Page Ranking Algorithm using Gauss Seidel and Jacobi Iterative methods

1Kodathala Sai Varun, 2 Kandagadla Ashok Kumar
1,2 Department of Electronics and Communication Engineering
GITAM School of Technology – Bengaluru

Email: 1kodathalasaivarun@gmail.com,
2kandagadla.ashokkumar@gmail.com.

***Abstract* – We address the PageRank problem of allotting a relative importance value to all web pages in the Internet so that a search engine can use them to sort which pages to display to user. This includes the solution using eigen values and in this paper, we investigate the potential benefits of solving the problem as a solution of a set of linear equations using iterative methods such as Jacobi and Gauss Seidel. The results suggest that proposed method works out better than the previous method. In simulations the data set is characterized for the search keyword 'GITAM' in google search engine where the backlink analysis is obtained from analytics.moz.com.**

Keywords: search engine, Jacobi, Gauss Seidel, page ranking

INTRODUCTION

A search engine is a tool that allows users to supply a query and obtain web pages related to the information they are searching. A crucial task for these tools is some sort of ranking mechanism that selects meaningful links to be presented first. One of the most well-known algorithms is the PageRank from Google, which was initially proposed in [1], to rank pages based on their relative importance and on the number of links to each specific page. The interested reader is referred to [2] for a summary of tools, techniques and results that contribute to the PageRank algorithm and to the surveys in [3], [4] and [5] for the full description of how the search engine of Google operates.

The PageRank algorithm's main idea is that the ranking of a web page is dependent both on the number of links connecting to it and their corresponding ranking. An alternative way of perceiving the algorithm is to think of it as a random walk over the entire internet structure. Then, the rank of each page is the percentage of time this random surfer would spend in that particular page. Both views can be expressed mathematically as the problem of finding the eigenvector associated with the largest eigenvalue of a stochastic matrix, i.e., the stationary distribution of the corresponding Markov chain. The problem can also be viewed as a dual to a class of opinion dynamics [6]. Computation of the page rank has been investigated by the research community using different techniques. In [7], the authors present a distributed randomized algorithm for the PageRank computation. In [8], by the same authors in a collaboration have studied ergodic randomized algorithms that can be applied to the PageRank problem. In essence, both algorithms resort to the power method and incorporates other research work in the field of consensus and distributed linear algorithms. In this setup, each page would compute its rank and send information to the neighbour web pages (i.e., those which it has a link to). A similar type of algorithms is discussed in [9]. In this paper, our aim is to tackle the convergence speed of the power method and show that using Gauss-Seidel iterations can yield a faster convergence in a scenario where a set of processors jointly computes the PageRank for a subset of the entire network structure.

PROBLEM STATEMENT AND PRILIMINARIES

The Pagerank algorithm was invented by Page and Brin around 1998 and used in the prototype of Google's search engine. The objective is to estimate the popularity, or the importance, of a webpage, based on the interconnection of the web. The rationale behind it is (i) a page with more incoming links is more important than a page with less incoming links, (ii) a page with a link from a page which is known to be of high importance is also important. In this note, we study the convergence of the Pagerank algorithm from matrix's point of view.

In practice the number of web pages is huge. In this notes, only examples of small size will be given.

we will consider n set of equations with n unknowns as given below:

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n &= b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \cdots + a_{3n}x_n &= b_3 \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_n &= b_n
\end{aligned}
$$

From the above set of equations one can deduce another n set of equations corresponding from each equation.
Those equations can be formulated as follows:

$$
\begin{aligned}
x_1 &= \frac{b_1 - a_{12}x_2 - a_{13}x_3 - \cdots - a_{1n}x_n}{a_{11}} \\
x_2 &= \frac{b_2 - a_{11}x_1 - a_{13}x_3 - \cdots - a_{1n}x_n}{a_{12}} \\
x_3 &= \frac{b_3 - a_{12}x_2 - a_{11}x_1 - \cdots - a_{1n}x_n}{a_{13}} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
x_n &= \frac{b_n - a_{11}x_1 - a_{12}x_2 - \cdots - a_{1n-1}x_{n-1}}{a_{1n}}
\end{aligned}
$$

These equations can be utilized to compute the values of $x_1, x_2, x_3 \ldots, x_n$ by using iterative methods.

The iterative methods are as follows:

1. Jacobi Method:

In this method the initial guesses are considered and are substituted in the set of equations specified earlier, this computation is carried out till desired output by substituting the obtained values of unknowns in set of equations in each and every iteration.

The generalization of Jacobi method can be given as follows:

Let $x_1^0, x_2^0, x_3^0, \ldots, x_n^0$ be the initial guesses for the variables $x_1, x_2, x_3 \ldots, x_n$ respectively, then after first iteration the value of unknowns turns out to be

$$
\begin{aligned}
x_1^1 &= \frac{b_1 - a_{12}x_2^0 - a_{13}x_3^0 - \cdots - a_{1n}x_n^0}{a_{11}} \\
x_2^1 &= \frac{b_2 - a_{11}x_1^0 - a_{13}x_3^0 - \cdots - a_{1n}x_n^0}{a_{12}} \\
x_3^1 &= \frac{b_3 - a_{12}x_2^0 - a_{11}x_1^0 - \cdots - a_{1n}x_n^0}{a_{13}} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
x_n^1 &= \frac{b_n - a_{12}x_2^0 - a_{13}x_3^0 - \cdots - a_{1n-1}x_{n-1}^0}{a_{1n}}
\end{aligned}
$$

The second iteration is given by,

$$
\begin{aligned}
x_1^2 &= \frac{b_1 - a_{12}x_2^1 - a_{13}x_3^1 - \cdots - a_{1n}x_n^1}{a_{11}} \\
x_2^2 &= \frac{b_2 - a_{11}x_1^1 - a_{13}x_3^1 - \cdots - a_{1n}x_n^1}{a_{12}} \\
x_3^2 &= \frac{b_3 - a_{12}x_2^1 - a_{11}x_1^1 - \cdots - a_{1n}x_n^1}{a_{13}} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
x_n^2 &= \frac{b_n - a_{12}x_2^1 - a_{13}x_3^1 - \cdots - a_{1n-1}x_{n-1}^1}{a_{1n}}
\end{aligned}
$$

Therefore, the general expression for Jacobi method is given by

$$
\begin{aligned}
x_1^{new} &= \frac{b_1 - a_{12}x_2^{old} - a_{13}x_3^{old} - \cdots - a_{1n}x_n^{old}}{a_{11}} \\
x_2^{new} &= \frac{b_2 - a_{11}x_1^{old} - a_{13}x_3^{old} - \cdots - a_{1n}x_n^{old}}{a_{12}} \\
x_3^{new} &= \frac{b_3 - a_{12}x_2^{old} - a_{11}x_1^{old} - \cdots - a_{1n}x_n^{old}}{a_{13}} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
x_n^{new} &= \frac{b_n - a_{12}x_2^{old} - a_{13}x_3^{old} - \cdots - a_{1n-1}x_{n-1}^{old}}{a_{1n}}
\end{aligned}
$$

2. Gauss Seidel Method:

In this method the initial guesses are considered and are substituted in the set of equations specified earlier, this computation is carried out till desired output by substituting the obtained values of unknowns in set of equations in each and every iteration.

The generalization of Gauss Seidel method can be given as follows:

Let $x_1^0, x_2^0, x_3^0, \dots, x_n^0$ be the initial guesses for the variables $x_1, x_2, x_3 \dots, x_n$ respectively, then after first iteration the value of unknowns turns out to be

$$x_1^1 = \frac{b_1 - a_{12}x_2^0 - a_{13}x_3^0 - \cdots - a_{1n}x_n^0}{a_{11}}$$
$$x_2^1 = \frac{b_2 - a_{11}x_1^1 - a_{13}x_3^0 - \cdots - a_{1n}x_n^0}{a_{12}}$$
$$x_3^1 = \frac{b_3 - a_{12}x_2^1 - a_{11}x_1^1 - \cdots - a_{1n}x_n^0}{a_{13}}$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$
$$x_n^1 = \frac{b_n - a_{12}x_2^1 - a_{13}x_3^1 - \cdots - a_{1n-1}x_{n-1}^1}{a_{1n}}$$

The second iteration is given by,

$$x_1^2 = \frac{b_1 - a_{12}x_2^1 - a_{13}x_3^1 - \cdots - a_{1n}x_n^1}{a_{11}}$$
$$x_2^2 = \frac{b_2 - a_{11}x_1^2 - a_{13}x_3^1 - \cdots - a_{1n}x_n^1}{a_{12}}$$
$$x_3^2 = \frac{b_3 - a_{12}x_2^2 - a_{11}x_1^2 - \cdots - a_{1n}x_n^1}{a_{13}}$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$
$$x_n^2 = \frac{b_n - a_{12}x_2^2 - a_{13}x_3^2 - \cdots - a_{1n-1}x_{n-1}^2}{a_{1n}}$$

Therefore, the general expression for Jacobi method is given by

$$x_1^{new} = \frac{b_1 - a_{12}x_2^{old} - a_{13}x_3^{old} - \cdots - a_{1n}x_n^{old}}{a_{11}}$$
$$x_2^{new} = \frac{b_2 - a_{11}x_1^{new} - a_{13}x_3^{old} - \cdots - a_{1n}x_n^{old}}{a_{12}}$$
$$x_3^{new} = \frac{b_3 - a_{12}x_2^{new} - a_{11}x_1^{new} - \cdots - a_{1n}x_n^{old}}{a_{13}}$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots$$
$$x_n^{new} = \frac{b_n - a_{12}x_2^{ne} - a_{13}x_3^{new} - \cdots - a_{1n-1}x_{n-1}^{new}}{a_{1n}}$$

DATA STRUCTURING AND MODELLING

Since, whole internet WWW vector table is inaccessible we formulated the vector map for the websites appeared for key word search 'GITAM' obtained using the open source website **analytics.moz.org**. The top four obtained results are the following websites

1. www.gitam.edu
2. https://collegedunia.com/university/25354-gitam-visakhapatnam
3. https://collegedunia.com/university/55950-gitam-hyderabad
4. https://en.wikipedia.org/wiki/Gandhi_Institute_of_Technology_and_Management

The model graph obtained from the data set is as follows:

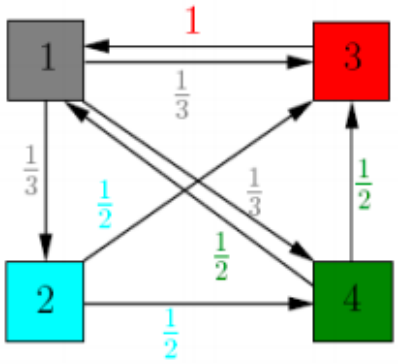


Figure 1: Model Graph for Four Web Pages

Node Terminology:

Node	Hyperlink
A	www.gitam.edu
C	https://en.wikipedia.org/wiki/ Gandhi_Institute_of_Technology_ and_Management
B	https://collegedunia.com/university/ 55950-gitam-hyderabad
D	https://collegedunia.com/university/ 25354-gitam-visakhapatnam

The equations formulated from above graph are as follows:

- Initially consider node 1 where there are three outcomes so we can evaluate its outgoing probability as $\frac{1}{3}$.
- Considering node 2 where there are two outbound links and therefore its outgoing probability is $\frac{1}{2}$
- Considering node 3 where there is one out bound links and therefore its outgoing probability is 1

- Considering node 4 where there are two out bound links and therefore its outgoing probability is $\frac{1}{2}$

From above hypothesis –

Let x_i represent the weightage of inbound link for i^{th} node. Where i refer to the node (A to D)

$$x_1 = 0\ x_1 + \ 0\ x_2 + 1\ x_3 + \frac{1}{2}\ x_4$$
$$x_2 = \frac{1}{3}\ x_1 + \ 0\ x_2 + 0\ x_3 + 0\ x_4$$
$$x_3 = \frac{1}{3}\ x_1 + \frac{1}{2}\ x_2 + 0\ x_3 + \frac{1}{2}\ x_4$$
$$x_4 = \frac{1}{3}\ x_1 + \frac{1}{2}\ x_2 + 0\ x_3 + 0\ x_4$$

PRESENT METHODOLOGY

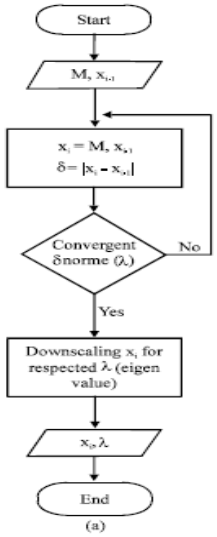
The usual solution for the equations is obtained using the eigen matrix solution. The solution analogy is that eigen solution is fast compared to usual matrix solution that are obtained by solving using Gauss Jordan or Gauss Elimination Method.

Preliminarily we determine the matrix from the equations framed:

$$A = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix}$$

From the above figure, the sum of any column returns to 1 by obeying the laws of probability that total probability is equals to 1.

Flow chart of Eigen Method:



(a)

Algorithm:

1. Initialize matrix A
2. Find Eigen Values of matrix A
3. Normalize the Eigen solution obtained from step 2
4. Display Solution

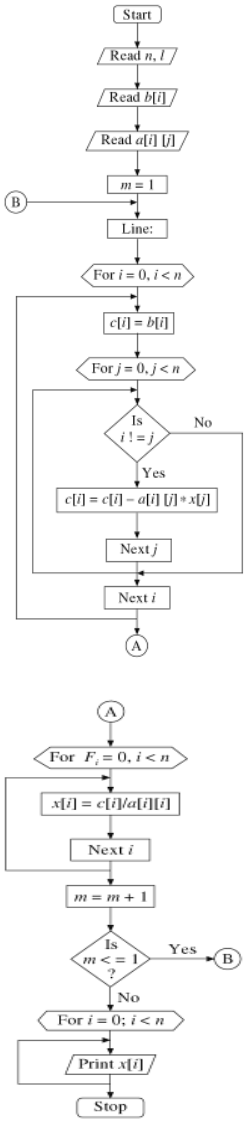
Program:

```
>> A = [0 0 1 1/2;
        1/3 0 0 0;
        1/3 1/2 0 1/2;
        1/3 1/2 0 0];
>> [V,D] = eigs(A);
>> u = V(:,1)

u =

0.721010121751332
0.240336707250444
0.540757591313499
0.360505060875666

>> x = u/sum(u);
>> x

x =

0.387096774193548
0.129032258064516
0.290322580645161
0.193548387096774
```

PROPOSED METHODOLOGY

Since before evaluation of eigen vectors we derived the matrix from the graph and proceeded for the solution. In this inference we can evaluate the matrix using gauss seidel or Jacobi iterative methods.

Flow chart for Jacobi Iterative method:



Algorithm for Jacobi Iterative Method:

1. Start
2. Declare the variables and read the order of the matrix n
3. Read the stopping criteria er
4. Read the coefficients aim as

   Do for i=1 to n

   Do for j=1 to n

   Read a[i][j]

   Repeat for j

   Repeat for i
5. Read the coefficients b[i] for i=1 to n
6. Initialize x0[i] = 0 for i=1 to n
7. Set key=0
8. For i=1 to n

   Set sum = b[i]

   For j=1 to n

   If (j not equal to i)

   Set sum = sum – a[i][j] * x0[i]

   Repeat j

   x[i] = sum/a[i][i]

   If absolute value of ((x[i] – x0[i]) / x[i]) > er,

   then

   Set key = 1

   Set x0[i] = x[i]

   Repeat i
9. If key = 1, then

   Goto step 6

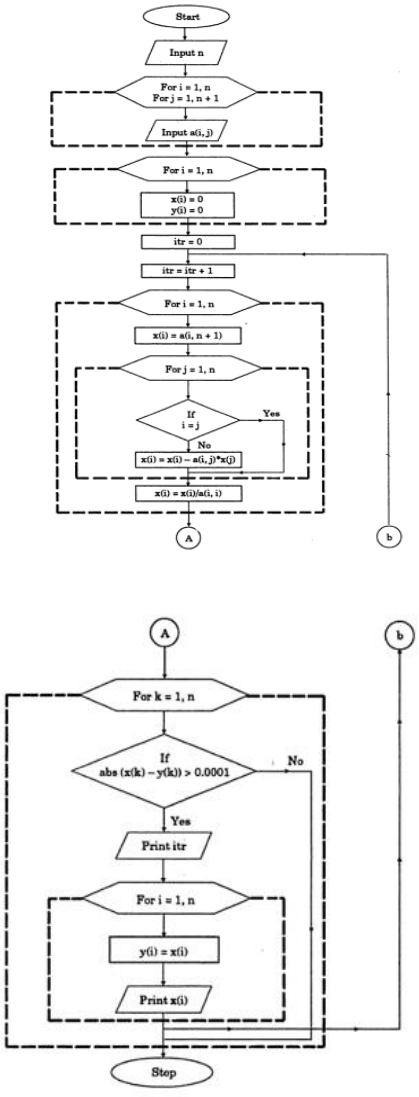
   Otherwise print results

Program For Jacobi Iterative method:

```
function [a,b,c,itr] =
jacobi_user(f1,f2,f3,a0,b0,c0)
x1(1) = a0;
x2(1) = b0;
x3(1) = c0;
i = 2;
while(1)
    x1(i) = f1(x2(i-1),x3(i-1));
    x2(i) = f2(x1(i-1),x3(i-1));
    x3(i) = f3(x1(i-1),x2(i-1));
    er1 = (abs(x1(i)-x1(i-
1))/(x1(i)))*100;
    er2 = (abs(x2(i)-x2(i-
1))/(x2(i)))*100;
```

```
    er3 = (abs(x3(i)-x3(i-
1))/(x3(i)))*100;

if(er1<0.0001&&er2<0.0001&&er3<0.0
001)
        a = x1(i);
        b = x2(i);
        c = x3(i);
        break;
    end
    i=i+1;
end
itr = i-1;
end
```

Flow chart for Gauss Seidel Method:



Algorithm:

1. Start
2. Declare the variables and read the order of the matrix n
3. Read the stopping criteria er
4. Read the coefficients aim as

   Do for i=1 to n

   Do for j=1 to n

   Read a[i][j]

   Repeat for j

   Repeat for i
5. Read the coefficients b[i] for i=1 to n
6. Initialize x0[i] = 0 for i=1 to n
7. Set key=0
8. For i=1 to n

   Set sum = b[i]

   For j=1 to n

   If (j not equal to i)

   Set sum = sum – a[i][j] * x0[i]

   Repeat j

   x[i] = sum/a[i][i]

   If absolute value of ((x[i] – x0[i]) / x[i]) > er, then

   Set key = 1

   Set x0[i] = x[i]

   Repeat i
9. If key = 1, then

   Goto step 6

   Otherwise print results

Program:

```
function [a,b,c,itr] =
gauss_siedel_user(f1,f2,f3,a0,b0,c
0)
x1(1) = a0;
x2(1) = b0;
x3(1) = c0;
i = 2;
while(1)
    x1(i) = f1(x2(i-1),x3(i-1));
    x2(i) = f2(x1(i),x3(i-1));
    x3(i) = f3(x1(i),x2(i));
```

```
    er1 = (abs(x1(i)-x1(i-
1))/(x1(i)))*100;
    er2 = (abs(x2(i)-x2(i-
1))/(x2(i)))*100;
    er3 = (abs(x3(i)-x3(i-
1))/(x3(i)))*100;

if(er1<0.0001&&er2<0.0001&&er3<0.0
001)
        a = x1(i);
        b = x2(i);
        c = x3(i);
        break;
    end
    i=i+1;
end
itr = i-1;
```

RESULTS AND DISCUSSIONS

The simulation is performed for same matrix using different methods collectively – Eigen method, Jacobi, Gauss Seidel the results obtained are tabulated as follows:

Eigen Method	Jacobi Method	Gauss Seidel Method
0.38709	0.4	0.3988
0.12903	0.133	0.1329
0.29032	0.3	0.2999
0.19354	0.2	0.1993

Table 1: Simulation Results of matrix A

The inference from the table 1 is that the solution is differed slightly from the eigen value solution though it's diverged from original solution the analysis is similar to the analysis of eigen solution.

The Ranking from the solution is given by:

Rank	Node
1	1
2	3
3	4
4	2

Table 2: Ranking from Matrix A

The time simulations are performed and the results are obtained as follows:

Number of Nodes	Time Elapsed (ms)		
	Eigen	Jacobi	Seidel
4	0.381704	0.01212	0.009183
5	0.745512	0.14070	0.010661
6	1.288244	0.15664	0.011868
7	2.045684	0.17012	0.012890
8	3.053617	0.18180	0.013774
9	4.347826	0.19209	0.014554

Table 3: Simulation according to number of nodes

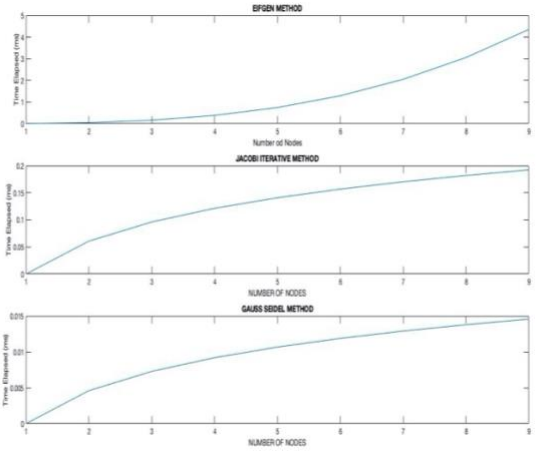


Figure 2: Time v/s nodes Plot

CONCLUSIONS

We addressed the PageRank problem of allotting a relative importance value to all web pages in the Internet so that a search engine can use them to sort which pages to display to user. This includes the solution using eigen values and in this paper, we investigated the potential benefits of solving the problem as a solution of a set of linear equations using iterative methods such as Jacobi and Gauss Seidel. The results suggest that proposed method works out better than the previous method**.** The time complexity of eigen method is referred as $O(n^3)$, time complexity of Jacobi Method is referred as O(30log(n)) and time complexity of gauss seidel method is O(20log(n)). From the analogy of time complexities, we can conclude that gauss seidel has better performance compared to other two methods.

REFERENCES

1. Datta B N. Numerical Linear Algebra and Applications. Brooks/Cole Publishing Company,1995.
2. Jin X Q, Wei Y M, Tam H S. Preconditioning technique for symmetricM-matrices. CALCOLO,2005, 42: 105-113.
3. Kerayechian A, Salkuyeh D K. On the existence, uniqueness and approximation of a class ofelliptic problems. Int. J. Appl. Math., 2002, 11: 49-60.
4. Li W. A note on the preconditioned Gauss seidel (GS) method for linear system. J. Comput.Appl. Math., 2005, 182: 81-90.
5. Meyer C D. Matrix Analysis and Applied Linear Algebra. SIAM,2000.
6. Saad Y. Iterative Methods for Sparse Linear Systems. PWS Press, New York, 1995.
7. Saad Y, Schultz M H. GMRES: a generalized minimal residual algorithm for solving nonsym-metric linear systems. SIAM J. Sci. Stat. Comput., 1986, 7: 856-869.
8. Varga R S. Matrix Iterative Analysis. Prentice Hall, Englewood Cliffs, NJ, 1962.
9. van der Vorst H A. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for thesolution of nonsymmetric linear systems. SIAM J. Sci. Stat.Comput., 1992, 12: 631-644.