Filomat 35:2 (2021), 679–706
https://doi.org/10.2298/FIL2102679T

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

A General Multi-Step Matrix Splitting Iteration Method for Computing PageRank

Zhaolu Tian[a], Xiaojing Li[b], Zhongyun Liu[c]

[a]*College of Applied Mathematics, Shanxi University of Finance and Economics, Taiyuan 030006, P.R.China*
[b]*Department Party Committee, Shanxi University of Finance and Economics, Taiyuan 030006, P.R.China*
[c]*School of Mathematics and Statistics, Changsha University of Science and Technology, Changsha 410076, P.R.China*

Abstract. Based on the general inner-outer (GIO) iteration method [5,34] and the iteration framework [6], we present a general multi-step matrix splitting (GMMS) iteration method for computing PageRank, and analyze its overall convergence property. Moreover, the same idea can be used as a preconditioning technique for accelerating the Krylov subspace methods, such as GMRES method. Finally, several numerical examples are given to illustrate the effectiveness of the proposed algorithm.

1. Introduction

With the booming advance of the Internet and its technology, web search engines have become the most popular tools to retrieve information. The PageRank algorithm, as the key technology of Google, has attracted much attention in the scientific community for last decades.

Firstly, let us review the mathematical background of the PageRank problem, for more details, readers can refer to [1,15]. In fact, the link structure of web pages can be viewed as a direct graph $\mathcal{W}$, each of the n pages is a node. If there is a link from node i to node j, then we have an edge for node i to node j. The nonnegative link matrix $\tilde{P} \in R^{n\times n}$ is defined by

$$\tilde{P}_{ij} = \begin{cases} \dfrac{1}{n_i}, & i \to j, \\ 0, & \text{otherwise}, \end{cases}$$

where the scalar n_i is the number of outlinks of page i, and $i \to j$ denotes page i can link to page j.

Let d be the n-dimensional column vector, which elements satisfy

$$d_i = \begin{cases} 1, & \text{if } n_i = 0, \\ 0, & \text{otherwise}, \end{cases}$$

2010 *Mathematics Subject Classification*. Primary 15A24; Secondary 65F30, 65F35
Keywords. PageRank, Inner-outer iteration, Convergence, Matrix splitting, Precondition
Received: 11 May 2019; Revised: 09 May 2020; Accepted: 18 May 2020
Communicated by Yimin Wei
Research supported by the National Natural Science Foundation of China (Grant No. 11371075), the Hunan Key Laboratory of mathematical modeling and analysis in engineering
Email address: tianzhaolu2004@126.com (Zhaolu Tian)

that identifies the nodes with outdegree 0.

Define an n-dimensional column vector

$$v = \left[\frac{1}{n}\right]_{n\times 1},$$

which represents a uniform probability distribution over all n nodes. Then the matrix $\hat{P}$ can be expressed as follows:

$$\hat{P} = \tilde{P} + dv^T,$$

where v^T is the transpose of v. Hence, the matrix $\hat{P}$ is now a proper row stochastic matrix.

From the ergodic theorem [30], it follows that the stationary distribution of the Markov chain is unique and the limiting distribution starting from any initial distribution when the transition matrix is aperiodic and irreducible. For PageRank problem, a convex combination of $\hat{P}$ with a certain rank-1 matrix can achieve these desirable properties, so we define

$$\bar{P} = \alpha\hat{P} + (1-\alpha)ev^T,$$

where $\alpha \in (0,1)$ is the damping factor, e is a column vector of all ones, and v is called the personalization or the teleportation vector. The PageRank vector x satisfies the following relation:

$$\mathcal{A}x = x, \tag{1.1}$$

where $\mathcal{A} = \bar{P}^T = \alpha P + (1-\alpha)ve^T$ is the Google matrix with $P = \hat{P}^T$, and x is a nonnegative vector with $x \geq 0$. Since $\|x\|_1 = 1$, then the PageRank problem (1.1) can be rewritten as the following linear system [4,8,15]:

$$(I - \alpha P)x = (1-\alpha)v, \tag{1.2}$$

where I is an $n \times n$ identity matrix. The PageRank problem (1.1) often arises in web ranking [2,5,7,10], for example, the hyperlink structure of the web and modeling the graph by the Markov chain, etc.

In the last decades, a large amount of numerical algorithms have been proposed for computing the PageRank vector. However, the fast eigenvector solvers by using matrix inversions or decompositions are unsuitable and prohibitive due to the large and sparse matrix $\mathcal{A}$. Instead, the iteration methods only requiring matrix-vector products have been popular for the PageRank problem. The power method [1,16] is the original method used to compute the PageRank vector. For the case that the largest eigenvalue is not well separated from the second one, the power method may perform poorly, so many extrapolation methods [3,12,19,21] are proposed to accelerate the power method. Since the matrix $I-\alpha P$ is a nonsingular $M-$matrix [20,27,29], based on the regular splittings of the matrix $I - \alpha P$, a class of splitting iteration methods [9] were presented for solving (1.2). The Krylov subspace methods have also been used for solving the PageRank problem. In [10], two strategies to accelerate the Arnoldi-type algorithm were proposed. Moreover, theoretical analysis verified that the new algorithms can improve the efficiency of the original Arnoldi-type algorithm, and circumvent the drawback of stagnation considerably. In [31], an Arnoldi-type algorithm was presented, which is a variant of the restarted Arnoldi method [23]. By combining the power method with the thick restarted Arnoldi algorithm [25], the Power-Arnoldi algorithm was constructed in [26]. A Ritz-value-based Arnoldi-Extrapolation algorithm [18] was developed, which periodically knits extrapolation method with the Arnoldi-type algorithm. By using the thick restarted Arnoldi method to accelerate the inner-outer method [24], an Arnoldi-Inout algorithm was proposed in [22]. Many other algorithms are also established for solving the PageRank problem, such as [32,33,35-41], etc.

Recently, by applying the two-step splitting iteration framework, Gu et al. [11] constructed the PIO iteration method by combining the power method with the IO iteration [24]. Next, Wen et al. [6] gave the MPIO iteration method, which is a variant of the PIO iteration. Inspired by these iteration frameworks in [6,11], based on multi-step matrix splitting iterations and GIO iteration [5,34], we propose the GMMS iteration method for solving the linear system (1.2). In addition, by choosing appropriate matrix splittings and parameters, the GMMS iteration method can reduce to the MPIO and PIO iteration methods, respectively.

The remainder of this paper is organized as follows. In Section 2, the PIO and MPIO iteration methods are reviewed. In the following section, we present the GMMS iteration method and analyze its global convergence in detail. In Section 4, some comparison results are given for the GMMS iteration method. In Section 5, we consider how to use the GMMS iteration as a preconditioner to accelerate the Krylov subspace methods, and discuss the effectiveness of the preconditioner by analyzing the clustering of eigenvalues of the preconditioned coefficient matrices. The choices of the parameters are discussed in the GMMS iteration method in Section 6. Several numerical examples are provided in Section 7 to show the efficiency of the proposed algorithm. Finally, we draw some conclusions in Section 8.

2. The PIO and MPIO iteration methods

Applying the power method to solve the linear system (1.2), then we have the following iteration sequence:

$$x_{k+1} = \alpha P x_k + (1-\alpha)v,\ k = 0, 1, 2, \cdots. \tag{2.1}$$

On the other hand, Gleich et al. [24] proposed an IO iteration method for solving the linear system (1.2). First, (1.2) is reformulated as

$$(I - \beta P)x = (\alpha - \beta)Px + (1-\alpha)v, \tag{2.2}$$

with $0 < \beta < \alpha$, then the outer iteration for solving (2.2) is given by

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

In order to speed up the outer iteration (2.3), an inner Richardson iteration is used to approximate x_{k+1}. Setting the right-hand side of (2.3) as

$$f = (\alpha - \beta)Px_k + (1-\alpha)v,$$

and defining the following inner linear system

$$(I - \beta P)y = f, \tag{2.4}$$

then the inner iteration for solving (2.4) is

$$y_{j+1} = \beta P y_j + f,\ j = 0, 1, 2, \cdots, l-1, \tag{2.5}$$

where y_0 is given by x_k as the initial guess and y_l is assigned as the new x_{k+1}.

Combining (2.1) with the outer iteration (2.3), Gu et al. [11] developed the PIO iteration method as follows:

The PIO iteration method:

$$\begin{cases} x_{k+\frac{1}{2}} = \alpha P x_k + (1-\alpha)v, \\ (I - \beta P)x_{k+1} = (\alpha - \beta)Px_{k+\frac{1}{2}} + (1-\alpha)v, \end{cases} \tag{2.6}$$

Theorem 2.1 [11]. The iteration matrix of the PIO iteration (2.6) is given by

$$\mathcal{R}_{PIO} = \alpha(\alpha - \beta)(I - \beta P)^{-1}P^2,$$

and the modulus of its eigenvalues is bounded by

$$\tilde{s} = \frac{\alpha(\alpha - \beta)}{1 - \beta}$$

with $0 < \alpha < 1$ and $0 < \beta < \alpha$. Then, $\rho(\mathcal{R}_{PIO}) \leq \tilde{s} < 1$ holds, where $\rho(\mathcal{R}_{PIO})$ denotes the spectral radius of the matrix $\mathcal{R}_{PIO}$, which implies that the PIO iteration method converges to the exact solution of the linear system (1.2) for any initial vector x_0.

By using multi-step power method and the IO iteration [24], Wen et al. [6] presented the following MPIO iteration method:

The MPIO iteration method:

$$\begin{cases} x_{k+\frac{1}{\tilde{m}+1}} = \alpha P x_k + (1-\alpha)v, \\ x_{k+\frac{2}{\tilde{m}+1}} = \alpha P x_{k+\frac{1}{\tilde{m}+1}} + (1-\alpha)v, \\ \vdots \\ x_{k+\frac{\tilde{m}}{\tilde{m}+1}} = \alpha P x_{k+\frac{\tilde{m}-1}{\tilde{m}+1}} + (1-\alpha)v, \\ (I-\beta P)x_{k+1} = (\alpha-\beta)P x_{k+\frac{\tilde{m}}{\tilde{m}+1}} + (1-\alpha)v, \end{cases} \tag{2.7}$$

where $0 < \beta < \alpha$ and $0 < \alpha < 1$, $\tilde{m}$ ($\tilde{m} \geq 2$) is the iteration number of using the power method. If $\tilde{m} = 1$, then the MPIO iteration method is just the PIO iteration method (2.6).

Algorithm 1: The MPIO iteration method

Input: $P, \alpha, \beta, v, \tilde{\tau}, \kappa, \tilde{m}$ ($\tilde{m} \geq 2$)
Output: x
1: $x \leftarrow v$
2: $z \leftarrow Px$
3: **while** $\|\alpha z + (1-\alpha)v - x\|_1 \geq \tilde{\tau}$
4: **for** i=1:$\tilde{m}$
5: $x \leftarrow \alpha z + (1-\alpha)v$
6: $z \leftarrow Px$
7: **end**
8: $f \leftarrow (\alpha-\beta)z + (1-\alpha)v$
9: **for** i=1:κ
10: $x \leftarrow f + \beta z$
11: $z \leftarrow Px$
12: **end**
13: **end while**
14: $x \leftarrow \alpha z + (1-\alpha)v$

Theorem 2.2 [6]. The iteration matrix of the MPIO iteration (2.7) is defined by

$$\mathcal{R}_{MPIO} = \alpha^{\tilde{m}}(\alpha-\beta)(I-\beta P)^{-1}P^{\tilde{m}+1},$$

and the modulus of its eigenvalues is bounded by

$$\hat{s} = \frac{\alpha^{\tilde{m}}(\alpha-\beta)}{1-\beta}$$

with $0 < \alpha < 1$ and $0 < \beta < \alpha$. Therefore, the relation $\rho(\mathcal{R}_{MPIO}) \leq \hat{s} < 1$ holds, and the MPIO iteration method converges to the exact solution of the linear system (1.2) for any initial vector x_0.

3. The GMMS iteration method

In this section, we firstly review the GIO iteration method [5,34], then propose the GMMS iteration method, and analyze its global convergence without imposing any restrictions on the damping factors and the stopping tolerances.

Lemma 3.1 [27]. For all operator norms $\rho(W) \leq \|W\|$. For all W and for all $\varepsilon > 0$, there is an operator norm $\|W\|_\star \leq \rho(W) + \varepsilon$. The norm $\|\cdot\|_\star$ depends on both W and ε.

Lemma 3.2 [27]. Let $\|AB\| \leq \|A\| \cdot \|B\|$. Then $\|X\| < 1$ implies that $I - X$ is invertible, $(I-X)^{-1} = \sum_{i=0}^{\infty} X^i$, and $\|(I-X)^{-1}\| \leq \frac{1}{1-\|X\|}$.

3.1. The GIO iteration method [5,34]

Let

$$I - \alpha P = M - N, \tag{3.1}$$

where M is an invertible matrix, then from (3.1) the iteration sequence for solving the linear system (1.2) has the following form:

$$Mx_{k+1} = Nx_k + (1-\alpha)v \tag{3.2}$$

The outer iteration method for solving (1.2) is defined by

$$(M - \psi N)x_{k+1} = (1-\psi)Nx_k + (1-\alpha)v,\ k = 0, 1, 2, \cdots \tag{3.3}$$

with $0 < \psi < 1$.

Let $g = (1-\psi)Nx_k + (1-\alpha)v$, then the inner iteration can be described as

$$My_{j+1} = \psi Ny_j + g,\ j = 0, 1, 2, \cdots, m_k - 1, \tag{3.4}$$

where we take $y_0 = x_k$ as the initial guess and y_{m_k} as the new x_{k+1}. If $M = I, N = \alpha P$ and $\beta = \alpha\psi$, then the GIO iteration method reduces to the IO iteration method [24].

Algorithm 2: The GIO iteration method

Input: $M, N, \alpha, \psi, \epsilon, m_k, v$
Output: x
1: $x \leftarrow v$
2: $w \leftarrow Nx$
3: **while** $\|(1-\alpha)v - Mx + w\|_1 \geq \epsilon$
4: $\quad g \leftarrow (1-\psi)w + (1-\alpha)v$
5: $\quad$ **for** i=1:m_k
6: $\quad\quad Mx \leftarrow \psi w + g$
7: $\quad\quad w \leftarrow Nx$
8: $\quad$ **end**
9: **end while**
10: $Mx \leftarrow w + (1-\alpha)v$

Theorem 3.1 [5]. Let $I - \alpha P = M - N$ be a matrix splitting with $\rho(M^{-1}N) < 1$, $0 < \psi < 1$, and m_k be the number of the inner iteration steps at the $k-$th outer iteration. Then the iteration sequence $\{x_k\}_{k=0}^{\infty}$ derived from the GIO iteration method converges to the exact PageRank vector for any initial vector x_0. Moreover, the GIO iteration method converges faster than the iteration sequence (3.2).

3.2. The GMMS iteration method

By combining the multi-step matrix splitting iterations (3.2) with the GIO iteration in Section 3.1, we obtain the following GMMS iteration method:

The GMMS iteration method:

$$\begin{cases} Mx_{k+\frac{1}{m+1}} = Nx_k + (1-\alpha)v, \\ Mx_{k+\frac{2}{m+1}} = Nx_{k+\frac{1}{m+1}} + (1-\alpha)v, \\ \vdots \\ Mx_{k+\frac{m}{m+1}} = Nx_{k+\frac{m-1}{m+1}} + (1-\alpha)v, \\ (M - \psi N)x_{k+1} = (1-\psi)Nx_{k+\frac{m}{m+1}} + (1-\alpha)v, \end{cases} \tag{3.5}$$

where the GIO iteration is used for the last iteration in (3.5). If $M = I, N = \alpha P$ and $\beta = \alpha\psi$, then the GMMS iteration becomes the MPIO iteration (2.7). Let $m = 1$, then we obtain the following general two-step matrix splitting (GTMS) iteration method:

The GTMS iteration method:

$$\begin{cases} Mx_{k+\frac{1}{2}} = Nx_k + (1-\alpha)v, \\ (M-\psi N)x_{k+1} = (1-\psi)Nx_{k+\frac{1}{2}} + (1-\alpha)v, \end{cases} \tag{3.6}$$

Let $M = I$, $N = \alpha P$ and $\beta = \alpha\psi$, then the GTMS iteration is just the PIO iteration (2.6).

Algorithm 3: The GMMS iteration method

Input: M, N, α, ψ, v, ζ, m_k, m ($m \geq 2$)
Output: x
1: $x \leftarrow v$
2: $w \leftarrow Nx$
3: **while** $\|w + (1-\alpha)v - Mx\|_1 \geq \zeta$
4: **for** i=1:m
5: $Mx \leftarrow w + (1-\alpha)v$
6: $w \leftarrow Nx$
7: **end**
8: $g \leftarrow (1-\psi)w + (1-\alpha)v$
9: **for** i=1:m_k
10: $Mx \leftarrow \psi w + g$
11: $w \leftarrow Nx$
12: **end**
13: **end while**
14: $Mx \leftarrow w + (1-\alpha)v$

Next, we will analyze the global convergence of the GMMS iteration method. In fact, the GMMS iteration method can be written as the following two-stage matrix splitting iteration framework [17,28]:

$$\begin{cases} x_{k,0} = x_k,\ x_{k+1} = x_{k,m_k}, \\ Mx_{k,j+1} = \psi N x_{k,j} + (1-\psi)N(M^{-1}N)^m x_k + (1-\alpha)\left((1-\psi)N\sum_{i=0}^{m-1}(M^{-1}N)^i M^{-1} + I\right)v, \\ k = 0,1,2,\cdots,\ j = 0,1,2,\cdots,m_k - 1. \end{cases} \tag{3.7}$$

Theorem 3.2. Let $I - \alpha P = M - N$ be a matrix splitting with $\rho(M^{-1}N) < 1$, $0 < \psi < 1$, and m_k be the number of the inner iteration steps at the $k-$th outer iteration with $m_k \geq 1$. Then the iteration sequence $\{x_k\}_{k=0}^{\infty}$ generated by (3.7) converges to the PageRank vector for any initial vector x_0.

Proof. From (3.7), we have

$$\begin{aligned} x_{k,j+1} &= \left((\psi R)^{j+1} + (1-\psi)\sum_{t=0}^{j}(\psi R)^t R^{m+1}\right)x_k \\ &\quad + (1-\alpha)\sum_{t=0}^{j}(\psi R)^t\left((1-\psi)N\sum_{i=0}^{m-1}(M^{-1}N)^i M^{-1} + I\right)v \end{aligned}$$

with $R = M^{-1}N$. For $j = m_k - 1$, then it follows that

$$x_{k+1} = H_k x_k + F_k v,\ \ k = 0,1,2,\cdots, \tag{3.8}$$

where

$$\begin{cases} H_k = (\psi R)^{m_k} + (1-\psi)\sum_{t=0}^{m_k-1}(\psi R)^t R^{m+1}, \\ F_k = (1-\alpha)\sum_{t=0}^{m_k-1}(\psi R)^t\left((1-\psi)N\sum_{i=0}^{m-1}(M^{-1}N)^i M^{-1} + I\right)v,\ k = 0,1,2,\cdots. \end{cases} \tag{3.9}$$

Let x^* be the exact solution to the linear system (1.2), then from (3.7), (3.8) we obtain

$$x^* = H_k x^* + F_k v,\ k = 0, 1, 2, \cdots . \tag{3.10}$$

Subtracting (3.10) from (3.8), then

$$x_{k+1} - x^* = H_k(x_k - x^*) = \cdots = H_k H_{k-1} \cdots H_0(x_0 - x^*),\ \ k = 0, 1, 2, \cdots .$$

Let $\nu_i^{(k)}$ be an eigenvalue of H_k and λ_i be an eigenvalue of R, respectively. From (3.9), we have

$$\begin{aligned} \nu_i^{(k)} &= (\psi\lambda_i)^{m_k} + (1-\psi)\lambda_i^{m+1} \sum_{t=0}^{m_k-1} (\psi\lambda_i)^t \\ &= (\psi\lambda_i)^{m_k} + (1-\psi)\lambda_i^{m+1} \tfrac{1-(\psi\lambda_i)^{m_k}}{1-\psi\lambda_i} \\ &= \tfrac{(\psi\lambda_i)^{m_k}(1-\psi\lambda_i)+(1-\psi)\lambda_i^{m+1}(1-(\psi\lambda_i)^{m_k})}{1-\psi\lambda_i}. \end{aligned}$$

Since $0 < \psi < 1$ and $|\lambda_i| < 1$, then

$$\begin{aligned} |\nu_i^{(k)}| &= \left| \tfrac{(\psi\lambda_i)^{m_k}(1-\psi\lambda_i)+(1-\psi)\lambda_i^{m+1}(1-(\psi\lambda_i)^{m_k})}{1-\psi\lambda_i} \right| \\ &\le \left| \tfrac{(\psi\rho(R))^{m_k}(1+\psi\rho(R))+(1-\psi)\rho(R)^{m+1}(1+(\psi\rho(R))^{m_k})}{1-\psi\rho(R)} \right| \\ &< \left| \tfrac{(\psi\rho(R))^{m_k}(1+\psi\rho(R))+(1-\psi)\rho(R)^{m+1}(1+(\psi\rho(R))^{m_k})}{1-\psi} \right| \\ &= \rho(R)^{m+1} < 1 \end{aligned}$$

as $m_k \to \infty$, so $\rho(H_k) < 1$.

Let $\sigma = \max\limits_k\{\rho(H_k)\} < 1$ $(k = 0, 1, 2, \cdots)$ and ξ_i be an eigenvalue of $H_k H_{k-1} \cdots H_0$, then we have

$$\xi_i = \Pi_{s=0}^{k} \frac{(\psi\lambda_i)^{m_s}(1-\psi\lambda_i) + (1-\psi)\lambda_i^{m+1}(1-(\psi\lambda_i)^{m_s})}{1-\psi\lambda_i},$$

so

$$\rho(H_k H_{k-1} \cdots H_0) \le \rho(H_k)\rho(H_{k-1}) \cdots \rho(H_0) \le \sigma^{k+1} < \phi^{k+1}$$

with $0 < \sigma < \phi < 1$.

From Lemma 3.1, there exists an operator norm $\|\cdot\|_\varrho$ such that

$$\|H_k H_{k-1} \cdots H_0\|_\varrho < \phi^{k+1}.$$

Then

$$\|x_{k+1} - x^*\|_\varrho \le \|H_k H_{k-1} \cdots H_0\|_\varrho \|x_0 - x^*\|_\varrho < \phi^{k+1}\|x_0 - x^*\|_\varrho \to 0 \tag{3.11}$$

as $k \to \infty$. Therefore, the iteration sequence $\{x_k\}_{k=0}^{\infty}$ converges to the exact PageRank vector x^* according to (3.11), and the proof is completed. □

Let $P = D + L + U$, where D is the diagonal part of P, and L, U are the strictly lower and upper triangular parts of P, respectively. Then the matrix splitting of the AOR iteration method [14] for solving the linear system (1.2) is

$$M_A = \frac{1}{\omega}(I - \alpha D - \gamma\alpha L),\ N_A = \frac{1}{\omega}((1-\omega)(I - \alpha D) + (\omega - \gamma)\alpha L + \omega\alpha U), \tag{3.12}$$

where $\omega,\ \gamma$ are two real parameters with $\omega \neq 0$.

For different ω and γ, we can obtain the corresponding iteration methods from (3.12):

(1) Jacobi method: $\omega = 1,\ \gamma = 0$.

(2) Gauss-Seidel method: $\omega = \gamma = 1$.

(3) SOR method: $\omega = \gamma$.

Theorem 3.3. Let $G_A = M_A^{-1}N_A$ and $J = (I-\alpha D)^{-1}(\alpha L+\alpha U)$ be the AOR iteration matrix and Jacobi iteration matrix for solving the linear system (1.2), respectively. If $0 < \omega < \frac{2}{1+\rho(J)}$ and $0 \le \gamma \le \omega$, then

$$\rho(G_A) \le |1-\omega| + \omega\rho(J) < 1.$$

Proof. This is a special case of Theorem 3.3 [13]. □

Remark 1. For different ω and γ in the AOR splitting (3.12), we can construct the GMMS iteration method based on the corresponding splittings, such as the Jacobi and Gauss-Seidel splittings, etc.

4. Some comparison results for the GMMS iteration method

Definition 4.1 [20]. For a matrix $A \in R^{n\times n}$, $A = M - N$ is a regular splitting if M is nonsingular with $M^{-1} \ge 0$ and $N \ge 0$.

Theorem 4.1 [20]. Let $A = M - N$ be a regular splitting of A. If $A^{-1} \ge 0$, then

$$\rho(M^{-1}N) = \frac{\rho(A^{-1}N)}{1+\rho(A^{-1}N)} < 1.$$

Theorem 4.2 ([20]). Let $A = M_1 - N_1 = M_2 - N_2$ be two regular splittings of A, where $A^{-1} \ge 0$. If $N_2 \ge N_1 \ge 0$, then

$$0 \le \rho(M_1^{-1}N_1) \le \rho(M_2^{-1}N_2) < 1.$$

Theorem 4.3. Let $I-\alpha P = M_i - N_i$ $(i = 1, 2)$ be two regular splittings and $0 < \psi < 1$. If $\rho(M_1^{-1}N_1) < \rho(M_2^{-1}N_2)$, then the GMMS iteration method derived from (M_1, N_1) converges faster than that based on (M_2, N_2) for any initial vector x_0.

Proof. From (3.5), the iteration sequence of the GMMS iteration method is

$$\begin{aligned} x_{k+1} &= (1-\psi)(M-\psi N)^{-1}N(M^{-1}N)^m x_k \\ &\quad +(1-\alpha)(M-\psi N)^{-1}\left((1-\psi)N\textstyle\sum_{i=0}^{m-1}(M^{-1}N)^i M^{-1}+I\right)v, \end{aligned} \tag{4.1}$$

which iteration matrix is

$$\begin{aligned} \mathcal{R}_{GMMS} &= (1-\psi)(M-\psi N)^{-1}N(M^{-1}N)^m \\ &= (1-\psi)(I-\psi R)^{-1}R^{m+1}. \end{aligned} \tag{4.2}$$

Let λ_i be an eigenvalue of R, then

$$\theta_i = \frac{(1-\psi)\lambda_i^{m+1}}{1-\psi\lambda_i} \tag{4.3}$$

is an eigenvalue of $\mathcal{R}_{GMMS}$.

From (4.3), it follows that

$$|\theta_i| = \left|\frac{(1-\psi)\lambda_i^{m+1}}{1-\psi\lambda_i}\right| \le \frac{(1-\psi)|\lambda_i|^{m+1}}{1-\psi|\lambda_i|} \le \frac{(1-\psi)\rho(R)^{m+1}}{1-\psi\rho(R)}$$

with $0 < \psi < 1$ and $\rho(R) < 1$. Thus,

$$\rho(\mathcal{R}_{GMMS}) \le \frac{(1-\psi)\rho(R)^{m+1}}{1-\psi\rho(R)}. \tag{4.4}$$

Since $I - \alpha P = M_i - N_i (i = 1, 2)$ is a regular splitting, by [29] and Theorem 4.1, it is clear that the matrix $R_i = M_i^{-1}N_i$ is a nonnegative matrix and $\rho(R_i)$ is an eigenvalue of R_i with $\rho(R_i) < 1$.

Let $G(R_i)$ be the iteration matrix of the GMMS iteration method based on (M_i, N_i). From (4.4), it is clear that

$$\rho(G(R_i)) = \frac{(1-\psi)\rho(R_i)^{m+1}}{1-\psi\rho(R_i)},\ (i = 1, 2). \tag{4.5}$$

Let $f(\eta) = \frac{(1-\psi)\eta^{m+1}}{1-\psi\eta}$ $(0 < \eta < 1)$, by some simple calculations, we obtain

$$f'(\eta) = \frac{(1-\psi)\eta^m((m+1)(1-\psi\eta)+\psi\eta)}{(1-\psi\eta)^2} > 0 \tag{4.6}$$

with $0 < \psi < 1$, which means $f(\eta)$ is monotonically increasing. From the assumption $\rho(R_1) < \rho(R_2)$ and (4.6), it follows that

$$\rho(G(R_1)) < \rho(G(R_2)),$$

and the proof is completed. □

Theorem 4.4. Let $I-\alpha P = M-N$ be a matrix splitting and $\rho(M^{-1}N) < \rho(\alpha P)$ or $\rho(M^{-1}N) < \alpha$, then the GMMS iteration method obtained from (M, N) is faster than the MPIO iteration method for any initial vector x_0.

Proof. From Definition 4.1, it is clear that the matrix splitting

$$I - \alpha P = \hat{M} - \hat{N}$$

with $\hat{M} = I$ and $\hat{N} = \alpha P$ is a regular splitting and $\hat{M}^{-1}\hat{N} = \alpha P$. From (4.5), we have

$$\rho(\mathcal{R}_{MPIO}) = \frac{(1-\psi)\rho(\alpha P)^{m+1}}{1-\psi\rho(\alpha P)}. \tag{4.7}$$

For the matrix splitting $I - \alpha P = M - N$, it follows from (4.4) that

$$\rho(\mathcal{R}_{GMMS}) \leq \frac{(1-\psi)\rho(R)^{m+1}}{1-\psi\rho(R)} \tag{4.8}$$

with $R = M^{-1}N$.

In fact, the assumption $\rho(M^{-1}N) < \rho(\alpha P)$ is equivalent to $\rho(M^{-1}N) < \alpha$. Since P is a column-stochastic matrix, then

$$e^T(\alpha P) = \alpha e^T, \tag{4.9}$$

where e is a column vector of all ones, so α is an eigenvalue of αP with the corresponding eigenvector e^T. By Lemma 3.1 and (4.9), then we obtain

$$\alpha \leq \rho(\alpha P) \leq \|\alpha P\|_1 = \alpha$$

with $\|P\|_1 = 1$. Therefore, $\rho(\alpha P) = \alpha$.

By (4.6), (4.7), (4.8) and the assumption $\rho(R) < \rho(\alpha P)$, then we have

$$\frac{(1-\psi)\rho(R)^{m+1}}{1-\psi\rho(R)} < \frac{(1-\psi)\rho(\alpha P)^{m+1}}{1-\psi\rho(\alpha P)}.$$

Thus

$$\rho(\mathcal{R}_{MMSIO}) < \rho(\mathcal{R}_{MPIO})$$

and the proof is completed. □

Theorem 4.5. Let $J = (I - \alpha D)^{-1}(\alpha L + \alpha U)$ be the Jacobi iteration matrix and $0 \leq \gamma \leq \omega$. If $\frac{1-\alpha}{1-\rho(J)} < \omega < \frac{1+\alpha}{1+\rho(J)}$ and $\omega \neq 1$, then the GMMS iteration method based on (3.12) converges faster than the MPIO iteration method.

Proof. For $\omega = 1$ and $\gamma = 0$, the Jacobi splitting from (3.12) is

$$M_J = I - \alpha D, N_J = \alpha L + \alpha U.$$

Since

$$N_J = \alpha(L + U) \leq \hat{N} = \alpha P,$$

by Theorem 4.2, then we have

$$\rho(J) \leq \rho(\alpha P) \Leftrightarrow \rho(J) \leq \alpha. \tag{4.10}$$

Case 1: $\frac{1-\alpha}{1-\rho(J)} < \omega < 1$. It follows from (4.10) and Theorem 3.3 that

$$\begin{aligned} \rho(G_A) &\leq |1-\omega| + \omega\rho(J) \\ &= 1 - \omega + \omega\rho(J) \\ &= 1 - \omega(1-\rho(J)) \\ &< 1 - \tfrac{1-\alpha}{1-\rho(J)}(1-\rho(J)) \\ &= \alpha. \end{aligned} \tag{4.11}$$

Case 2: $1 < \omega < \frac{1+\alpha}{1+\rho(J)}$. From (4.10) and Theorem 3.3, then we obtain

$$\begin{aligned} \rho(G_A) &\leq |1-\omega| + \omega\rho(J) \\ &= \omega - 1 + \omega\rho(J) \\ &= \omega(1+\rho(J)) - 1 \\ &< \tfrac{1+\alpha}{1+\rho(J)}(1+\rho(J)) - 1 \\ &= \alpha. \end{aligned} \tag{4.12}$$

From the assumptions, (4.11) and (4.12), we have $\rho(G_A) < \alpha$. Thus, the proof is completed by Theorem 4.4. □

Corollary 4.1. Let $I - \alpha P = M - N$ be a matrix splitting. If $\rho(M^{-1}N) < \rho(\alpha P)$ or $\rho(M^{-1}N) < \alpha$, then the GTMS iteration method based on (M, N) converges faster than the PIO iteration method for any initial vector x_0.

Proof. The proof can be easily completed by referring to Theorem 4.4. □

In fact, the above results are derived from the assumption that the inner iteration in the GIO iteration method is a precise iteration. In practice, the inner iteration is an inaccurate iteration, so we need to compare the overall convergence rate of the GMMS iteration method.

Theorem 4.6. Let $I - \alpha P = M_i - N_i$ $(i = 1, 2)$ be two regular splittings, $0 < \psi < 1$, and m_k be the number of the inner iteration steps at the $k-$th outer iteration. If $\rho(M_1^{-1}N_1) < \rho(M_2^{-1}N_2)$, then the GMMS iteration method with the matrix splitting (M_1, N_1) performs better than that with the matrix splitting (M_2, N_2) for any initial vector x_0.

Proof. Let $m_k = \tau(k = 0, 1, 2, \cdots)$, then from (3.9) it follows that the iteration matrix of the GMMS iteration method is

$$H_{GMMS} = (\psi R)^\tau + (1-\psi)\sum_{t=0}^{\tau-1}(\psi R)^t R^{m+1}.$$

Let δ_i be an eigenvalue of the matrix H_{GMMS} and λ_i be an eigenvalue of the matrix R, respectively. Then,

$$\delta_i = (\psi\lambda_i)^\tau + (1-\psi)\sum_{t=0}^{\tau-1}(\psi\lambda_i)^t\lambda_i^{m+1} \tag{4.13}$$

From (4.13), we have

$$\rho(H_{GMMS}) = \max_i |\delta_i| \leq (\psi\rho(R))^\tau + (1-\psi)\sum_{t=0}^{\tau-1}(\psi\rho(R))^t(\rho(R))^{m+1}. \tag{4.14}$$

Let $H(R_i)$ be the iteration matrix of the GMMS iteration method based on the regular splitting (M_i, N_i) with $R_i = M_i^{-1}N_i$. Then it follows from (4.14) that

$$\rho(H(R_i)) = (\psi\rho(R_i))^\tau + (1-\psi)\sum_{t=0}^{\tau-1}(\psi\rho(R_i))^t(\rho(R_i))^{m+1}.$$

Define the function

$$\tilde{f}(\tilde{\eta}) = (\psi\tilde{\eta})^{\tau} + (1-\psi)\sum_{t=0}^{\tau-1}(\psi\tilde{\eta})^{t}(\tilde{\eta})^{m+1}$$

with $0 < \tilde{\eta} < 1$. Then we have

$$\tilde{f}'(\tilde{\eta}) = \tau\psi(\psi\tilde{\eta})^{\tau-1} + (1-\psi)\sum_{t=1}^{\tau-1}(t+m+1)\psi^{t}\tilde{\eta}^{t+m} > 0 \tag{4.15}$$

with $0 < \psi < 1$, which implies that $\tilde{f}(\tilde{\eta})$ is monotonically increasing.

From the assumption $\rho(M_1^{-1}N_1) < \rho(M_2^{-1}N_2)$ and (4.15), thus

$$\rho(H(R_1)) < \rho(H(R_2))$$

and the proof is completed. □

Theorem 4.7. Let $I - \alpha P = M - N$ be a regular splitting, $0 < \psi < 1$, and $\tilde{m}_k, m_k$ be the numbers of the inner iteration steps at the $k-$th outer iteration in the GIO and GMMS iteration methods, respectively. If $m \geq 1$ and $\tilde{m}_k = m_k = \tau$, then the GMMS iteration method converges faster than the GIO iteration method for any initial vector x_0.

Proof. From Theorem 3.1 [5], the iteration matrix of the GIO iteration method is

$$H_{GIO} = (\psi R)^{\tau} + (1-\psi)\sum_{s=0}^{\tau-1}(\psi R)^{s}R.$$

According to the proof of Theorem 4.6, then

$$\rho(H_{GIO}) = (\psi\rho(R))^{\tau} + (1-\psi)\sum_{s=0}^{\tau-1}(\psi\rho(R))^{s}\rho(R)$$

and

$$\rho(H_{GMMS}) = (\psi\rho(R))^{\tau} + (1-\psi)\sum_{t=0}^{\tau-1}(\psi\rho(R))^{t}(\rho(R))^{m+1}.$$

Since $0 < \psi < 1$, $m \geq 1$ and $\rho(R) < 1$, then

$$\rho(H_{GMMS}) - \rho(H_{GIO}) = (1-\psi)\rho(R)\sum_{t=0}^{\tau-1}(\psi\rho(R))^{t}\left((\rho(R))^{m} - 1\right) < 0.$$

Therefore, $\rho(H_{GMMS}) < \rho(H_{GIO})$ and the proof is completed. □

Remark 2. From Theorem 2.11 [5], it shows that the AOR splitting (3.12) is a regular splitting, then the above comparison results are meaningful. Furthermore, by Theorem 4.6, we can obtain the similar conclusions to Theorems 4.3, 4.4, 4.5 and Corollary 4.1.

5. Preconditioning for Krylov subspace methods

From (4.1), the matrix splitting of the GMMS iteration method is

$$I - \alpha P = \tilde{M} - \tilde{N}, \tag{5.1}$$

where

$$\begin{cases} \tilde{M} = \left((1-\psi)N \sum_{i=0}^{m-1} (M^{-1}N)^i M^{-1} + I \right)^{-1} (M - \psi N), \\ \tilde{N} = (1-\psi) \left((1-\psi)N \sum_{i=0}^{m-1} (M^{-1}N)^i M^{-1} + I \right)^{-1} N(M^{-1}N)^m. \end{cases}$$

Then we can solve the following preprocessed linear system

$$\tilde{M}^{-1}(I - \alpha P)x = \tilde{M}^{-1}(1-\alpha)v \tag{5.2}$$

by some Krylov subspace methods, such as GMRES method. In Section 3, we have verified that the GMMS iteration method is unconditionally convergent to the PageRank vector, which means that the spectrum of the preconditioned matrix $\tilde{M}^{-1}(I - \alpha P)$ lies entirely in a circle centered at (1, 0) with radius unity, this is a desirable property for Krylov subspace acceleration.

Since it is difficult to calculate $(M - \psi N)^{-1}$ in $\tilde{M}^{-1}$, so we consider a Neumann series approximation of $(M - \psi N)^{-1}$. Assume that $\rho(R) = \rho(M^{-1}N) < 1$, then by Lemmas 3.1 and 3.2, we obtain

$$(M - \psi N)^{-1} = (I - \psi R)^{-1} M^{-1} = \left(\sum_{i=0}^{\infty} (\psi R)^i \right) M^{-1}. \tag{5.3}$$

Here we adopt $s + 1$ terms of (5.3) as an approximation of $(M - \psi N)^{-1}$:

$$(M - \psi N)^{-1} \approx \left(I + \psi R + (\psi R)^2 + \cdots + (\psi R)^s \right) M^{-1}. \tag{5.4}$$

Thus, an approximated preconditioner $\bar{M}$ for $\tilde{M}$ can be constructed from (5.4) as follows:

$$\bar{M} = \left(\left(I + \psi R + (\psi R)^2 + \cdots + (\psi R)^s \right) \left((1-\psi)R \sum_{i=0}^{m-1} (M^{-1}N)^i M^{-1} + I \right) \right)^{-1}$$

In order to illustrate the efficiency of the preconditioner $\tilde{M}$ and $\bar{M}$ on the Krylov subspace methods, the clustering of eigenvalues of the matrices $\tilde{M}^{-1}(I - \alpha P)$ and $\bar{M}^{-1}(I - \alpha P)$ will be analyzed, respectively.

For the preconditioner $\tilde{M}$, it follows that

$$\begin{aligned} \tilde{M}^{-1}(I - \alpha P) &= (M - \psi N)^{-1} \left((1-\psi)N \sum_{i=0}^{m-1} (M^{-1}N)^i M^{-1} + I \right) (M - N) \\ &= (I - \psi R)^{-1} \left((1-\psi) \sum_{i=0}^{m-1} R^{i+1} + I \right) (I - R) \end{aligned}$$

with $I - \alpha P = M - N$. Let λ_i be an eigenvalue of R, then

$$\tilde{\lambda}_i = \frac{(1-\psi)\left(1 - \lambda_i^{m+1}\right) + \psi(1 - \lambda_i)}{1 - \psi\lambda_i} = 1 + \frac{(\psi - 1)\lambda_i^{m+1}}{1 - \psi\lambda_i} \tag{5.5}$$

is an eigenvalue of $\tilde{M}^{-1}(I - \alpha P)$.

If we precondition (1.2) by using $\bar{M}$, then

$$\bar{M}^{-1}(I - \alpha P) = \left(I + \psi R + (\psi R)^2 + \cdots + (\psi R)^s \right) \left((1-\psi) \sum_{i=0}^{m-1} R^{i+1} + I \right) (I - R)$$

and

$$\hat{\lambda}_i = \frac{\left(1 - (\psi\lambda_i)^{s+1}\right)\left((\psi - 1)\lambda_i^{m+1} + (1 - \psi\lambda_i)\right)}{1 - \psi\lambda_i} \tag{5.6}$$

is an eigenvalue of the matrix $\bar{M}^{-1}(I - \alpha P)$.

From (5.5), we obtain

$$\begin{aligned} \left|\tilde{\lambda}_i - 1\right| &= \left|\frac{(\psi-1)\lambda_i^{m+1}}{1-\psi\lambda_i}\right| \\ &\leq \left|\frac{(\psi-1)\rho(R)^{m+1}}{1-\psi\rho(R)}\right| \\ &< \frac{1-\psi}{1-\psi} = 1 \end{aligned} \tag{5.7}$$

with $|\lambda_i| < 1$. Thus, all the eigenvalues of $\tilde{M}^{-1}(I - \alpha P)$ are located in a circle centered at (1,0) with the radius unity.

According to (5.6), we have

$$\begin{aligned} \left|\hat{\lambda}_i - 1\right| &= \left|\frac{\left(1-(\psi\lambda_i)^{s+1}\right)\left((\psi-1)\lambda_i^{m+1}+(1-\psi\lambda_i)\right)}{1-\psi\lambda_i} - 1\right| \\ &= \left|\frac{(\psi-1)\lambda_i^{m+1}\left(1-(\psi\lambda_i)^{s+1}\right)-(\psi\lambda_i)^{s+1}(1-\psi\lambda_i)}{1-\psi\lambda_i}\right| \\ &\leq \left|\frac{(\psi-1)\lambda_i^{m+1}\left(1-(\psi\lambda_i)^{s+1}\right)}{1-\psi\lambda_i}\right| + |(\psi\lambda_i)^{s+1}| \\ &< \left|\lambda_i^{m+1}\left(1-(\psi\lambda_i)^{s+1}\right)\right| + |(\psi\lambda_i)^{s+1}| \\ &\leq \rho(R)^{m+1}\left(1+(\psi\rho(R))^{s+1}\right) + (\psi\rho(R))^{s+1} \\ &= \rho(R)^{m+1} + (\psi\rho(R))^{s+1}\left(1+\rho(R)^{m+1}\right), \end{aligned} \tag{5.8}$$

then all the eigenvalues of $\bar{M}^{-1}(I - \alpha P)$ are located in a circle centered at (1,0) with the radius $\rho(R)^{m+1} + (\psi\rho(R))^{s+1}\left(1+\rho(R)^{m+1}\right)$. Let $m = s = 1$, if $\psi = \rho(R) = 0.7$, then

$$\rho(R)^{m+1} + (\psi\rho(R))^{s+1}\left(1+\rho(R)^{m+1}\right) = 0.847749 < 1,$$

so $\bar{M}$ can be used as an appropriate preconditioner for the Krylov subspace methods for solving the linear system (1.2).

When using Krylov subspace method for the preconditioned linear system (5.2), a residual equation will be solved as follows:

$$\bar{M}z = r. \tag{5.9}$$

Let $m = s = 1$, then

$$\bar{M} = \left((I + \psi M^{-1}N)((1-\psi)M^{-1}NM^{-1} + M^{-1})\right)^{-1}. \tag{5.10}$$

By making use of (5.10) and $z = \bar{M}^{-1}r$, then the following algorithm can be obtained for solving (5.9).

Algorithm 4. We can calculate the vector z in (5.9) by the following steps:

(1) $Mt = r$;
(2) $u = (1-\psi)Nt$;
(3) $Mv = u$;
(4) $\zeta = t + v$.
(5) $M\varpi = \psi N\zeta$
(6) $z = \zeta + \varpi$

Moreover, for some special matrix splittings of $I - \alpha P$, it is simple to obtain the preconditioner $\bar{M}$. For example, Based on the matrix splitting $M = I$ and $N = \alpha P$ with $m = s = 1$, then

$$\bar{M} = \left((I + \psi\alpha P)((1-\psi)\alpha P + I)\right)^{-1}.$$

For this case, the vector z can be easily computed with several matrix-vector products:

(1) $\tilde{t} = \alpha(1-\psi)Pr$;
(2) $\hat{v} = r + \tilde{t}$;
(3) $\hat{\zeta} = \alpha\psi P\hat{v}$;
(4) $z = \hat{v} + \hat{\zeta}$.

6. The choices of the parameters

In this section, we will discuss the choices of the parameters in the GMMS iteration method, the preconditioners $\tilde{M}$ and $\bar{M}$, respectively. Since it is difficult to obtain the optimal parameters, then we only give some heuristic strategies for the choices of the parameters.

6.1. The choices of the parameters in the GMMS iteration method

It follows from (4.2) that the iteration matrix of the GMMS iteration method is

$$\begin{aligned} \mathcal{R}_{GMMS} &= (1-\psi)(M-\psi N)^{-1}N(M^{-1}N)^m \\ &= (1-\psi)(I-\psi R)^{-1}R^{m+1} \end{aligned} \tag{6.1}$$

with $R = M^{-1}N$.

At first, we consider the choice of the parameter ψ. Let λ_i be an eigenvalue of the matrix R with $|\lambda_i| < 1$, so

$$\theta_i = (1-\psi)(1-\psi\lambda_i)^{-1}\lambda_i^{m+1} \tag{6.2}$$

is an eigenvalue of the iteration matrix $\mathcal{R}_{GMMS}$. Then from (6.2) we have

$$\begin{aligned} \rho(\mathcal{R}_{GMMS}) &= \max_i |\theta_i| \\ &= \max_i \left|(1-\psi)(1-\psi\lambda_i)^{-1}\lambda_i^{m+1}\right| \\ &\leq \tfrac{(1-\psi)\rho(R)^{m+1}}{1-\psi\rho(R)}. \end{aligned} \tag{6.3}$$

Let $\hat{f}(\psi) = \frac{(1-\psi)\rho(R)^{m+1}}{1-\psi\rho(R)}$, by simple calculation, then

$$\hat{f}'(\psi) = \frac{\rho(R)^{m+1}(\rho(R)-1)}{(1-\psi\rho(R))^2} < 0. \tag{6.4}$$

Thus, $\hat{f}(\psi)$ is monotonically decreasing, and $\hat{f}(\psi)$ is smaller for larger ψ. However, similar to the analysis in [5,24,34], the outer iterations (3.3) converge faster if ψ is close to 1, while the inner iterations (3.4) is faster if ψ tends to zero. In practice, the values of $\psi \in [0.5, 1)$ are appropriate choices for the GMMS iteration method. It is important to point out that an appropriate parameter ψ only reduces the upper bound of the spectral radius of the iteration matrix (6.4), but does not decrease the spectral radius itself. However, the choices of parameter $\psi \in [0.5, 1)$ can achieve better numerical results, which is verified in Section 7.

For the parameter m_k, just as the analysis in [5,34], which is also true for m_k. A larger m_k may spend a long computational time performing inner iterations (3.4), just to implement a single outer iteration (3.3) and m iterations (3.2) at a time, then slows the overall convergence and takes more computational time. For a smaller m_k, on the other hand, may result in inner iterations (3.4) that do not sufficiently approximate the exact solution of inner linear system, then do not yield sufficient progress for the exact PageRank vector. Therefore, $m_k = 3, 4, 5$ may be appropriate choices, which is illustrated through many numerical experiments in Section 7.

For the parameter m, it is clear that a large m can reduce the upper bound in (6.3) and accelerate the GMMS iteration method for solving the PageRank problem. From the numerical results in Section 7, we find that the GMMS iteration method with the values of m around 7 can achieve better convergence performances.

For the GMMS iteration method based on the AOR splitting (3.12), it is important to analyze the choices of the parameters ω, γ to improve its convergence rate. Just as the analyses in [5,34], for large and sparse coefficient matrices in (1.2), the GMMS iteration method converges faster with the values of ω around 1 and $\gamma = 0$, which is confirmed by the numerical results in Section 7.

6.2. The choices of the parameters in the preconditioners $\tilde{M}$ and $\bar{M}$

For the choices of the parameter m, s. From (5.7) and (5.8), it follows that the preconditioned GMRES (PGMRES) method with large m, s usually converges faster than that with small m, s. However, in order to overcome the memory limitation of the computer and reduce the computational cost, small values of m, s are often the appropriate choices in $\tilde{M}$ and $\bar{M}$, such as $m = s = 1$.

For the choice of the parameter ψ. First, we discuss the choice of the parameter ψ in the preconditioner $\tilde{M}$. From (5.5), we have

$$\left|\tilde{\lambda}_i - 1\right| = \left|\frac{(\psi - 1)\lambda_i^{m+1}}{1 - \psi\lambda_i}\right| \leq \frac{(1 - \psi)\rho(R)^{m+1}}{1 - \psi\rho(R)}. \tag{6.5}$$

From (6.4), we know that the function $\frac{(1-\psi)\rho(R)^{m+1}}{1-\psi\rho(R)}$ is monotonically decreasing with ψ, which means that the Krylov subspace methods converges faster by using the preconditioner $\tilde{M}$ with a larger ψ.

For large and sparse linear system (1.2), we prefer to $m = s = 1$ in the preconditioner $\bar{M}$. For the case $m = s = 1$, from (5.6), then we obtain

$$\begin{aligned}\hat{\lambda}_i &= \frac{\left(1-(\psi\lambda_i)^2\right)\left((\psi-1)\lambda_i^2+(1-\psi\lambda_i)\right)}{1-\psi\lambda_i} \\ &= (1 + \psi\lambda_i)\left((\psi - 1)\lambda_i^2 + (1 - \psi\lambda_i)\right).\end{aligned}$$

Thus,

$$\begin{aligned}\left|\hat{\lambda}_i - 1\right| &= \left|(1 + \psi\lambda_i)\left((\psi - 1)\lambda_i^2 + (1 - \psi\lambda_i)\right) - 1\right| \\ &= \left|(\psi - 1 - \psi^2)\lambda_i^2 + \psi(\psi - 1)\lambda_i^3\right| \\ &\leq (\psi^2 - \psi + 1)\rho(R)^2 + (1 - \psi)\psi\rho(R)^3\end{aligned} \tag{6.6}$$

Let

$$\vec{f}(\psi) = (\psi^2 - \psi + 1)\rho(R)^2 + (1 - \psi)\psi\rho(R)^3.$$

By some calculations, we have

$$\vec{f'}(\psi) = (2\psi - 1)(\rho(R)^2 - \rho(R)^3).$$

Since $\rho(R)^2 - \rho(R)^3 > 0$ with $\rho(R) < 1$, then $\vec{f'}(\psi) < 0$ if $0 < \psi < 1/2$ and $\vec{f'}(\psi) > 0$ if $1/2 < \psi < 1$, so the function $\vec{f}(\psi)$ achieves minimum value for $\psi = 0.5$, which implies that the precondtioned Krylov subspace methods converges faster with $\psi = 0.5$. We notice that the optimal $\psi = 0.5$ only have the minimal upper bound in (6.6), however, the PGMRES method usually converges faster with the values of ψ around 0.5, which is verified by Example 3 in Section 7.

For the preconditioner $\bar{M}$ based on the AOR splitting (3.12), we need to consider the choices of the parameters ω, γ. From (5.8), the upper bound is sharper with a smaller $\rho(R)$. From analyses of ω, γ in Section 6.1, the values of ω around 1 and $\gamma = 0$ are better choices in preconditioner $\bar{M}$ for large and sparse linear system (1.2), which is illustrated clearly by Example 3 in Section 7.

7. Numerical results

In this section, several numerical examples are given to show the effectiveness of the proposed algorithm. The numerical experiments are performed in Matlab R2010 on an Intel dual core processor (2.30 GHz, 8GB RAM). Four parameters are used to test these algorithms, which are the iteration step (denoted as IT), the computing time in seconds (denoted as CPU), the number of matrix-vector product (denoted as MV), and the relative residual (denoted as RES) defined by $\frac{\|r_k\|_2}{\|(1-\alpha)v\|_2}$ with $r_k = (1-\alpha)v-(I-\alpha P)x_k$. Six sparse matrices P for the PageRank problem are listed in Table 1, which are obtained from the Internet (available from [42,43]), where "Average Nonzeros" means the average number of the nonzero elements per row. Four damping factors $\alpha = 0.85, 0.90, 0.95, 0.99$ are adopted in all numerical experiments. All algorithms in this section are started with the initial vector $x_0 = v$ for the sake of justice, which are terminated once the residual norms are below 10^{-8}.

Table 1: Five test matrices for PageRank problem.

Name	Size	Nonzeros	Average Nonzeros
Minnesota	2,642× 2,642	6,606	2.50
Email-Enron	36,692×36,692	367,662	10.0
Usroads	129,164× 129,164	330,870	2.60
Stanford-Berkeley	683,446×683,446	7,583,376	11.0
Flickr	820,878×820,878	9,837,214	1.45×10^{-5}
Wikipedia-20051105	1,634,989×1,634,989	19,753,078	7.38×10^{-6}

Example 1. In this example, we compare the GMMS iteration method with the GIO and MPIO iteration methods, respectively, where the GMMS and GIO iteration methods are constructed based on the AOR splitting (3.12). The test matrices are the Minnesota, Usroads, Stanford-Berkeley, Flickr and Wikipedia-20051105 matrices, respectively. The numerical results are reported in Tables 2-6.

First, we compare the GMMS iteration method with the GIO iteration method, where we choose $\varphi = \psi = 0.5$, $\tilde{m}_k = m_k = 2$ and $\omega = \gamma = 1$ in Table 2, $\varphi = \psi = 0.7$, $\tilde{m}_k = m_k = 2$ and $\omega = 0.9, \gamma = 0$ in Table 3, $\varphi = \psi = 0.8$, $\tilde{m}_k = m_k = 2$ and $\omega = 1, \gamma = 0$ in Table 4, respectively. Let $m = 1, 3, 5, 7$ in the GMMS iteration method. From Tables 2-4, it follows that the GMMS iteration method performs better than the GIO iteration method in terms of iteration number and CPU time with $m > 1$, such as the cases $m = 3, 5, 7$, which is more obvious for larger α, for example, the case $\alpha = 0.99$ in Tables 2-4.

Next, we examine the performance of the GMMS iteration method with the MPIO iteration method. Let $\beta = \alpha\psi, \psi = 0.5$, $\kappa = m_k = 2$ and $\omega = 1, \gamma = 0$ in Table 5, $\beta = \alpha\psi, \psi = 0.7$, $\kappa = m_k = 2$ and $\omega = 1, \gamma = 0$ in Table 6, respectively. From Tables 5 and 6, we notice that the GMMS iteration method outperforms the MPIO iteration method in both iteration number and CPU time, especially for larger α, such as the case $\alpha = 0.99$ in Table 6. From Tables 2-6, we observe that the GMMS iteration method can achieve better numerical results for the appropriate values of m, for example, the cases $m = 5, 7$.

Table 2: Numerical results for the Minnesota matrix

α		GIO	GMMS(m=1)	GMMS(m=3)	GMMS(m=5)	GMMS(m=7)
0.85	IT	33(66)	20(80)	11(66)	8(64)	6(60)
	CPU	0.64520	0.64298	0.49107	0.47011	0.41911
	RES	6.50×10^{-9}	4.71×10^{-9}	4.93×10^{-9}	2.15×10^{-9}	2.88×10^{-9}
0.90	IT	48(96)	29(116)	16(96)	11(88)	9(90)
	CPU	0.92670	0.93342	0.71608	0.64091	0.64400
	RES	9.37×10^{-9}	7.61×10^{-9}	7.87×10^{-9}	8.50×10^{-9}	2.95×10^{-9}
0.95	IT	95(190)	57(228)	32(192)	22(176)	17(170)
	CPU	1.78376	1.77767	1.42875	1.25661	1.18939
	RES	8.89×10^{-9}	8.45×10^{-9}	7.00×10^{-9}	7.66×10^{-9}	6.52×10^{-9}
0.99	IT	453(906)	272(1088)	151(906)	105(840)	80(800)
	CPU	8.46769	8.32363	6.48976	5.85051	4.92910
	RES	9.98×10^{-9}	9.78×10^{-9}	9.82×10^{-9}	9.21×10^{-9}	9.68×10^{-9}

Example 2. In this example, we make a comparison of the convergence performances of the GMMS iteration method with different parameters. The test matrices are the Minnesota, Email-Enron, Usroads, Stanford-Berkeley, Flickr and Wikipedia-20051105 matrices, respectively. The numerical results are listed in Tables 7-11 and Figs. 1-6.

Tables 7-9 contain the iteration numbers of the GMMS iteration method with different m_k, where we adopt $\psi = 0.6$, $m = 3$, $\omega = 0.7$ and $\gamma = 0$ in Table 7, $\psi = 0.7$, $m = 3$, $\omega = 0.8$ and $\gamma = 0$ in Table 8, $\psi = 0.8$, $m = 3$, $\omega = 0.9$ and $\gamma = 0$ in Table 9, respectively. From Tables 7-9, it follows that the GMMS iteration method needs less iteration number for larger m_k, but also takes more CPU time and matrix-vector products at the same time. Thus, $m_k = 2, 3, 4$ can be used for the GMMS iteration method in practice.

Table 3: Numerical results for the Usroads matrix

α		GIO	GMMS(m=1)	GMMS(m=3)	GMMS(m=5)	GMMS(m=7)
	IT	50(100)	32(128)	18(108)	13(104)	10(100)
0.85	CPU	1.18168	1.11038	0.83740	0.75186	0.69099
	RES	9.04×10^{-9}	6.88×10^{-9}	8.83×10^{-9}	5.84×10^{-9}	5.89×10^{-9}
	IT	76(152)	48(192)	28(168)	20(160)	15(150)
0.90	CPU	1.79363	1.67022	1.29810	1.15489	1.04693
	RES	9.37×10^{-9}	8.70×10^{-9}	6.91×10^{-9}	5.34×10^{-9}	7.63×10^{-9}
	IT	153(306)	97(388)	56(336)	39(312)	30(300)
0.95	CPU	3.61575	3.38990	2.57614	2.27484	1.44494
	RES	9.81×10^{-9}	8.81×10^{-9}	8.16×10^{-9}	8.94×10^{-9}	9.06×10^{-9}
	IT	759(1518)	478(1912)	275(1650)	193(1544)	149(1490)
0.99	CPU	17.6137	16.4251	12.5791	11.0476	10.3039
	RES	9.97×10^{-9}	9.91×10^{-9}	9.71×10^{-9}	9.64×10^{-9}	9.33×10^{-9}

Table 4: Numerical results for the Flickr matrix

α		GIO	GMMS(m=1)	GMMS(m=3)	GMMS(m=5)	GMMS(m=7)
	IT	55(110)	35(140)	21(126)	15(120)	12(120)
0.85	CPU	21.5950	21.8492	17.0091	15.5959	16.0248
	RES	8.38×10^{-9}	9.55×10^{-9}	5.36×10^{-9}	4.18×10^{-9}	2.08×10^{-9}
	IT	80(160)	51(204)	30(180)	21(168)	17(170)
0.90	CPU	30.3534	33.2581	25.7794	21.2547	21.0875
	RES	8.76×10^{-9}	9.85×10^{-9}	8.13×10^{-9}	9.43×10^{-9}	3.77×10^{-9}
	IT	139(278)	90(360)	52(312)	37(296)	29(290)
0.95	CPU	45.0251	53.2527	47.0028	42.3921	40.3794
	RES	9.48×10^{-9}	8.05×10^{-9}	9.52×10^{-9}	8.09×10^{-9}	6.10×10^{-9}
	IT	480(960)	308(1232)	180(1080)	127(1016)	98(980)
0.99	CPU	199.494	199.026	144.260	140.461	122.419
	RES	9.74×10^{-9}	9.95×10^{-9}	9.68×10^{-9}	9.73×10^{-9}	9.91×10^{-9}

Table 5: Numerical results for the Stanford-Berkeley matrix

		MPIO			GMMS		
α	m, $\tilde{m}$	IT(MV)	CPU	RES	IT(MV)	CPU	RES
	m=$\tilde{m}$=1	44(176)	9.1916	8.96×10^{-9}	36(144)	7.5067	9.57×10^{-9}
	m=$\tilde{m}$=3	25(150)	5.5432	5.47×10^{-9}	20(120)	4.7510	8.85×10^{-9}
0.85	m=$\tilde{m}$=5	17(136)	5.8941	7.47×10^{-9}	14(112)	5.2092	7.07×10^{-9}
	m=$\tilde{m}$=7	13(130)	6.3179	7.36×10^{-9}	11(110)	5.3592	4.27×10^{-9}
	m=$\tilde{m}$=1	68(272)	13.886	9.49×10^{-9}	56(224)	11.426	8.46×10^{-9}
0.90	m=$\tilde{m}$=3	38(228)	10.455	8.13×10^{-9}	31(186)	9.4303	8.57×10^{-9}
	m=$\tilde{m}$=5	26(208)	8.6940	9.94×10^{-9}	22(176)	7.6577	5.42×10^{-9}
	m=$\tilde{m}$=7	20(200)	7.1513	8.83×10^{-9}	17(170)	6.0318	5.14×10^{-9}
	m=$\tilde{m}$=1	140(560)	31.182	9.89×10^{-9}	114(456)	27.063	9.74×10^{-9}
0.95	m=$\tilde{m}$=3	78(468)	20.433	9.15×10^{-9}	64(384)	16.704	7.91×10^{-9}
	m=$\tilde{m}$=5	54(432)	22.140	9.07×10^{-9}	45(360)	16.642	7.45×10^{-9}
	m=$\tilde{m}$=7	42(420)	22.598	6.51×10^{-9}	17(370)	16.578	9.31×10^{-9}
	m=$\tilde{m}$=1	668(2672)	149.58	9.88×10^{-9}	606(2424)	116.52	9.95×10^{-9}
0.99	m=$\tilde{m}$=3	371(2226)	102.97	9.89×10^{-9}	337(2022)	91.757	9.75×10^{-9}
	m=$\tilde{m}$=5	257(2056)	101.01	9.76×10^{-9}	233(1864)	79.814	9.95×10^{-9}
	m=$\tilde{m}$=7	197(1970)	87.527	9.32×10^{-9}	179(1790)	79.698	9.20×10^{-9}

Table 6: Numerical results for the Wikipedia-20051105 matrix

		MPIO			GMMS		
α	m, $\tilde{m}$	IT(MV)	CPU	RES	IT(MV)	CPU	RES
0.85	m=$\tilde{m}$=1	38(152)	98.4163	6.75×10^{-9}	32(128)	80.4694	7.02×10^{-9}
	m=$\tilde{m}$=3	22(132)	78.3030	5.53×10^{-9}	18(108)	62.2578	9.23×10^{-9}
	m=$\tilde{m}$=5	15(120)	68.6740	9.08×10^{-9}	13(104)	57.5893	5.80×10^{-9}
	m=$\tilde{m}$=7	12(120)	66.5642	4.47×10^{-9}	11(110)	61.2265	7.86×10^{-9}
	m=$\tilde{m}$=10	9(117)	60.7950	3.16×10^{-9}	8(104)	54.7985	3.77×10^{-9}
0.90	m=$\tilde{m}$=1	57(228)	145.039	8.28×10^{-9}	48(192)	120.021	8.59×10^{-9}
	m=$\tilde{m}$=3	33(198)	117.242	6.95×10^{-9}	28(168)	97.3812	6.57×10^{-9}
	m=$\tilde{m}$=5	23(184)	103.866	7.70×10^{-9}	20(160)	88.3542	4.89×10^{-9}
	m=$\tilde{m}$=7	18(180)	91.8019	5.71×10^{-9}	15(150)	77.0139	7.34×10^{-9}
	m=$\tilde{m}$=10	13(169)	90.8506	9.59×10^{-9}	11(143)	75.0639	9.04×10^{-9}
0.95	m=$\tilde{m}$=1	111(444)	282.749	8.96×10^{-9}	93(372)	232.650	9.49×10^{-9}
	m=$\tilde{m}$=3	64(384)	218.583	8.23×10^{-9}	54(324)	184.198	7.93×10^{-9}
	m=$\tilde{m}$=5	45(360)	202.861	7.83×10^{-9}	38(304)	166.573	7.50×10^{-9}
	m=$\tilde{m}$=7	35(350)	173.304	6.51×10^{-9}	29(290)	146.120	8.60×10^{-9}
	m=$\tilde{m}$=10	26(338)	167.154	6.61×10^{-9}	22(286)	150.403	6.29×10^{-9}
0.99	m=$\tilde{m}$=1	478(1792)	1166.07	9.71×10^{-9}	437(1748)	1083.21	9.72×10^{-9}
	m=$\tilde{m}$=3	275(1650)	887.457	9.44×10^{-9}	251(1506)	767.024	9.71×10^{-9}
	m=$\tilde{m}$=5	193(1544)	855.962	9.35×10^{-9}	176(1408)	708.473	9.76×10^{-9}
	m=$\tilde{m}$=7	149(1490)	831.553	8.97×10^{-9}	136(1360)	725.923	9.27×10^{-9}
	m=$\tilde{m}$=10	111(1443)	799.266	8.69×10^{-9}	101(1313)	711.993	9.44×10^{-9}

Figs. 1-4 depicts the iteration numbers of the GMMS iteration method with different ψ. Let $m_k = 3, \omega = \gamma = 0.9$ in Fig. 1, $m_k = 3, \omega = 0.8, \gamma = 0$ in Fig. 2, $m_k = 3, \omega = 0.7, \gamma = 0$ in Fig. 3, $m_k = 3, \omega = 1, \gamma = 0$ in Fig. 4, respectively. From the convergence performances in Figs. 1-4, it is clear that the GMMS iteration method converges faster for larger ψ, especially for small m, such as the case $m = 1$. However, for large m, we observe that the iteration number does not change obviously for $\psi \in (0.5, 1)$, for example, the cases $m = 3, 5, 7$, then it is better to choose $\psi \in (0.5, 1)$ for the GMMS iteration method.

Now, we verify the choices of the parameters ω, γ for the GMMS iteration method based on the AOR splitting (3.12). The numerical results are reported in Figs. 5, 6 and Tables 10, 11. For large test matrices, it is appropriate to choose $\gamma = 0$ in order to make use of their sparse property. Here we choose $\psi = 0.7, m_k = 3$ in Figs. 5 and 6, $\psi = 0.6, m = 3, m_k = 2$ in Table 10 and $\psi = 0.8, m = 3, m_k = 2$ in Table 11, respectively. From these numerical experiments, it is clear that the GMMS iteration method performs better in terms of iteration number and CPU time with larger ω. Moreover, the better numerical results can be obtained as ω around 1.

Table 7: Numerical results for the Email-Enron matrix with different m_k.

		$m_k = 2$	$m_k = 3$	$m_k = 4$	$m_k = 5$
$\alpha = 0.85$	IT(MV)	30(180)	28(196)	27(216)	27(243)
	CPU	0.61184	0.51696	0.49001	0.62420
	RES	3.67×10^{-9}	8.66×10^{-9}	8.75×10^{-9}	6.31×10^{-9}
$\alpha = 0.90$	IT(MV)	44(264)	41(287)	40(320)	39(351)
	CPU	0.76804	0.73140	0.72064	0.72846
	RES	9.71×10^{-9}	9.64×10^{-9}	7.65×10^{-9}	8.33×10^{-9}
$\alpha = 0.95$	IT(MV)	82(492)	77(539)	74(592)	72(648)
	CPU	1.86866	1.73887	1.43502	2.25038
	RES	9.56×10^{-9}	8.12×10^{-9}	8.05×10^{-9}	8.70×10^{-9}
$\alpha = 0.99$	IT(MV)	343(2058)	318(2226)	305(2440)	298(2682)
	CPU	5.04836	4.92962	5.73543	5.85459
	RES	9.54×10^{-9}	9.67×10^{-9}	9.62×10^{-9}	9.47×10^{-9}

Table 8: Numerical results for the Usroads matrix with different m_k.

		$m_k = 2$	$m_k = 3$	$m_k = 4$	$m_k = 5$
	IT(MV)	21(126)	19(133)	18(144)	17(153)
$\alpha = 0.85$	CPU	0.87018	0.94585	0.94270	1.07402
	RES	6.07×10^{-9}	6.79×10^{-9}	6.72×10^{-9}	9.35×10^{-9}
	IT(MV)	31(186)	29(203)	27(216)	26(234)
$\alpha = 0.90$	CPU	1.23563	1.04231	1.12695	1.59546
	RES	9.26×10^{-9}	6.36×10^{-9}	7.64×10^{-9}	7.81×10^{-9}
	IT(MV)	63(378)	57(399)	54(432)	52(468)
$\alpha = 0.95$	CPU	1.58565	1.40916	1.46817	2.67641
	RES	8.46×10^{-9}	8.83×10^{-9}	8.05×10^{-9}	7.88×10^{-9}
	IT(MV)	309(1854)	280(1960)	263(2104)	252(2268)
$\alpha = 0.99$	CPU	10.9199	12.7027	13.4229	12.3863
	RES	9.92×10^{-9}	9.90×10^{-9}	9.80×10^{-9}	9.91×10^{-9}

Table 9: Numerical results for the Stanford-Berkeley matrix with different m_k.

		$m_k = 2$	$m_k = 3$	$m_k = 4$	$m_k = 5$
	IT(MV)	24(144)	21(147)	19(152)	18(162)
$\alpha = 0.85$	CPU	4.64513	4.13484	4.18906	4.42786
	RES	5.17×10^{-9}	6.83×10^{-9}	9.86×10^{-9}	9.89×10^{-9}
	IT(MV)	36(216)	32(224)	30(240)	28(252)
$\alpha = 0.90$	CPU	6.29236	6.38801	6.61860	6.87349
	RES	8.08×10^{-9}	7.69×10^{-9}	5.51×10^{-9}	6.66×10^{-9}
	IT(MV)	73(438)	65(455)	59(472)	56(504)
$\alpha = 0.95$	CPU	12.7336	12.6569	13.0166	13.6287
	RES	9.33×10^{-9}	8.26×10^{-9}	9.88×10^{-9}	8.23×10^{-9}
	IT(MV)	351(2106)	310(2170)	284(2272)	266(2394)
$\alpha = 0.99$	CPU	59.8736	60.1742	62.5374	78.4841
	RES	9.84×10^{-9}	9.75×10^{-9}	9.47×10^{-9}	9.43×10^{-9}

Table 10: Numerical results for the Flickr matrix with different ω.

		ω=0.3	ω=0.5	ω=0.7	ω=0.9	ω=1	ω=1.15
	IT(MV)	76(456)	45(270)	32(192)	24(144)	22(132)	21(126)
α=0.85	CPU	60.7249	42.9220	30.5575	23.1190	20.1690	20.2193
	RES	8.73×10^{-9}	8.07×10^{-9}	6.24×10^{-9}	8.72×10^{-9}	5.14×10^{-9}	7.29×10^{-9}
	IT(MV)	109(654)	65(390)	46(276)	35(210)	32(192)	34(204)
α=0.90	CPU	103.566	57.1506	44.2808	34.0971	29.9100	32.2891
	RES	8.49×10^{-9}	7.52×10^{-9}	7.04×10^{-9}	8.39×10^{-9}	5.43×10^{-9}	6.79×10^{-9}
	IT(MV)	186(1116)	111(666)	79(474)	61(366)	55(330)	77(462)
α=0.95	CPU	177.450	105.321	75.3854	58.2579	50.6345	60.8820
	RES	9.28×10^{-9}	8.94×10^{-9}	8.31×10^{-9}	8.28×10^{-9}	7.42×10^{-9}	8.86×10^{-9}
	IT(MV)	628(3768)	377(2262)	269(1614)	209(1254)	188(1128)	180(1080)
α=0.99	CPU	593.937	344.917	251.879	198.522	174.634	167.471
	RES	9.99×10^{-9}	9.72×10^{-9}	9.66×10^{-9}	9.60×10^{-9}	9.58×10^{-9}	8.54×10^{-9}

Example 3: In this example, we aim at testing the effectiveness of the preconditioner $\bar{M}$ for the GMRES method, where the preconditioner $\bar{M}$ is generated by the AOR splitting (3.12). The test matrices are the Minnesota, Email-Enron, Usroads, Stanford-Berkeley, Flickr and Wikipedia-20051105 matrices, respectively. The numerical results are reported in Tables 12-15 and Figs. 7-8.

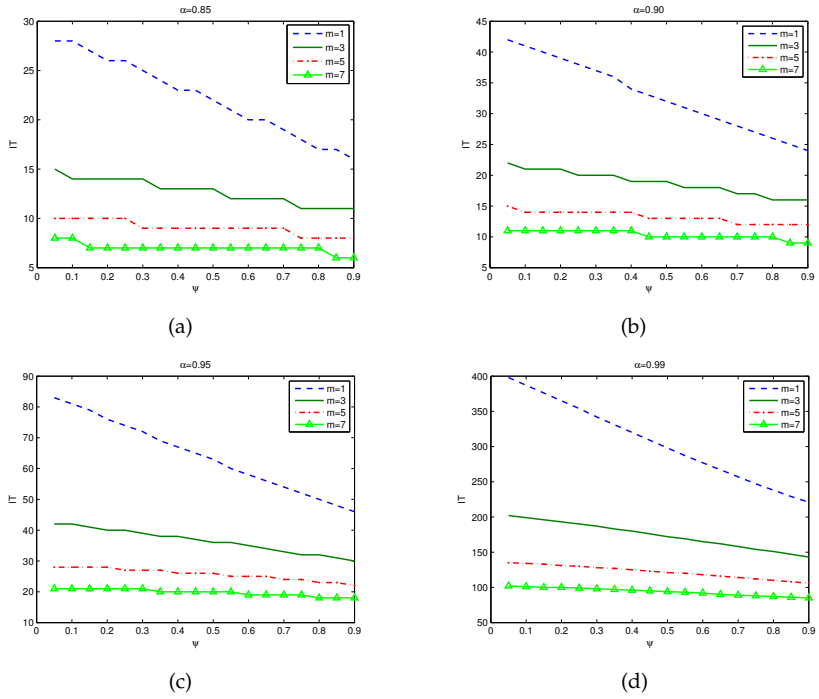

Figure 1: The iteration number for the Minnesota matrix with different ψ.

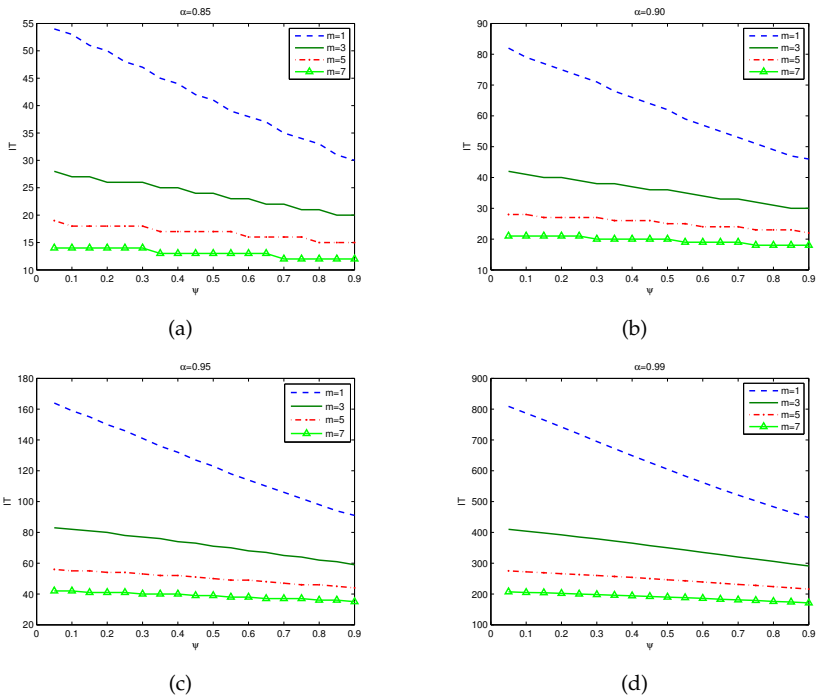

Figure 2: The iteration number for the Email-Enron matrix with different ψ.

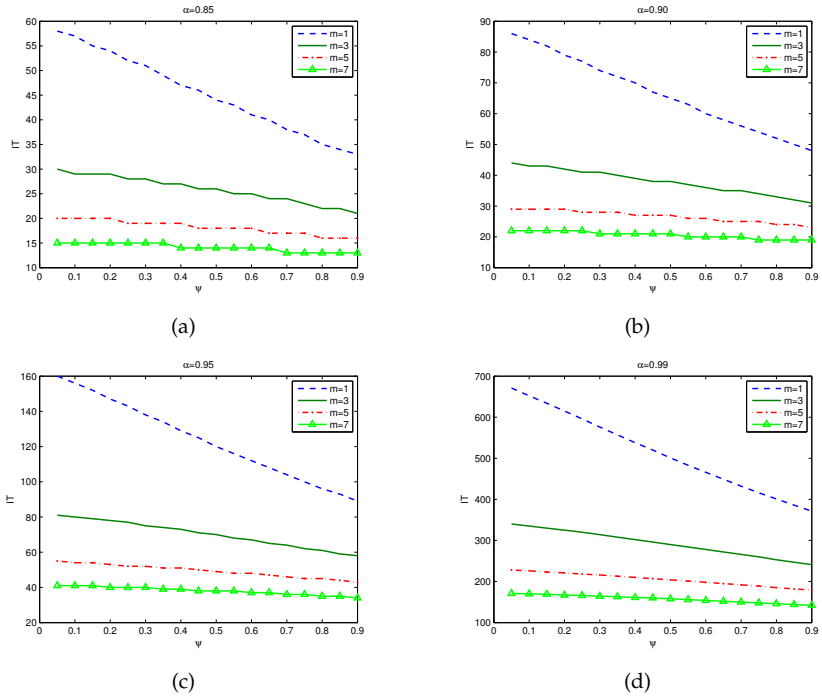

Figure 3: The iteration number for the Usroads matrix with different ψ.

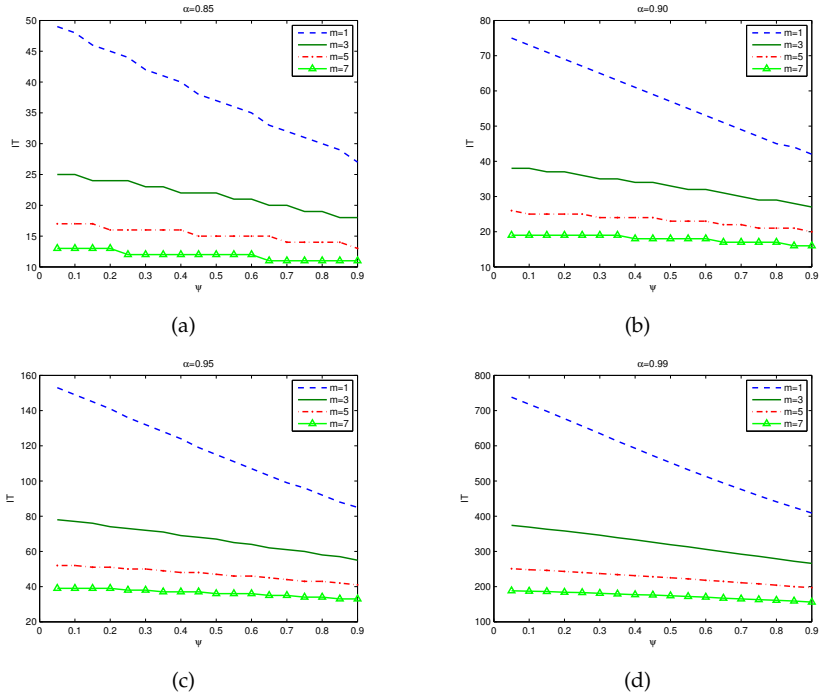

Figure 4: The iteration number for the Stanford-Berkeley matrix with different ψ.

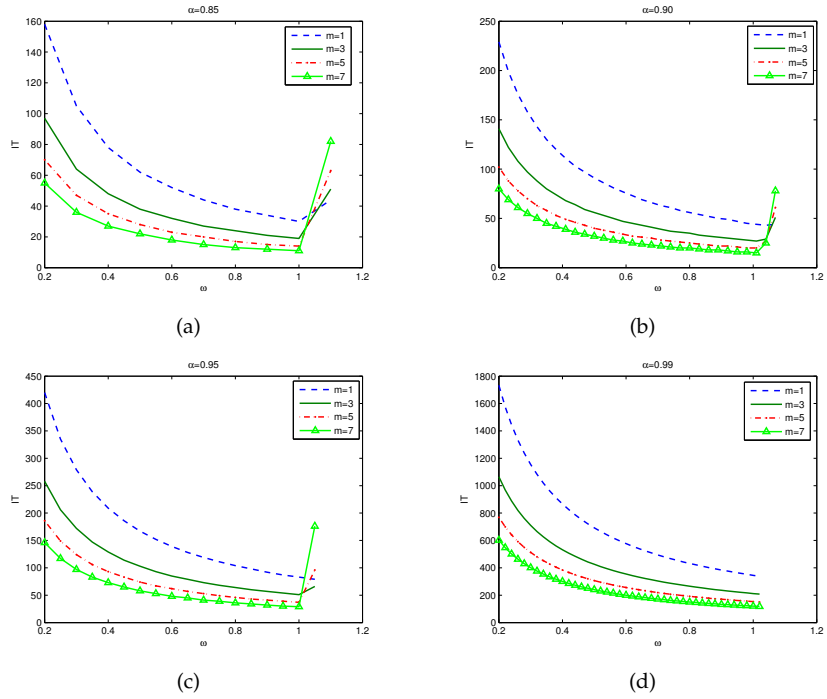

Figure 5: The iteration number for the Email-Enron matrix with different ω.

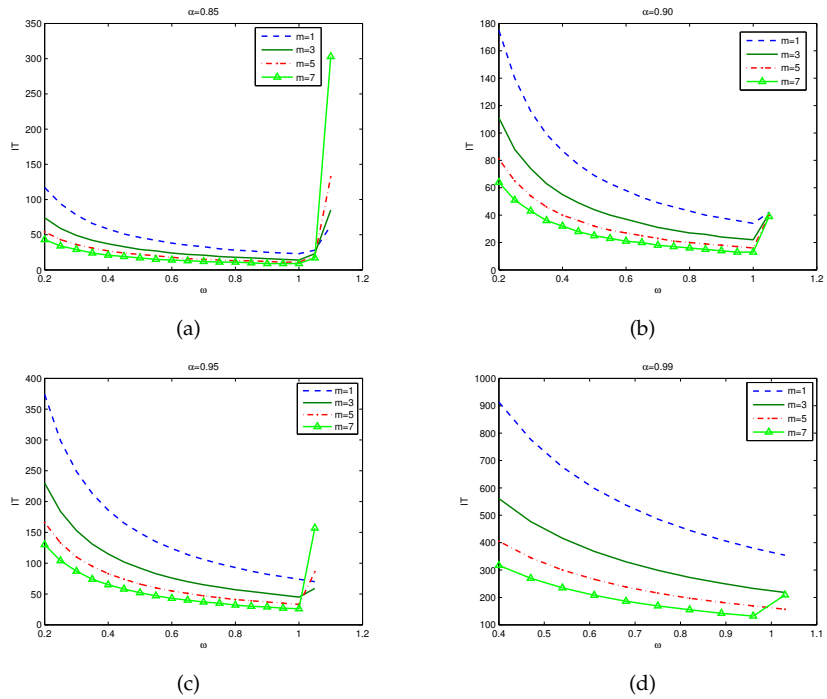

Figure 6: The iteration number for the Usroads matrix with different ω.

Table 11: Numerical results for the Wikipedia-20051105 matrix with different ω.

		ω=0.3	ω=0.5	ω=0.7	ω=0.9	ω=1	ω=1.05
	IT(MV)	63(378)	37(222)	26(156)	20(120)	18(108)	22(132)
α=0.85	CPU	219.615	126.709	89.2781	70.6051	60.7606	67.2119
	RES	7.99×10^{-9}	8.56×10^{-9}	8.42×10^{-9}	7.58×10^{-9}	6.59×10^{-9}	6.48×10^{-9}
	IT(MV)	93(558)	55(330)	39(234)	30(180)	27(162)	26(156)
α=0.90	CPU	308.586	173.337	138.215	99.8963	57.9838	82.6035
	RES	9.29×10^{-9}	9.74×10^{-9}	9.13×10^{-9}	9.03×10^{-9}	8.24×10^{-9}	6.63×10^{-9}
	IT(MV)	178(1068)	106(636)	76(456)	59(354)	53(318)	50(300)
α=0.95	CPU	616.122	319.627	243.726	207.243	180.830	175.913
	RES	9.34×10^{-9}	9.57×10^{-9}	8.24×10^{-9}	7.74×10^{-9}	7.60×10^{-9}	8.60×10^{-9}
	IT(MV)	822(4932)	493(2958)	352(2112)	273(1638)	246(1476)	327(1962)
α=0.99	CPU	2873.99	1686.01	1247.13	856.421	618.277	712.707
	RES	9.85×10^{-9}	9.72×10^{-9}	9.60×10^{-9}	9.84×10^{-9}	9.57×10^{-9}	8.52×10^{-9}

We compare the preconditioned GMRES (PGMRES) method with the GMRES method for solving the PageRank problem in Tables 12 and 13. Let $\psi = 0.8, \omega = 1, \gamma = 0$ in Table 12 and $\psi = 0.7, \omega = 1.1, \gamma = 0$ in Table 13, respectively. It follows from Tables 12, 13 that the PGMRES method converges faster than the GMRES method in term of the iteration number and CPU time with different m, s in the preconditioner $\bar{M}$, especially for larger α, such as the case $\alpha = 0.99$ in Tables 12, 13.

Next, we will discuss the choice of the parameter ψ in the preconditioner $\bar{M}$. We choose $\omega = 1, \gamma = 0$ in Fig. 7, $\omega = 0.9, \gamma = 0$ in Fig. 8, $m = s = 1, \omega = 1.2, \gamma = 0$ in Tables 14, 15, respectively. From Figs. 7, 8 and Tables 14, 15, we observe that the PGMRES method can obtain better numerical results with ψ around 0.6, which is more obvious for larger α, such as the case $\alpha = 0.99$ in Table 14 and the cases $\alpha = 0.90, 0.95, 0.99$ in Table 15, respectively.

Finally, we consider how to choose the parameters ω, γ when the preconditioner $\bar{M}$ is based on the AOR splitting (3.12). Fig. 9 depicts the iteration numbers of the PGMRES method for the Email-Enron, Usroads and Stanford-Berkeley matrices with different ω, respectively. Let $\gamma = 0$, $m = s = 1$ in Fig. 9. From Fig. 9, we can find that the PGMRES method converges faster for larger ω, then it is appropriate to choose the values of ω around 1 for the PGMRES method.

8. Conclusion

In this paper, we present an iteration method for solving the PageRank problem, which is called the GMMS iteration method, and analyze its global convergence in detail. Furthermore, the same idea can be used as a preconditioning technique to accelerate some Krylov subspace methods, such as the GMRES method. The Numerical results on several PageRank problems have showed the superiority of our proposed algorithm. Since the GMMS iteration method is rather parameter-dependent, then how to determine the optimal parameters is an interesting question, which will be further investigated in the future.

9. Acknowledgements

The authors are grateful to thank the anonymous referee for their recommendations and valuable suggestions and Professor Yimin Wei for the communication.

Table 12: Numerical results of the PGMRES method for the Minnesota matrix.

	m, s	IT	CPU	RES
	m=s=1	15	0.07304	2.86×10^{-9}
$\alpha = 0.85$	m=s=2	14	0.07457	6.26×10^{-9}
	m=s=3	10	0.07278	1.88×10^{-9}
	m=s=4	10	0.05386	1.96×10^{-9}
GMRES		30	0.23252	7.68×10^{-9}
	m=s=1	18	0.09855	6.13×10^{-9}
$\alpha = 0.90$	m=s=2	18	0.08887	5.05×10^{-9}
	m=s=3	12	0.06315	4.72×10^{-9}
	m=s=4	12	0.09452	6.45×10^{-9}
GMRES		37	0.25605	9.49×10^{-9}
	m=s=1	26	0.15018	5.56×10^{-9}
$\alpha = 0.95$	m=s=2	26	0.15217	6.01×10^{-9}
	m=s=3	17	0.11918	7.25×10^{-9}
	m=s=4	18	0.10847	5.35×10^{-9}
GMRES		53	0.42809	9.45×10^{-9}
	m=s=1	56	0.39101	7.95×10^{-9}
$\alpha = 0.99$	m=s=2	57	0.42247	8.21×10^{-9}
	m=s=3	38	0.21893	6.38×10^{-9}
	m=s=4	40	0.27674	7.01×10^{-9}
GMRES		116	1.29084	9.34×10^{-9}

Table 13: Numerical results of the PGMRES method for the Usroads matrix.

	m, s	IT	CPU	RES
	m=s=1	14	1.10701	8.93×10^{-9}
$\alpha = 0.85$	m=s=2	15	1.45063	7.70×10^{-9}
	m=s=3	10	1.02632	5.41×10^{-9}
	m=s=4	11	1.74244	3.90×10^{-9}
GMRES		30	2.28122	8.25×10^{-9}
	m=s=1	17	1.21931	6.43×10^{-9}
$\alpha = 0.90$	m=s=2	19	1.51533	8.24×10^{-9}
	m=s=3	12	1.26498	4.81×10^{-9}
	m=s=4	14	1.63505	5.01×10^{-9}
GMRES		38	3.92952	6.45×10^{-9}
	m=s=1	24	2.13781	8.92×10^{-9}
$\alpha = 0.95$	m=s=2	28	3.15262	7.55×10^{-9}
	m=s=3	16	2.03914	6.90×10^{-9}
	m=s=4	21	2.78483	4.90×10^{-9}
GMRES		54	6.69342	7.94×10^{-9}
	m=s=1	54	6.68532	9.35×10^{-9}
$\alpha = 0.99$	m=s=2	63	7.62764	9.91×10^{-9}
	m=s=3	36	4.82273	9.01×10^{-9}
	m=s=4	48	6.80835	7.03×10^{-9}
GMRES		120	21.0056	8.91×10^{-9}

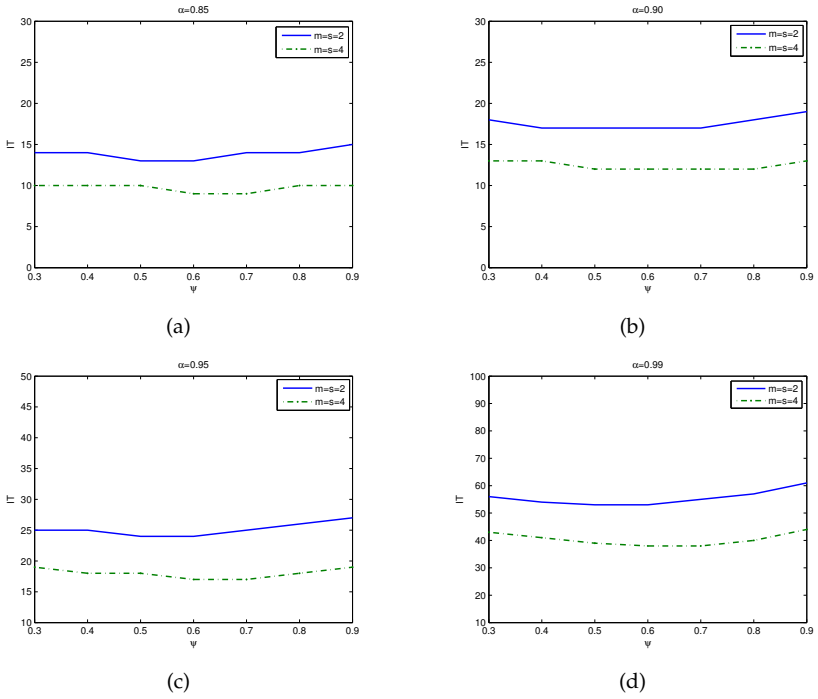

Figure 7: The iteration number for the Minnesota matrix with different ψ.

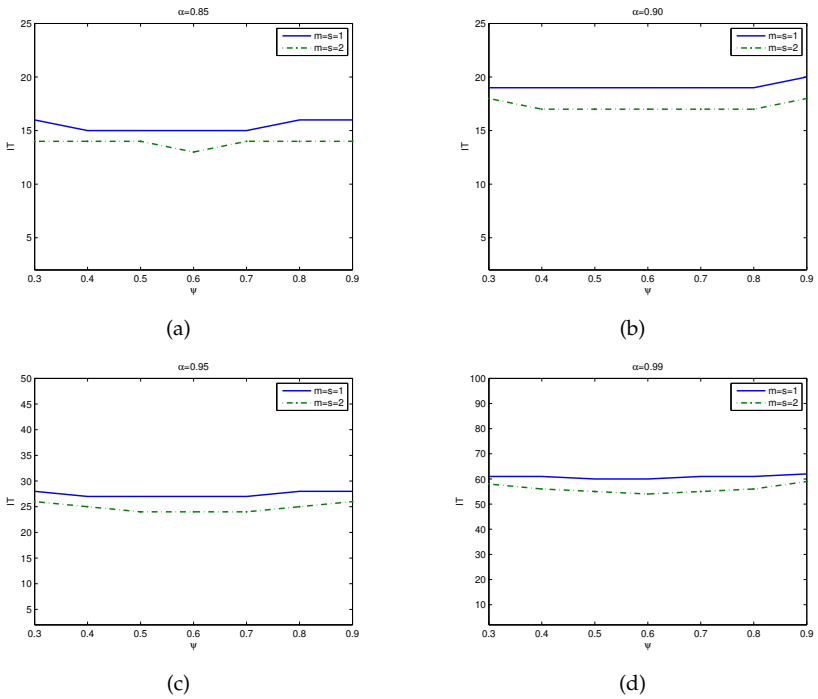

Figure 8: The iteration number for the Usroads matrix with different ψ.

Table 14: Numerical results of the PGMRES method for the Flickr matrix with different ψ.

		GMRES	PGMRES					
			ψ=0.4	ψ=0.5	ψ=0.6	ψ=0.7	ψ=0.8	ψ=0.9
	IT	40	18	18	18	18	19	19
$\alpha = 0.85$	CPU	22.9441	44.5623	23.7771	26.4502	30.8856	27.4087	25.6504
	RES	8.15×10^{-9}	5.89×10^{-9}	5.40×10^{-9}	5.89×10^{-9}	7.60×10^{-9}	4.94×10^{-9}	9.27×10^{-9}
	IT	49	22	22	22	23	23	24
$\alpha = 0.90$	CPU	35.3682	64.8491	41.7243	45.6202	44.6981	43.3974	60.1796
	RES	8.68×10^{-9}	8.86×10^{-9}	7.97×10^{-9}	8.86×10^{-9}	5.39×10^{-9}	9.05×10^{-9}	8.95×10^{-9}
	IT	69	31	31	31	32	33	34
$\alpha = 0.95$	CPU	62.7403	97.9051	66.5882	53.6703	58.0302	59.5976	63.9984
	RES	7.52×10^{-9}	9.85×10^{-9}	8.95×10^{-9}	9.85×10^{-9}	7.98×10^{-9}	7.56×10^{-9}	9.87×10^{-9}
	IT	139	61	60	61	62	64	67
$\alpha = 0.99$	CPU	233.596	149.861	127.473	137.722	138.494	140.796	176.491
	RES	9.85×10^{-9}	7.34×10^{-9}	8.93×10^{-9}	7.34×10^{-9}	7.75×10^{-9}	8.15×10^{-9}	9.10×10^{-9}

Table 15: Numerical results of the PGMRES method for the Wikipedia-20051105 matrix with different ψ.

		GMRES	PGMRES				
			ψ=0.4	ψ=0.5	ψ=0.6	ψ=0.7	ψ=0.8
	IT	42	28	25	23	24	30
$\alpha = 0.85$	CPU	109.805	118.678	114.002	87.3299	105.547	125.943
	RES	8.34×10^{-9}	5.75×10^{-9}	4.59×10^{-9}	8.51×10^{-9}	5.99×10^{-9}	5.68×10^{-9}
	IT	54	33	31	30	32	51
$\alpha = 0.90$	CPU	146.833	179.504	151.817	142.069	174.444	388.379
	RES	7.12×10^{-9}	9.53×10^{-9}	7.04×10^{-9}	5.87×10^{-9}	5.74×10^{-9}	8.18×10^{-9}
	IT	78	50	48	43	46	49
$\alpha = 0.95$	CPU	236.994	352.059	296.166	239.445	259.048	334.832
	RES	9.84×10^{-9}	6.85×10^{-9}	9.66×10^{-9}	7.00×10^{-9}	6.30×10^{-9}	9.07×10^{-9}
	IT	165	94	88	87	94	103
$\alpha = 0.99$	CPU	814.898	772.323	582.547	588.010	612.619	855.340
	RES	8.76×10^{-9}	9.47×10^{-9}	8.77×10^{-9}	9.56×10^{-9}	8.96×10^{-9}	8.15×10^{-9}

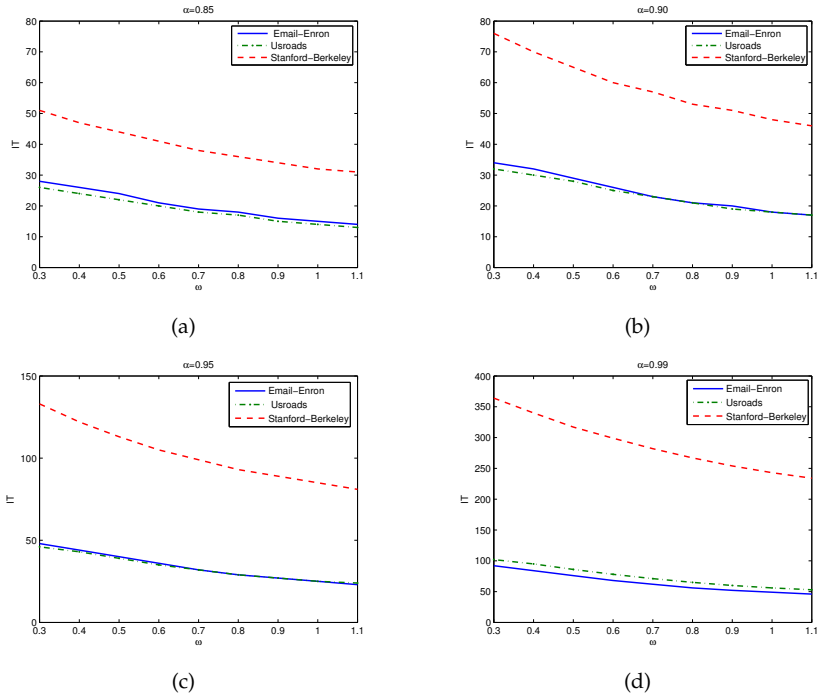

Figure 9: The iteration number for the PGMRES method with different ω.

References

[1] L. Page, S. Brin, R. Motwami, T. Winograd, The Pagerank citation ranking: bringing order to the web, Technical Report, Computer Science Department, Stanford University, 1998.

[2] P. Boldi, M. Santini,S. Vigna, PageRank as a function of the damping factor,in:Proceedings of the 14th International World Web Conference, ACM, NewYork,2005.

[3] T.H. Haveliwala, S.D. Kamvar, D. Klein, C. Manning, G.H. Golub, Computing PageRank using power extrapolation, Stanford University Technical Report, 2003.

[4] A. Arasu, J. Novak, A. Tomkins, J.Tomlin,PageRank computation and the structure of the web: experiments and algorithms, in:Proceedings of 11th International World Web Conference, Honolulu, 2002.

[5] Z.L. Tian, Y. Liu, Y. Zhang, Z.Y. Liu, M.Y. Tian, The general inner-outer iteration method based on regular splittings for the PageRank problem, Appl. Math. Comput. 356 (2019) 479-501.

[6] C. Wen, T.Z. Huang, Z.L. Shen, A note on the two-step matrix splitting iteration for computing PageRank, J. Comput. Appl. Math. 315 (2017) 87-97.

[7] C.Q. Gu, L. Wang, On the multi-splitting iteration method for computing PageRank, J.Appl. Math.Comput. 42 (2013) 479-490.

[8] M. Bianchini, M. Gori, F. Scarselli, Inside PageRank, ACMTrans. Internet Technol. 5 (2005) 92-128.

[9] N. Huang, C.F. Ma, Parallel multisplitting iteration methods based on M-splitting for the PageRank problem, Appl. Math. Comput. 271 (2015) 337-343.

[10] G. Wu, Y. Zhang, Y. Wei, Accelerating the Arnoldi-type algorithm for the PageRank problem and the ProteinRank problem, J. Sci. Comput. 57 (2013) 74-104.

[11] C.Q. Gu, F. Xie, K. Zhang, A two-step matrix splitting iteration for computing PageRank, J. Comput. Appl. Math. 278, 19-28 (2015).

[12] B.Y. Pu, T.Z. Huang, C. Wen, A preconditioned and extrapolation-accelerated GMRES method for PageRank, Appl. Math. Lett. 37 (2014) 95-100.

[13] Y.Z. Song, On the convergence of the MAOR method, J. Comput. Appl. Math. 79 (1997) 299-317.

[14] A. Hadjimos, Accelerated overrelaxation method, Math.Comp. 32 (1978) 149-157.

[15] A.N. Langville, C.D. Meyer, Google's PageRank and Beyond: The Science of Search Engine Rankings, Princet on University Press, Princeton, NJ, 2006.

[16] G.H. Golub, C.F. Van Loan, Matrix Computations, third ed., The Johns Hopkins University Press, Baltimore, London, 1996.

[17] Z.Z. Bai, J.C. Sun, D.R. Wang, A unified framework for the construction of various matrix multisplitting iterative methods for large sparse system of linear equations, Comput. Math. Appl. 32 (1996) 51-76.

[18] G. Wu, Y.M. Wei, An Arnoldi-extrapolation algorithm for computing PageRank, J. Comput. Appl. Math. 234 (2010) 3196-3212.

[19] S.D. Kamvar, T.H. Haveliwala, G.H. Golub, Extrapolation methods for accelerating PageRank computations, Technique Report SCCM 03-02, Stanford
[20] R.S. Varga, Matrix Iterative Analysis, Springer-Verlag, Berlin Heidelberg, 2000
[21] X.Y. Tan, A new extrapolation method for PageRank computations, J. Comput. Appl. Math. 313 (2017) 383-392.
[22] C.Q. Gu,W.W. Wang, An Arnoldi-Inout algorithm for computing PageRank problems, J. Comput. Appl. Math. 309 (2017) 219-229.
[23] Z.X. Jia, Refined iterative algorithms based on Arnoldi's process for large unsymmetric eigenproblems, Linear Algebra Appl. 259 (1997) 1-23.
[24] D.F. Gleich, A.P. Gray, C. Greif, T. Lau, An inner-outer iteration method for computing PageRank, SIAM J. Sci. Comput. 32 (2010) 349-371.
[25] R. Morgan, M. Zeng, A harmonic restarted Arnoldi algorithm for calculating eigenvalues and determining multiplicity, Linear Algebra Appl. 415 (2006) 96-113.
[26] G. Wu, Y.M. Wei, A power-Arnoldi algorithm for computing PageRank, Numer. Linear Algebra Appl. 14 (2007) 521-546.
[27] J.W. Demmel, Applied Numerical Linear Algebra, Society for Industrial and Applied Mathematics, Philadelphia, 1997.
[28] Z.Z. Bai, On convergence of the inner Couter iteration method for computing PageRank, Numer. Algebra Control Optim. 2 (2012) 855-862.
[29] A. Berman, R.J. Plemmons, Nonnegative Matrices in the Mathematical Sciences, Academic Press, NewYork, 1979.
[30] G. Grimmett, D. Stirzaker, Probability and Random Processes, third ed., Oxford University Press, Oxford, UK, 2001.
[31] G.H. Golub, C. Greif, An Arnoldi-type algorithm for computing PageRank, BIT Numerical Mathematics 46 (2006) 759-771.
[32] Q. Yu, Z. Miao, G. Wu, Y. Wei, Lumping algorithms for computing Google's PageRank and its derivative, with attention to unreferenced nodes, Inform. Retrieval, 15 (2012) 503-526.
[33] G. Wu, W. Xu, Y. Zhang, Y. Wei, A preconditioned conjugate gradient algorithm for GeneRank with application to microarray data mining, Data Min. Knowl. Disc. 26 (2013) 27-56.
[34] M.Y. Tian, Y. Zhang, Y.D. Wang, Z.L. Tian, A generalmulti-splitting iteration method for computing PageRank, Comp. Appl. Math. (2019) 38:60. https://doi.org/10.1007/s40314-019-0830-8.
[35] G. Wu, Y. Wei, Arnoldi versus GMRES for computing PageRank: A theoretical contribution to Googles PageRank problem, ACM Transactions on Information Systems, 28 (2010), Article 11.
[36] Y. Lin, X. Shi, Y. Wei, On computing PageRank via lumping the Google matrix, J. Comput. Appl. Math. 224 (2009) 702-708.
[37] Z.L. Shen, T.Z. Huang, B. Carpentieri, X.M. Gu, C. Wen, An efficient elimination strategy for solving PageRank problems, Appl. Math. Comput. 298(2017) 111-122.
[38] C.Q. Gu, X.L. Jiang, Y. Nie, Z.B. Chen, A preprocessed multi-step splitting iteration for computing PageRank, Appl. Math. Comput. 338(2018) 87-100.
[39] G. Wu, Y. Zhang, Y. Wei, Krylov subspace algorithms for computing GeneRank for the analysis of microarray data mining, J. Comput. Biol. 17 (2010), 631-646.
[40] C.Q. Gu, X.L. Jiang, C.C. Shao, Z.B. Chen, A GMRES-Power algorithm for computing PageRank problems, J. Comput. Appl. Math. 343 (2018) 113-123.
[41] Z.L. Tian, X.Y. Liu, Y.D. Wang, P.H. Wen, The Modified Matrix Splitting Iteration Method for Computing PageRank Problem, Filomat. 33 (2019) 725-740.
[42] http://www.cise.ufl.edu/research/sparse/matrices/Gleich/index.html.
[43] http://www.stanford.edu/sdkamvar/research.html.