Google matrix A is so extremely large and sparse that many algorithms for fast eigenvector computation in the field of the numerical linear algebra are unsuitable for this problem, because they require matrix inversions or matrix decompositions. For this reason, there are a variety of PageRank optimization methods which use the web link graph [13, 15, 21, 32]. Meanwhile, iterative methods based on matrix-vector products have been widely studied for this problem.

The original PageRank algorithm uses the power method to compute successive iterates that converge to the principal eigenvector of the Google matrix. However, the eigenvalues of A is the scaling of those of P, except for the dominant eigenvalue [6, 17]. Thus, when the largest eigenvalue of A is not well separated from other eigenvalues if the damping factor α is close to 1, then the power method converges very slowly. Many researchers proposed several methods to accelerate the convergence of the power method, such as the quadratic extrapolation method [14], the adaptive method [16], and the power extrapolation method [12]. And a number of iterative methods based on Krylov subspace were proposed for computing PageRank. For instance, Arnoldi-type method [9], power-Arnoldi method [25], Arnoldi-extrapolation method [26] and adaptively accelerated Arnoldi method [30]. For more theoretical analysis, we refer to [5, 27].

Generally speaking, these techniques can be classified into two groups: one is to reduce the total iterative steps, and the other is to reduce the computational cost per iteration. Unfortunately, these goals usually contradict each other. For a general review of these celebrated approaches, we refer the readers to [18–20].

On the other hand, some researchers focused their attention on the PageRank linear system [1, 7]. Iterative methods based on the linear system are good alternatives for computing the PageRank. We note that the smaller the damping factor is, the easier it is to solve the problem. Gleich et al. [8] proposed an inner-outer iteration method combined with Richardson iterations in which each iteration requires solving a linear system similar in its algebraic structure to the original one. This approach is very efficient as it can solve the PageRank problem with a lower damping factor β.

Recently, the idea of introducing iterative scheme based on splitting of the coefficient matrix into the linear systems has been successfully applied into the solution of linear equation (1.1) [2, 3, 29]. By taking the advantage of the inner-outer iteration algorithm in [8] and the idea of the splitting iterations, we propose a new algorithm for computing the PageRank vector in this paper.

This paper is organized as follows. In Sect. 2, we briefly introduce the PageRank linear system and the inner-outer iteration method for PageRank problem. In Sect. 3, we first study the multiplicative splitting iteration method, its theoretical properties, and then propose a multiplicative splitting iteration method for computing the PageRank. Numerical tests and comparisons are reported in Sect. 4. Finally, some brief concluding remarks are given in Sect. 5.

2 The inner-outer iteration method for computing PageRank

In this section, we briefly review the linear system formulation of PageRank problem and the inner-outer iteration method for computing PageRank.

2.1 PageRank linear system

By the Ergodic Theorem for Markov chains [11], the Markov chain defined by matrix A has a unique stationary probability distribution if A is aperiodic and irreducible. In the case of the PageRank model, the Google matrix A achieves these desirable properties.

Assuming that the probability distribution over the surfer's location at time 0 is given by vector $x^{(0)}$, the probability distribution for the surfer's location at time k is given by $x^{(k)} = A^k x^{(0)}$. The unique stationary distribution of the Markov chain is defined as $\lim_{k\to\infty} x^{(k)}$, which is equivalent to $\lim_{k\to\infty} A^k x^{(0)}$, and is independent of the initial distribution $x^{(0)}$. This is simply the principal eigenvector of Google matrix A, which is exactly the PageRank vector we would like to compute. So the PageRank vector is defined as the vector satisfying $x = Ax$, with A is defined in (1.2).

Notice that A in (1.2) is dense because ve^T is typically dense, but it never needs to be explicitly formed. In particular, the matrix-vector multiplication Ax can be efficiently computed by

$$Ax = \alpha Px + (1-\alpha)v, \tag{2.1}$$

where x is a probability vector, i.e., $\|x\|_1 = 1$. Obviously, if we give an initial vector with a unit 1-norm, then so do all the iterators x throughout the iteration, and the normalization is avoided.

Since $e^T x = 1$, the original PageRank problem can be recast from (2.1) as the linear system:

$$(I - \alpha P)x = (1 - \alpha)v. \tag{2.2}$$

In what follows, we will refer to (2.2) as the PageRank linear system, and more details of the linear system formulation of PageRank problem can be found in [1, 7, 8].

2.2 The inner-outer iteration method for computing PageRank

By taking the fact that the original PageRank problem is easier to solve when the damping factor α is small, Gleich et al. [8] proposed an inner-outer iteration method for computing PageRank by using the linear system formulation of PageRank (2.2) and the stationary inner-outer iterations.

Instead of solving (2.2) directly, they consider the following stationary iteration:

$$(I - \beta P)x_{k+1} = (\alpha - \beta)Px_k + (1 - \alpha)v, \quad k = 0, 1, 2, \dots, \tag{2.3}$$

defined as the outer iteration with a lower damping factor β, i.e., $0 < \beta < \alpha$. Since solving the linear system with $I - \beta P$ is still computationally difficult, they compute x_{k+1} using an inner Richardson iteration as follows. Setting the right-hand items of (2.3) as

$$f = (\alpha - \beta)Px_k + (1 - \alpha)v, \tag{2.4}$$

then they define the inner linear system as

$$(I - \beta P)y = f, \tag{2.5}$$

and apply the inner iteration

$$y_{j+1} = \beta P y_j + f, \quad j = 0, 1, 2, \dots, l - 1, \tag{2.6}$$

where, they take $y_0 = x_k$ as the initial vector and assign the computed solution y_l as the new x_{k+1}.

To terminate the iterations, they apply the 1-norm of the residuals of the inner system (2.5) and the outer system (2.2) respectively as stopping criteria. For the inner iteration (2.6) they use

$$\left\| f - (I - \beta P)y_{j+1} \right\|_1 < \eta, \tag{2.7}$$

and for the outer iteration (2.3) they apply

$$\left\| (1 - \alpha)v - (I - \alpha P)x_{k+1} \right\|_1 < \tau, \tag{2.8}$$

the parameter η and τ are the inner and outer tolerances respectively.

The above algorithm is the core idea of the inner-outer iteration method for PageRank. More details and advantages of the inner-outer iteration method can be found in [8].

3 The multi-splitting iteration method for computing PageRank

In this part, we study the multiplicative splitting iterations and its theoretical properties, and then introduce our new algorithm based on the multiplicative splitting iterations. The motivation and implementation of our new algorithm are studied in details.

3.1 The multi-splitting iteration method for linear systems

The matrix splitting methods play an important role for solving large sparse system of linear equations (1.1). Let $A = M_i - N_i$ $(i = 1, 2)$ be two splittings of the matrix $A \in \mathcal{C}^{n\times n}$. The multiplicative splitting iteration method for solving the system of linear equations (1.1) is defined as follows (see [2]):

Given an initial guess x_0, for $k = 0, 1, 2, \dots$, until $\{x_k\}$ converges, compute

$$\begin{cases} u_{k+1} = M_1^{-1}N_1x_k + M_1^{-1}b, \\ x_{k+1} = M_2^{-1}N_2u_{k+1} + M_2^{-1}b, \end{cases} \quad k = 0, 1, 2, \dots. \tag{3.1}$$

If we introduce matrices

$$H_{msi} = M_2^{-1}N_2M_1^{-1}N_1, \qquad G_{msi} = M_2^{-1}N_2M_1^{-1} + M_2^{-1},$$

then the multiplicative splitting iteration method can be equivalently written in the form:

$$x_{k+1} = H_{msi}x_k + G_{msi}b, \quad k = 0, 1, 2, \dots. \tag{3.2}$$

We note that (3.1) can be generalized to the two-step splitting iteration framework, and the following lemma describes a general convergence criterion for a two-step splitting iteration (see [3]).

Lemma 3.1 *Let $A \in \mathcal{C}^{n\times n}$, $A = M_i - N_i$ $(i = 1, 2)$ be two splittings*[1] *of the matrix A, and let $x_0 \in \mathcal{C}^n$ be a given initial vector. If $\{x_k\}$ is a two-step iteration sequence defined by*

$$\begin{cases} M_1x_{k+\frac{1}{2}} = N_1x_k + b, \\ M_2x_{k+1} = N_2x_{k+\frac{1}{2}} + b, \end{cases} \quad k = 0, 1, 2, \dots,$$

then

$$x_{k+1} = M_2^{-1}N_2M_1^{-1}N_1x_k + M_2^{-1}\big(I + N_2M_1^{-1}\big)b, \quad k = 0, 1, 2, \dots.$$

Moreover, if the spectral radius $\rho(M_2^{-1}N_2M_1^{-1}N_1)$ is less than 1, *then the iteration sequence $\{x_k\}$ converges to the unique solution $x^* \in \mathcal{C}^{n\times n}$ of the system of linear equations* (1.1) *for all initial vectors $x_0 \in \mathcal{C}^n$.*

Evidently, the multiplicative splitting iteration method for the original system of linear equations (1.1) is convergent if the spectral radius of the iteration matrix H_{msi} is less than 1, i.e., $\rho(H_{msi}) < 1$.

[1] Here and in what follows, $A = M - N$ is called a splitting of the matrix A if M is a nonsingular matrix.

3.2 The multi-splitting iteration method for computing PageRank

We now present our new algorithm for computing PageRank by using the PageRank linear system (2.2) and the multi-splitting iterations (3.1).

As is described in Sect. 2.1, the PageRank problem $x = Ax$ can be reformulated as (2.2). Moreover, it is obvious that the smaller the damping factor is, the easier it is to solve the problem. In other words, the same PageRank problem with β as a damping factor, namely

$$(I - \beta P)x = (1 - \beta)v,$$

is easier to solve than the original problem (2.2) if β is smaller than α.

Due to this consideration, we introduce two parameters β_1 and β_2 in the multiplicative splitting iterations for PageRank computation. This leads to the following multiplicative splitting iteration method for computing PageRank (or, simply denoted by MSI method).

Method 3.1 (MSI method) Given an initial guess x_0, for $k = 0, 1, 2, \dots,$ until $\{x_k\}$ converges, compute

$$\begin{cases} (I - \beta_1 P)u_{k+1} = (\alpha - \beta_1)Px_k + (1 - \alpha)v, \\ (I - \beta_2 P)x_{k+1} = (\alpha - \beta_2)Pu_{k+1} + (1 - \alpha)v, \end{cases} \quad k = 0, 1, 2, \dots, \tag{3.3}$$

where α is the given damping factor, two parameters β_1 and β_2 satisfying $0 \le \beta_1 < \alpha$, $0 \le \beta_2 < \alpha$, as for the initial guess we may take $x_0 = e/\|e\|_1$, which is commonly used, see the following section.

The new method MSI is actually a two-parameter two-step iterative method and we will argue that there exists a reasonable convergent domain of two-parameters for MSI method.

Clearly, the multiplicative splitting iteration method for (2.2) is associated with the splitting of the coefficient matrix $I - \alpha P$:

$$\begin{aligned} I - \alpha P &= M_i - N_i \quad (i = 1, 2), \\ M_1 = I - \beta_1 P, &\qquad N_1 = (\alpha - \beta_1)P, \\ M_2 = I - \beta_2 P, &\qquad N_2 = (\alpha - \beta_2)P. \end{aligned}$$

And the corresponding two-step iteration matrix of the multiplicative splitting iteration method (3.3) is given by

$$\widetilde{H}_{msi}(\beta_1, \beta_2) = M_2^{-1}N_2M_1^{-1}N_1 = (I - \beta_2 P)^{-1}(\alpha - \beta_2)P(I - \beta_1 P)^{-1}(\alpha - \beta_1)P.$$

We now study the convergence properties of the multiplicative splitting iteration method for computing PageRank. For the convergence properties of the MSI method, we apply Lemma 3.1 to obtain the following main theorem.

Theorem 3.1 *Let α be the damping factor in the PageRank linear system* (2.2), *and let $M_i = I - \beta_i P$, $N_i = (\alpha - \beta_i)P$ $(i = 1, 2)$ are the two splittings of the matrix $I - \alpha P$. Then the iterative matrix $\widetilde{H}_{msi}(\beta_1, \beta_2)$ of the MSI method for PageRank computation is given by*

$$\widetilde{H}_{msi}(\beta_1,\beta_2)=(I-\beta_2 P)^{-1}(\alpha-\beta_2)P(I-\beta_1 P)^{-1}(\alpha-\beta_1)P, \tag{3.4}$$

and its spectral radius $\rho(\widetilde{H}_{msi}(\beta_1,\beta_2))$ is bounded by

$$\sigma(\beta_1,\beta_2)\equiv\frac{(\alpha-\beta_2)(\alpha-\beta_1)}{(1-\beta_2)(1-\beta_1)},$$

therefore, it holds that

$$\rho\big(\widetilde{H}_{msi}(\beta_1,\beta_2)\big)\le\sigma(\beta_1,\beta_2)<1 \quad \forall\, 0\le\beta_1<\alpha,\ 0\le\beta_2<\alpha;$$

i.e., the multiplicative splitting iteration method for PageRank computation converges to the unique solution $x^*\in\mathcal{C}^n$ of the linear system of equations (2.2).

Proof By putting

$$M_1=I-\beta_1 P,\quad N_1=(\alpha-\beta_1)P,\quad M_2=I-\beta_2 P,\quad \text{and}\quad N_2=(\alpha-\beta_2)P$$

in Lemma 3.1 and noting that $M_1=I-\beta_1 P$ and $M_2=I-\beta_2 P$ are diagonally dominant M-matrix for any positive constant $\beta_i<\alpha$ $(i=1,2)$ [24], we obtain Eq. (3.4).

By observing that if λ_i is an eigenvalue of P, then

$$\mu_i=\frac{(\alpha-\beta_2)(\alpha-\beta_1)\lambda_i^2}{(1-\beta_2\lambda_i)(1-\beta_1\lambda_i)}$$

is an eigenvalue of $\widetilde{H}_{msi}(\beta_1,\beta_2)$. Since $|\lambda_i|\le 1$, we get

$$\begin{aligned}|\mu_i|&=\left|\frac{(\alpha-\beta_2)(\alpha-\beta_1)\lambda_i^2}{(1-\beta_2\lambda_i)(1-\beta_1\lambda_i)}\right|\le\frac{(\alpha-\beta_2)(\alpha-\beta_1)|\lambda_i|^2}{(1-\beta_2|\lambda_i|)(1-\beta_1|\lambda_i|)}\\&=\frac{(\alpha-\beta_2)(\alpha-\beta_1)}{(|\lambda_i|^{-1}-\beta_2)(|\lambda_i|^{-1}-\beta_1)}\le\frac{(\alpha-\beta_2)(\alpha-\beta_1)}{(1-\beta_2)(1-\beta_1)}\end{aligned}$$

with equality holding for $\lambda_1=1$, so $\rho(\widetilde{H}_{msi}(\beta_1,\beta_2))=\frac{(\alpha-\beta_2)(\alpha-\beta_1)}{(1-\beta_2)(1-\beta_1)}<1$. □

Theorem 3.1 shows that the MSI method converges to the unique solution of the linear system (2.2), i.e., PageRank vector, for any given constant β_1 and β_2 satisfying $0\le\beta_1<\alpha$, $0\le\beta_2<\alpha$.

As an immediate consequence of Theorem 3.1, we have the following corollary.

Corollary 3.1 *Let α, β_1, β_2 and $|\mu_i|$ be defined in Theorem* 3.1. *If $\beta_1\beta_2<\alpha$, then $|\mu_i|<\alpha$.*

From Corollary 3.1, it can be concluded that the asymptotic convergence rate of the MSI method is faster than that of the power method with β_1 and β_2 satisfying some moderate conditions.

Next, we will discuss the implementation detail of MSI method for PageRank computation. Notice that each iteration of this method involves solving a two half-step linear system:

$$\begin{cases}(I-\beta_1 P)u_{k+1}=(\alpha-\beta_1)Px_k+(1-\alpha)v,\\(I-\beta_2 P)x_{k+1}=(\alpha-\beta_2)Pu_{k+1}+(1-\alpha)v,\end{cases}\quad k=0,1,2,\ldots, \tag{3.5}$$

when $0 < \beta_1 < \alpha$, $0 < \beta_2 < \alpha$, but each of them is still computationally difficult, even when β_i $(i = 1, 2)$ is small. In this case, the implementation of the half-step of MSI method in each iteration is similar to that of the inner-outer iteration method [8], i.e., instead of solving linear system with $I - \beta_i P$ $(i = 1, 2)$ directly, we define the following inner linear systems as

$$(I - \beta_i P)y = f_i \quad (i = 1, 2), \tag{3.6}$$

where $f_1 = (\alpha - \beta_1)Px_k + (1 - \alpha)v$ or $f_2 = (\alpha - \beta_2)Pu_{k+1} + (1 - \alpha)v$. Then we use the inner iteration (2.6) to get approximate solution u_{k+1} and x_{k+1} in (3.5), respectively. To terminate the inner iterations, we apply

$$\left\| f_i - (I - \beta_i P)y_{j+1} \right\|_1 < \eta \quad (i = 1, 2), \tag{3.7}$$

as stopping criteria for (2.6).

For the original PageRank problem (2.2), the iterations of our algorithm is terminated, when the solution vector x_{k+1} satisfies the stopping criteria:

$$\left\| (1 - \alpha)v - (I - \alpha P)x_{k+1} \right\|_1 < \tau. \tag{3.8}$$

In particular, when at least one of the two parameters β_1 and β_2 in (3.3) equals to 0, we give some remarks about the proposed algorithm.

Remark 3.1 When $\beta_1 = \beta_2 = 0$, the MSI method reduces to the power method [23]. That is, in each iteration of MSI method for PageRank computation, we need to apply the power method twice. Obviously, the convergence rate of the MSI method also reduces to the square of power method's, i.e., α^2.

Remark 3.2 When $\beta_1 = 0$, $\beta_2 \neq 0$, the MSI method reduces to a new special two-step method. That is, in each iteration of MSI method for PageRank computation, we first need to run power method only once in the half iteration step and then solve the remaining linear sub-system by inner iterations, the similar results hold for $\beta_1 \neq 0$, $\beta_2 = 0$.

4 Numerical examples

In this section, we present the numerical experiments to illustrate the performance of the multiplicative splitting iteration method.

In our experiment, we test three methods for computing PageRank, i.e., the power method, the inner-outer iteration (called as 'in-out') method in [8] and the multiplicative splitting iteration (called as 'MSI') method, with damping factor α varying from 0.95 to 0.998. For the inner-outer method, it is observed in [8] that the algorithm is effective for a crude inner tolerance η and is not sensitive to the choice of the parameter β. Hence, our experiments use the choices $\eta = 10^{-2}$ and $\beta = 0.5$. And for a fair comparison, just like in the inner-outer method, the inner tolerance in (3.7) is taken as $\eta = 10^{-2}$.

Because the 1-norm is a natural choice for the PageRank computation [14], in our comparisons we choose the 1-norm of the residual as stopping criterion. For the power method, the stopping criterion is

Table 1 The characteristics of test matrices

No.	Matrix	n	nnz	den
1	Web-Stanford	281903	2312497	0.291×10^{-2}
2	Amazon0505	410236	3356824	0.199×10^{-2}
3	Stanford-Berkeley	683446	7583376	0.162×10^{-2}

$$\|x_{k+1} - x_k\|_1 < 10^{-8},$$

whereas for the inner-outer iteration method and the multiplicative splitting iteration method, the stopping criterion is

$$\left\|(1-\alpha)v - (I - \alpha P)x_{k+1}\right\|_1 < 10^{-8}.$$

The test matrices are obtained from: http://www.cise.ufl.edu/research/sparse/matrices/groups.html. In Table 1, we list the characteristics of the test matrices including matrix size (n), number of nonzeros (nnz) and density (den), which is defined by

$$den = \frac{nnz}{n \times n} \times 100.$$

In our experiments, the largest test matrix is of size 683,446 and has 7,583,376 nonzeros.

Numerical experiments were done on a computer with an Inter Corei3 2.27 GHz with 2G memory. For justification, we set the initial vector $x^{(0)} = e/\|e\|_1$, where $e = (1, 1, \dots, 1)_n$, and n is the dimension of the Google matrix. When we state a speedup, we use relative percentage gain

$$\frac{v_r - v_t}{v_r} \cdot 100\ \%, \tag{4.1}$$

where v_r is a reference performance and v_t is a test performance.

4.1 Choice of the number of β_1 and β_2

According to Theorem 3.1, the MSI method converges for any β_1 and β_2, satisfying $0 \le \beta_1 < \alpha, 0 \le \beta_2 < \alpha$. Table 2 and Table 3 show the number of iterations of the MSI method for two matrices 'Web-Stanford' and 'Stanford-Berkeley' when $\alpha = 0.99$, β_1 grows from 0.1 to 0.9 and β_2 varies from 0.1 to 0.9, respectively.

From Table 2 and Table 3, a *rough* conclusion is that with one of the parameters β_1 and β_2 determined, the iteration steps usually first decrease and then increase with the other parameter increasing; for instance, in Table 3, for $\beta_2 = 0.1$, the iteration steps first decrease for β_1 from 0.1 to 0.7, and then increase for β_1 from 0.8 to 0.9. However, to find an explicit relation between β_1 and β_2, or to find the optimal β_1 and β_2 appear very complicated for general PageRank matrices. Our extensive empirical experience shows that the choice with $\beta_1 = 0.5$ and $\beta_2 = 0.85$ generally yields good performance. That's why we choose $\beta_1 = 0.5$ and $\beta_2 = 0.85$ for MSI method in our experiments.

Table 2 The number of iteration of MSI versus β_1 and β_2 for 'Web-Stanford' matrix

$\beta_2\backslash\beta_1$	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.1	652	617	539	502	487	487	509	497	563
0.3	564	530	507	488	486	486	486	500	537
0.5	487	487	487	486	486	485	484	483	545
0.7	486	485	485	485	484	485	484	493	491
0.9	552	546	523	495	482	527	512	504	549

Table 3 The number of iteration of MSI versus β_1 and β_2 for 'Stanford-Berkeley' matrix

$\beta_2\backslash\beta_1$	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.1	620	570	528	514	513	512	511	514	530
0.3	533	521	515	514	513	512	511	510	516
0.5	514	513	513	512	512	512	511	509	522
0.7	512	512	511	511	511	510	510	509	507
0.9	541	527	524	511	508	508	509	515	523

4.2 Comparison of CPU time and the number of iteration

In Table 4, we list the number of iterations and CPU time for all the test matrices. In the MSI algorithm, according to the analysis of Sect. 4.1, we choose $\beta_1 = 0.5$ and $\beta_2 = 0.85$. The "Gain" in the last column of Table 4 denotes the efficiency of MSI over in-out in terms of CPU time calculated by (4.1). Numerical experiments indicate that our algorithm is better than the inner-outer iteration method.

In all examples, the multiplicative splitting iteration method converges with about half the iterative steps as that of the inner-outer iteration method. Obviously, our method requires less time than the inner-outer method for increasing value of α and n, where α and n denote respectively the damping factor and the size of the Google matrix. For the matrix "Stanford-Berkeley" with $\alpha = 0.998$, for example, the inner-outer iteration method and the multiplicative iteration method require 470.485 s, 426.671 s respectively to reach the desired accuracy; see Table 4. Thus the effectiveness of the proposed algorithm is verified. However, as for the matrix-vector products, they are approximately equal because each iteration of MSI method needs about two matrix-vector multiplications.

Before ending this section, we would like to discuss the complexity of the three methods. The computational cost of the power method per iteration is essentially a matrix-vector product. The in-out method is slightly more expensive than that of the power method per iteration but with much less total iteration steps. Since the MSI method per iteration consists of two in-out processes, then its cost per iteration approximates that of two steps of the in-out method; see, for instance, [8, 14].

Table 4 Comparison of various methods for test matrices

No.	α	Iterations			Time (s)			
		Power	in-out	MSI	Power	in-out	MSI	Gain
1	0.95	280	228	112	19.615	13.545	12.603	6.95 %
	0.99	1392	971	483	104.362	55.576	51.703	6.97 %
	0.995	2783	1763	878	325.609	100.625	93.446	7.13 %
	0.998	6963	3689	1844	576.400	210.096	194.917	7.22 %
2	0.95	238	233	114	23.193	20.059	18.654	7.00 %
	0.99	1043	1032	512	99.134	85.848	79.723	7.13 %
	0.995	1949	1922	957	183.340	159.404	147.926	7.20 %
	0.998	4691	4607	2299	441.627	380.476	352.243	7.42 %
3	0.95	277	225	111	36.143	25.284	23.271	7.96 %
	0.99	1341	1024	508	169.249	111.139	101.467	8.70 %
	0.995	2618	1925	959	343.625	207.843	189.239	8.95 %
	0.998	6350	4362	2177	912.610	470.485	426.671	9.35 %

5 Conclusions

In this paper, we proposed a generalized two-step iteration scheme for the computation of the PageRank vector and demonstrated that the iterative series produced by MSI method converge to the PageRank vector when the parameters β_1 and β_2 satisfy some moderate conditions. Our new algorithm is easy to be implemented. Numerical tests indicate that our new method works better than the inner-outer iteration method. In the future, the theory of the optimal choice for the two parameter β_1 and β_2 is still required to be further analyzed.

Acknowledgements The authors would like to thank the anonymous referees for their valuable comments and suggestion on the original manuscript. These brought several enhancements to our initial work.

References

1. Arasu, A., Novak, J., Tomkins, A., Tomlin, J.: PageRank computation and the structure of the web: experiments and algorithms. In: Proceedings of 11th International Conference on the World Wide Web, Honolulu, 2002. Available online from http://www2002.org/CDROM/poster/173.pdf
2. Bai, Z.Z.: On the convergence of additive and multiplicative splitting iterations for systems of linear equations. J. Comput. Appl. Math. **154**, 195–214 (2003)
3. Bai, Z.Z., Golub, G.H., Ng, M.K.: Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. SIAM J. Matrix Anal. Appl. **24**(3), 603–626 (2003)
4. Boldi, P., Santini, M., Vigna, S.: PageRank as a function of the damping factor. In: Proceedings of the 14th International World Web Conference. ACM, New York (2005)
5. Corso, G.D., Gullí, A., Romani, F.: Comparison of Krylov subspace methods on the PageRank problem. J. Comput. Appl. Math. **210**, 159–166 (2007)
6. Elden, L.: A note on the eigenvalues of the Google matrix. Report LiTH-MAT-R-04-01, Linkoping University (2003)

7. Gleich, D., Zhukov, L., Berkhin, P.: Fast parallel PageRank: a linear system approach. Yahoo! Research Technical Report YRL-2004-038 (2004). Available online from http://research.yahoo.com/publica-tion/YRL-2004-038.pdf
8. Gleich, D., Gray, A., Greif, C., Lau, T.: An inner-outer iteration method for computing PageRank. SIAM J. Sci. Comput. **32**(1), 349–371 (2010)
9. Golub, G.H., Greif, C.: An Arnoldi-type algorithm for computing PageRank. BIT Numer. Math. **46**, 759–771 (2006)
10. Golub, G.H., Van Loan, C.F.: Matrix Computation. Johns Hopkins University Press, Baltimore (1996)
11. Grimmett, G., Stirzakerand, D.: Probability and Random Processes, 3rd edn. Oxford University Press, Oxford (2001)
12. Haveliwala, T.H., et al.: Computing PageRank using power extrapolation. Stanford University Technical Report (2003)
13. Ipsen, I., Selee, T.: PageRank computation, with special attention to dangling nodes. SIAM J. Matrix Anal. Appl. **29**(4), 1281–1296 (2007)
14. Kamvar, S.D., Haveliwala, T.H., Manning, C.D., Golub, G.H.: Extrapolation methods for accelerating PageRank computation. In: Twelfth International World Wide Web Conference. ACM, New York (2003)
15. Kamvar, S.D., Haveliwala, T.H., Manning, C.D., Golub, G.H.: Exploiting the block structure of the web for computing PageRank. Stanford University Technical Report, SCCM-03-02 (2003)
16. Kamvar, S.D., Haveliwala, T.H., Golub, G.H.: Adaptive methods for the computation of the PageRank. Linear Algebra Appl. **386**, 51–65 (2004)
17. Langville, A.N., Meyer, C.D.: Fiddling with PageRank. Technical Report, Department of Mathematics, North Carolina State University (2003)
18. Langville, A.N., Meyer, C.D.: Deeper inside PageRank. Internet Math. **1**(3), 335–380 (2005)
19. Langville, A.N., Meyer, C.D.: A survey of eigenvector methods of web information retrieval. SIAM Rev. **47**(1), 135–161 (2005)
20. Langville, A.N., Meyer, C.D.: A reordering for the PageRank problem. SIAM J. Sci. Comput. **27**(6), 2112–2120 (2006)
21. Lin, Y., Shi, X., Wei, Y.: On computing PageRank via lumping the Google matrix. J. Comput. Appl. Math. **224**, 702–708 (2009)
22. Morrison, J.L., Breitling, R., Higham, D.J., Gilbert, D.R.: GeneRank: using search engine technology for the analysis of microarray experiments. BMC Bioinform. **6**, 233 (2005)
23. Page, L., Brin, S., Motwani, R., Winogrnd, T.: The PageRank citation ranking: bring order to the web. Stanford Digital Libraries Working Paper (1998)
24. Varga, R.: Matrix Iterative Analysis. Prentice Hall, Englewood Cliffs (1962)
25. Wu, G., Wei, Y.: A power-Arnoldi algorithm for computing PageRank. Numer. Linear Algebra Appl. **14**, 521–546 (2007)
26. Wu, G., Wei, Y.: An Arnoldi-extrapolation algorithm for computing PageRank. J. Comput. Appl. Math. **234**, 3196–3212 (2010)
27. Wu, G., Wei, Y.: Arnoldi versus GMRES for computing PageRank: a theoretical contribution to Google's PageRank problem. ACM Trans. Inform. Syst. **28**, 11 (2010)
28. Wu, G., Zhang, Y., Wei, Y.: Krylov subspace algorithms for computing GeneRank for the analysis of microarray data mining. J. Comput. Biol. **17**, 631–646 (2010)
29. Yang, A.L., An, J., Wu, Y.J.: A generalized preconditioned HSS method for non-Hermitian positive definite systems. Appl. Math. **216**, 1715–1722 (2010)
30. Yin, G., Yin, J.: On adaptively accelerated method for computing PageRank. Numer. Linear Algebra Appl. **19**, 73–85 (2012)
31. Young, D.M.: Iterative Solution of Large Linear Systems. Academic Press, New York (1971)
32. Yu, Q., Miao, Z., Wu, G., Wei, Y.: Lumping algorithms for computing Google's PageRank and its derivative, with attention to unreferenced nodes. Inf. Retr. **15**, 503–526 (2012)