POLYNOMIAL PRECONDITIONING FOR THE GENERANK PROBLEM

DAVOD KHOJASTEH SALKUYEH [†], VAHID EDALATPOUR[‡], AND DAVOD HEZARI[§]

Abstract. Identifying key genes involved in a particular disease is a very important problem which is considered in biomedical research. The GeneRank model is based on the PageRank algorithm and preserves many of its mathematical properties. The model brings together gene expression information with a network structure and ranks genes based on the results of microarray experiments combined with gene expression information, for example from gene annotations (GO). In this study, we present a polynomial preconditioned conjugate gradient algorithm to solve the GeneRank problem and study its properties. Some numerical experiments are given to show the effectiveness of the suggested preconditioner.

Key words. Gene network, Gene ontologies, Conjugate gradient, Chebyshev, polynomial, Preconditioner, M-matrix.

AMS subject classifications. 65F10, 65F50; 9208, 92D20.

1. Introduction. Identifying genes involved in a particular disease is regarded as a great challenge in post-genome medical research. Such identification can provide us with a better understanding of the disease. Furthermore, it is often considered as the first step in finding treatments for it. However, the genetic bases of many multifactorial diseases are still uncertain, and modern technologies usually report hundreds or thousands of genes related to a disease of interest. In this context is where gene-disease prioritization methods are of use.

The act of finding the most potentially successful genes among a variety of listed genes has been defined as the gene prioritization problem. Considering rapid growth in biological data sources containing gene-related information such as, for instance, sequence information, microarray expression data, functional annotation data, protein-protein interaction data, and the biological and medical literature, we can see much interest in recent years in developing bioinformatics approaches that can analyze this data and help with the identification of important genes. The common aim in the present study is to prioritize the genes in a way that those related to the disease under study possibly appear at the top of ranking.

In the last decade, several methods have been proposed for ranking or prioritizing genes by relevance to a disease. Some of these methods have been collected at Gene Prioritization Portal: `http://homes.esat.kuleuven.be/~bioiuser/gpp/index.php`. These methods fall into two broad classes. The first class of the methods mostly uses microarray expression data; these methods focus on identifying genes that are differentially expressed in a disease, and use simple statistical measures such as the t-statistic or related classification methods in machine learning to rank genes based on this property. The second class of methods is more general, often making use of a variety of data sources; these methods start with some existing knowledge of 'training' genes already known to be related to the disease under study, and directly or indirectly rank the remaining genes based on their similarity to these training genes. There are also

[†]Corresponding author. Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran
Email: khojasteh@guilan.ac.ir, salkuyeh@gmail.com

[‡]Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran
Email: vedalat.math@gmail.com

[§]Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran
Email: hezari_h@yahoo.com

some methods that rank or prioritize genes based on their overall likelihood of being involved in some disease in general.

Those kinds of methods that purpose to improve an initial ranking obtained from expression data by augmenting it with a network structure derived from other data sources can be related to the methods of second class. For example, the GeneRank algorithm of Morrison et al. [8], is an intuitive change of PageRank algorithm used by the Google search engine that preserves many of its mathematical properties. It combines gene expression information with a network structure derived from gene annotations (gene ontologies (GO)) or expression profile correlations. In the resulting gene ranking algorithm the ranking of genes can be obtained by solving a large linear system of equations. Wu et al. [10, 11] showed that this is equivalent to a symmetric positive definite linear system of equations and analyzed its properties. Then they used the conjugate gradient (CG) algorithm (see [7, 9]) in conjunction with a diagonal scaling to solve the system. Recently, Benzi and Kuhlemann in [4] have shown that the GeneRank problem system is equivalent to systems of linear equations where the coefficient matrices are M-matrix. Then, they have implemented the Chebyshev iteration method and the method based on polynomials of best approximation to solve the system. Their numerical experiments indicate that the Chebyshev acceleration is the most effective one among the tested methods in terms of solution times. In this paper, we propose a polynomial preconditioner for the GeneRank problem and study its properties.

Throughout this paper, we use the following notations, definitions and results. A matrix A is called nonnegative (positive) and is denoted by $A \geq 0$ ($A > 0$) if each entry of A is nonnegative (positive). Similarly, for n-dimensional vectors, by identifying them with $n \times 1$ matrices, we can also define $x \geq 0$ and $x > 0$. For a square matrix A, an eigenvalue of A is denoted by $\lambda(A)$ and the smallest and largest eigenvalues of A are given by $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$, respectively. For any symmetric positive definite matrix A we have $Cond(A) = \|A\|_2\|A^{-1}\|_2 = \lambda_{\max}(A)/\lambda_{\min}(A)$ (see [2]).

DEFINITION 1.1. *(see [5]) A matrix $A = (a_{ij}) \in \mathbb{R}^{n\times n}$ is called a nonsingular M-matrix if it can be expressed in the form $A = rI - B$ where $r > 0$ and B is nonnegative with spectral radius $\rho(B) < r$.*

The rest of the paper is organized as follows. In section 2 we introduce the GeneRank problem in details. Section 3 is devoted for presenting the proposed preconditioner. In section 4 we present some numerical experiments to show the effectiveness of the preconditioner. Finally, concluding remarks are given in section 5.

2. The GeneRank problem formulation. Let the set $G = \{g_1, \ldots, g_n\}$ be n genes in a microarray. Similar to the idea of PageRank, if a gene is connected with many highly ranked genes, it should be highly ranked as well, even if it may have a low rank from the experimental data. In GO, if two genes share at least one annotation with other genes, they are defined to be connected. From this idea, we can build a gene network whose adjacency matrix is W, with entries

$$W_{ij} = \begin{cases} 1, & g_i \text{ and } g_j \ (i \neq j) \text{ have the same annotation in GO,} \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

In contrast to PageRank, the connections are not directed. Thus, instead of a non-symmetric hyperlink matrix, GeneRank considers the symmetric adjacency matrix W

of the gene network, i.e., $W^T = W$. Let

$$deg_i = \sum_{j=1}^{n} w_{ij}$$

Note that since a gene might not be connected to any of the other genes, W may have zero rows. Now, suppose that the diagonal matrix D is defined by $D = \text{diag}(d_1, \ldots, d_n)$, where

$$d_i = \begin{cases} deg_i, & deg_i > 0, \\ 1, & \text{otherwise.} \end{cases} \tag{2.2}$$

Then the GeneRank problem can be written as the following large scale nonsymmetric linear system (Morrison et al. [8])

$$(I - \alpha W D^{-1})x = (1 - \alpha)ex, \tag{2.3}$$

where I denotes the $n \times n$ identity matrix. Also α is the damping factor with $0 < \alpha < 1$, and $ex = [ex_1, ex_2, \ldots, ex_n]^T$, with $ex_i \geq 0$, is the absolute value of the expression change for g_i, $i = 1, 2, \ldots, n$. The solution vector x is called GeneRank vector, and its entries provide information about the significance of a gene. Morrison et al. suggested using $\alpha = 0.5$. However, the optimal choice of α is still an interesting topic and deserves further study.

Note that W is an extremely sparse matrix in general. For the computation of GeneRank vector, Morrison et al. [8], used the Jacobi iteration to solve (2.3), which is inefficient when the problem size is very large or α is very close to 1. Yue et al. [12], reformulated the GeneRank model as a linear combination of three parts, and presented an explicit formulation for the GeneRank vector. In Wu et al. [10], the GeneRank problem was rewritten as a large scale eigenvalue problem, and it was solved by some Arnoldi-type algorithms. In [11], Wu and coworkers observed that the matrix $D - \alpha W$ is symmetric positive definite matrix and showed that the nonsymmetric linear system (2.3) can be rewritten as the following symmetric positive definite (SPD) linear system

$$(D - \alpha W)\hat{x} = (1 - \alpha)ex, \tag{2.4}$$

with $\hat{x} = D^{-1}x$. Note that equation (2.4) is equivalent to equation (2.3). With this modification, methods that are suitable for symmetric systems, can be used for the GeneRank problem. They implemented the Jacobi preconditioner (symmetric diagonal scaling) on $D - \alpha W$. In that case, the preconditioned system is given by

$$(I - \alpha D^{-\frac{1}{2}} W D^{-\frac{1}{2}})\bar{x} = (1 - \alpha)D^{-\frac{1}{2}} ex, \tag{2.5}$$

with $\bar{x} = D^{\frac{1}{2}}\hat{x} = D^{\frac{1}{2}}(D^{-1}x) = D^{-\frac{1}{2}}x$.

Wu and coworkers [11] also showed that the eigenvalues of the preconditioned matrix are bounded as follows:

$$\lambda_{\max}(I - \alpha D^{-\frac{1}{2}} W D^{-\frac{1}{2}}) \leq 1 + \alpha, \tag{2.6}$$

$$\lambda_{\min}(I - \alpha D^{-\frac{1}{2}} W D^{-\frac{1}{2}}) \geq 1 - \alpha. \tag{2.7}$$

These bounds are independent of the size of the matrix, and they only depend on the value of the parameter α used in the GeneRank model. It is noted that Benzi

and Kuhlemann in [4] showed that if $deg_i > 0$, for all i, then the inequality in (2.7) becomes equality. See also Lemma 3.2 below.

As mentioned, Benzi and Kuhlemann in [4] showed that the coefficient matrices of (2.3) and (2.4) have a nice property that we introduce here.

THEOREM 2.1. *Both of the matrices $D - \alpha W$ and $I - \alpha D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ are M-matrices for $0 < \alpha < 1$.*

They then implemented the classical Chebyshev iteration for the GeneRank problem. It is a polynomial scheme to expedite the convergence of the stationary iterative method

$$x^{(k+1)} = x^{(k)} + M^{-1}(b - Ax^k), \quad k = 0, 1, \ldots,$$

to solve a linear system of equation $Ax = b$, in which M is a nonsingular matrix. If the matrix $I - M^{-1}A$ is similar to a symmetric matrix with eigenvalues lying in an interval $[l_{\min}, l_{\max}]$, then the Chebyshev iteration to solve $Ax = b$ is given by (for more details, see [4, 6])

$$\begin{aligned} y^{(k+1)} = & \frac{\omega_{k+1}}{2 - (l_{\min} + l_{\max})} \left\{ 2M^{-1}(b - Ay^{(k)}) + [2 - (l_{\min} + l_{\max})](y^{(k)} - y^{(k-1)}) \right\} \\ & + y^{(k-1)}, \quad k = 1, 2, \ldots, \end{aligned}$$

where $y^{(0)} = x^{(0)}$, $y^{(1)} = x^{(1)}$ and

$$\omega_{k+1} = \frac{1}{1 - \frac{w_k^2}{4\omega^2}}, \quad w_2 = \frac{2\omega^2}{2\omega^2 - 1}, \quad \omega_1 = 1, \quad \omega = \frac{2 - (l_{\min} + l_{\max})}{l_{\max} - l_{\min}}.$$

For the GeneRank problem, it is considered $l_{\min} = 1 - \alpha$, $l_{\max} = 1 + \alpha$, $A = D - \alpha W$, $b = ex$ and $M = D = \mathrm{diag}(A)$.

Numerical results presented in [4] show that the number of iterations of the method is usually more than those of the CG method with the diagonal scaling. However, the cost per iteration is much lower and leads to faster convergence in terms of CPU time.

3. Main results. Wu et al. [10, 11] showed that the coefficient matrix of (2.4) is symmetric positive definite and therefore we can use the CG method to solve the system. As we know the convergence rate of the CG method depends on the condition number of the matrix in question, or more generally the distribution of the eigenvalues. If the eigenvalue distribution of the preconditioned system is better than that of the original one, the convergence will be accelerated drastically. Having this in mind, Wu and coworkers applied the Jacobi preconditioner to the system (2.4) to accelerate the convergence rate of the method. In this case, the linear system (2.5) is obtained. Considering (2.6) and (2.7), we can conclude that increasing α from 0 to 1 can cause an increase in the ratio of $\lambda_{\max}/\lambda_{\min}$ and therefore the coefficient matrix would be increasingly ill-conditioned and the rate of convergence of CG expected to decrease as α increases.

From now on, for the sake of the simplicity, let $J_\alpha = \alpha D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ and $S_\alpha = I - J_\alpha$. It is noted that the matrix S_α is the coefficient matrix of the system (2.5). We now propose the preconditioner $M_\alpha = I + J_\alpha$ for the system (2.5). In this case, the preconditioned system takes the form

$$M_\alpha S_\alpha \bar{x} = M_\alpha b_\alpha, \tag{3.1}$$

where $b_\alpha = (1-\alpha)D^{-\frac{1}{2}}ex$. In the sequel, we investigate the properties of the proposed preconditioner.

THEOREM 3.1. *For every $0 < \alpha < 1$, the matrix $M_\alpha S_\alpha$ is symmetric positive definite M-matrix.*

Proof. Since the matrix $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ is sub-stochastic (nonnegative with row sums less than or equal to 1), we have $\rho(J_\alpha) \leq \alpha < 1$. Hence $\rho(J_\alpha^2) \leq \alpha^2 < 1$, and since J_α is nonnegative it follows from Definition 1.1 that the matrix $T_\alpha = I - J_\alpha$ is an M-matrix. □

LEMMA 3.2. *If $W \neq 0$, then $\lambda_{\min}(S_\alpha) = 1 - \alpha$.*

Proof. Let $x = (x_1, \ldots, x_n)^T$, such that

$$x_i = \begin{cases} 1, & deg_i > 0, \\ 0, & \text{otherwise}, \end{cases}$$

for $i = 1, 2, \ldots, n$. Obviously, we have $Wx = Dx$. Now, observing that $y = D^{\frac{1}{2}}x \neq 0$, we get

$$(I - \alpha D^{-\frac{1}{2}}WD^{-\frac{1}{2}})y = (1-\alpha)y,$$

which is equivalent to $S_\alpha y = (1-\alpha)y$. Now by Eq. (2.7), we see that $1-\alpha$ is the smallest eigenvalue of S_α. □

REMARK 3.3. *In [11], it has been shown that*

$$\lambda_{\max}(WD^{-1}) \leq 1, \quad \lambda_{\min}(WD^{-1}) \geq -1. \tag{3.2}$$

Since the eigenvalues of $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ are the same as those of WD^{-1}, from Eq. (3.2) and Lemma 3.2 we deduce that

$$\lambda_{\max}(I - J_\alpha^2) \leq 1 + \alpha^2, \quad \textit{and} \quad \lambda_{\min}(I - J_\alpha^2) = 1 - \alpha^2. \tag{3.3}$$

THEOREM 3.4. *The spectral condition number of T_α is not greater than that of the matrix S_α, i.e.,*

$$Cond_2(T_\alpha) \leq Cond_2(S_\alpha).$$

Proof. Since both of the matrices S_α and T_α are symmetric positive definite, it is enough to prove that

$$\frac{\lambda_{\max}(T_\alpha)}{\lambda_{\min}(T_\alpha)} \leq \frac{\lambda_{\max}(S_\alpha)}{\lambda_{\min}(S_\alpha)}.$$

For every eigenvalue $\lambda(S_\alpha)$ of S_α, we have $\lambda_{\min}(S_\alpha) \leq \lambda(S_\alpha) \leq \lambda_{\max}(S_\alpha)$. Then,

$$1 - \lambda_{\max}(S_\alpha) \leq 1 - \lambda(S_\alpha) \leq 1 - \lambda_{\min}(S_\alpha).$$

From Lemma 3.2, we have $\lambda_{\min}(S_\alpha) \leq 1$. We now consider two cases, $\lambda_{\max}(S_\alpha) \geq 1$ and $\lambda_{\max}(S_\alpha) < 1$. If $\lambda_{\max}(S_\alpha) \geq 1$, then by Eq. (2.6) and Lemma 3.2 we have $\lambda_{\max}(S_\alpha) - 1 \leq 1 - \lambda_{\min}(S_\alpha)$. Therefore $1 - (1 - \lambda_{\max}(S_\alpha))^2 \geq 1 - (1 - \lambda_{\min}(S_\alpha))^2$ and since the maximum value of $1 - (1 - \lambda(S_\alpha))^2$ is equal to 1, we get

$$Cond(T_\alpha) \leq \frac{1}{1 - (1 - \lambda_{\min}(S_\alpha))^2} \leq \frac{\lambda_{\max}(S_\alpha)}{\lambda_{\min}(S_\alpha)} = Cond(S_\alpha)$$

Now, suppose that $\lambda_{\max}(S_\alpha) \leq 1$. In this case, it is easy to see that

$$Cond(T_\alpha) = \frac{1-(1-\lambda_{\max}(S_\alpha))^2}{1-(1-\lambda_{\min}(S_\alpha))^2} \leq \frac{\lambda_{\max}(S_\alpha)}{\lambda_{\min}(S_\alpha)} = Cond(S_\alpha)$$

Therefore, the proof of theorem is completed. □

This theorem shows that the eigenvalues of the matrix T_α are clustered at least as those of the matrix S_α. As we shall see, for the presented numerical experiments the eigenvalues of T_α are more clustered than those of the matrix S_α.

REMARK 3.5. *From Lemma 3.2, if $W \neq 0$, then $\lambda_{\min}(S_\alpha) = 1-\alpha$. Therefore, according to the relation $\lambda(T_\alpha) = 1-(1-\lambda(S_\alpha))^2$ we have $\lambda_{\min}(T_\alpha) = 1-\alpha^2$. Having in mind that $0 < \lambda(T_\alpha) < 1$, we deduce that $1-\alpha^2 \leq \lambda(T_\alpha) < 1$. Note that $1-\alpha \leq \lambda(S_\alpha) \leq 1+\alpha$.*

It is noted that the matrix M_α is the first degree polynomial preconditioner obtained by truncating the Neumann series expansion of S^{-1} to first order:

$$S_\alpha^{-1} = (I-J_\alpha)^{-1} = I + J_\alpha + J_\alpha^2 + \cdots \approx I + J_\alpha \equiv M_\alpha. \tag{3.4}$$

Some other properties of such a preconditioner can be found in [1, 9]. One may obtain other preconditioners by using higher degree polynomials from Eq. (3.4). Hereafter, we set $\bar{M}_\alpha = I + J_\alpha + J_\alpha^2$ and $\tilde{M}_\alpha = I + J_\alpha + J_\alpha^2 + J_\alpha^3$.

4. Numerical experiments. As mentioned, Wu et al. in [11] successfully employed the Jacobi preconditioner in conjunction with the CG algorithm for the solution of the linear system and deduced that it is the fastest among the tested methods for each of the presented examples. Also, Benzi and Kuhlemann in [4] compare the Chebyshev method and the method based on polynomials of best approximation with the CG method and a Jacobi-preconditioned CG method and they showed that in conjunction with diagonal scaling, Chebyshev acceleration can significantly outperform CG in terms of solution times.

In this section we compare the numerical results of the CG and the Chebyshev iteration methods with the first and third degree polynomial preconditioners with those of the Jacobi preconditioner and Chebyshev acceleration. With attention to (3.3), we consider the first degree polynomial preconditioner of Chebyshev iteration with $l_{\min} = 1-\alpha^2$, $l_{\max} = 1+\alpha^2$, $A = D - \alpha W$, $b = ex$ and $M = D = \mathrm{diag}(A)$.

All the numerical experiments presented in this section were computed in double precision and the algorithms are implemented in MATLAB 7.12.0 and tested on a 64-bit 1.73 GHz intel Q740 core i7 processor and 4GB RAM running windows 7. We use the stopping criteria based on the 1-norm of the residual. That is, we stop iterating as soon as $\|r\|_1 < tol$, where tol is a given tolerance. The initial guess is the null vector. We use two different choices for ex: $ex = (\frac{1}{n})e$, where e is the vector of all ones, and $ex = p$, where p is a randomly chosen probability vector – that is, a random vector with entries in $(0, 1)$. For each adjacency matrix, we use four different values of α to form the corresponding GeneRank matrices $D - \alpha W$: $\alpha = 0.5, 0.75, 0.80, 0.99$. The obtained numerical results are presented in some tables. In all the tables, "CG", "PCG", "Chebyshev", "Chebyshev-M_α", "CG-M_α", "CG-$\tilde{M}_\alpha$", are, respectively, denoted for the CG method implemented for the system (2.4), CG method applied to the system (2.5), Chebyshev iteration to (2.4), preconditioned Chebyshev iteration to the system (2.4) with the preconditioner M_α, and the CG algorithm to the system (2.4) in conjunction with the preconditioners M_α, $\bar{M}_\alpha$ and $\tilde{M}_\alpha$.

TABLE 4.1
Results for the w_All matrix in Example 4.1. Here ex = extr_data.

α	0.50	0.75	0.80	0.99
CG	381(0.700)	441(0.772)	456(0.785)	603(1.012)
PCG	26(0.051)	39(0.082)	42(0.081)	69(0.133)
Chebyshev	23(0.030)	36(0.048)	41(0.057)	177(0.195)
Chebyshev-M_α	15(0.086)	25(0.094)	28(0.108)	126(0.253)
Chebyshev-$\bar{M}_\alpha$	11(0.033)	19(0.106)	23(0.099)	104(0.324)
CG-M_α	11(0.023)	16(0.037)	18(0.044)	30(0.064)
CG-$\bar{M}_\alpha$	10(0.080)	15(0.076)	17(0.056)	29(0.099)
CG-$\tilde{M}_\alpha$	7(0.030)	11(0.046)	13(0.054)	22(0.102)

EXAMPLE 4.1. In this example there are three adjacency matrices, *w_All* which is of size 4047×4047, with 339596 nonzero entries, *w_Up* which is of size 2392×2392, with 128468 nonzero entries and *w_Down* which is of size 1625×1625 with 67252 nonzero entries and three expression change vectors *expr_data*, *expr_dataUp* and *expr_dataDown*. These matrices were constructed using all the three sections of the GO, where a link is presented between two genes if they share a GO annotation. Only genes which are up-regulated are included in *w_Up* and only down-regulated in *w_Down* [8]. The data files are available from [3]. The results for these matrices are given in Tables 4.1, 4.2 and 4.3. In these tables (hereafter for other tables) the number of iterations for the convergence together with the CPU time (in parenthesis) in seconds are given. Here we mention that, in this example, the tolerance *tol* is taken to be 10^{-14}. For this example, as the numerical results show the CG method with the preconditioners M_α, $\bar{M}_\alpha$ and $\tilde{M}_\alpha$ outperform the other tested methods in terms of both the number of iterations and the CPU time. Moreover, the results are especially attractive for α close to 1. We also observe that the number of iterations of the CG method with $\tilde{M}_\alpha$ is always less than those of the CG method with the preconditioner M_α. While the CPU time for the CG method with M_α is less than with the preconditioner $\tilde{M}_\alpha$. As Axelsson noted in [1], using first degree polynomials roughly halves the number of CG iterations (at price of two matrix-vector multiplications per iteration), using third degree polynomials reduces the number of iterations roughly by a factor of three (at the price of three matrix-vector multiplications per iteration) and so on. Hence, polynomial preconditioners do not reduce the total number of matrix-vector multiplications, only the number of vector operations (linked triads, inner products, etc). This means that these methods can be effective only if the coefficient matrices are extremely sparse (as in the GeneRank problem) or, more generally, when the matrix-vector operations are very cheap. This fact can be observed when the results of CG are compared with those of CG-M_α and CG-$\tilde{M}_\alpha$. The last comment which can be posed here is that, the number of iterations of the CG-$\bar{M}_\alpha$ method is slightly less than that of the CG-M_α but the difference is not significant. Additional performed numerical experiments showed that among the preconditioners of the form $I + J_\alpha + J_\alpha^2 + \cdots + J_\alpha^r$, the ones with odd r are preferred over such preconditioners with even r (see [1, page 181]).

For more investigation, in Figures 4.1, 4.2 and 4.3 we depict the eigenvalues distribution of S_α and T_α for test matrices *w_Down* when $\alpha = 0.5$, *w_Up* when $\alpha = 0.75$ and *w_All* when $\alpha = 0.9$, respectively. As seen the eigenvalues of the T_α are more clustered than those of S_α for all the three test matrices.

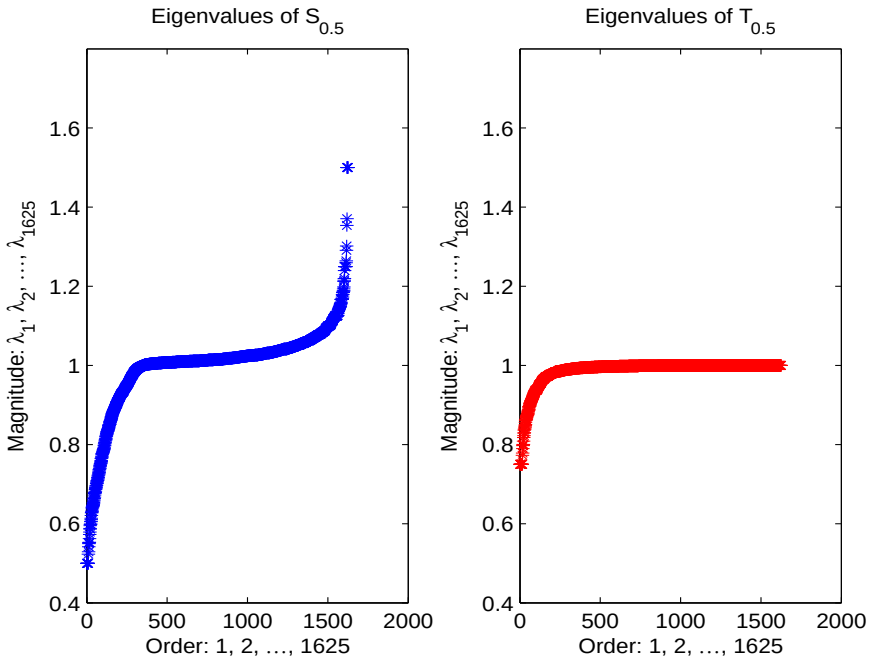

FIG. 4.1. *Eigenvalue distribution of $S_{0.5}$ and $T_{0.5}$ for the matrix w_Down.*

TABLE 4.2
Results for the w_Up matrix in Example 4.1. Here ex = extr_dataUp.

α	0.50	0.75	0.80	0.99
CG	309(0.142)	360(0.162)	377(0.173)	488(0.221)
PCG	26(0.017)	39(0.026)	42(0.022)	70(0.033)
Chebyshev	22(0.008)	36(0.013)	41(0.014)	177(0.058)
Chebyshev-M_α	15(0.019)	25(0.023)	28(0.025)	127(0.071)
Chebyshev-$\bar{M}_\alpha$	11(0.021)	20(0.024)	23(0.028)	105(0.082)
CG-M_α	11(0.006)	16(0.011)	18(0.012)	31(0.017)
CG-$\bar{M}_\alpha$	10(0.009)	15(0.013)	16(0.014)	29(0.025)
CG-$\tilde{M}_\alpha$	7(0.008)	11(0.016)	13(0.014)	22(0.022)

EXAMPLE 4.2. In this example, we use two different types of test data for our experiments. The first matrix is a SNPa adjacency matrix (single-nucleotide polymorphism matrix). This matrix has $n = 152520$ rows and columns, and is very sparse with only 639,248 nonzero entries. The second type is RENGA adjacency matrix (range-dependent random graph model). In our experiments we set $\lambda = 0.9$ and $\beta = 1$, the default values in RENGA. Both of these types of matrices are tested in [11] and [4]. The results for the SNPa matrix are given in Tables 4.4 and 4.5, and the results for the RENGA matrix with $n = 100000$ and $n = 500000$ are given in Tables 4.6 and 4.7. In this example the tolerance *tol* is assumed to be 10^{-10}. All the comments posed in the previous example remain valid.

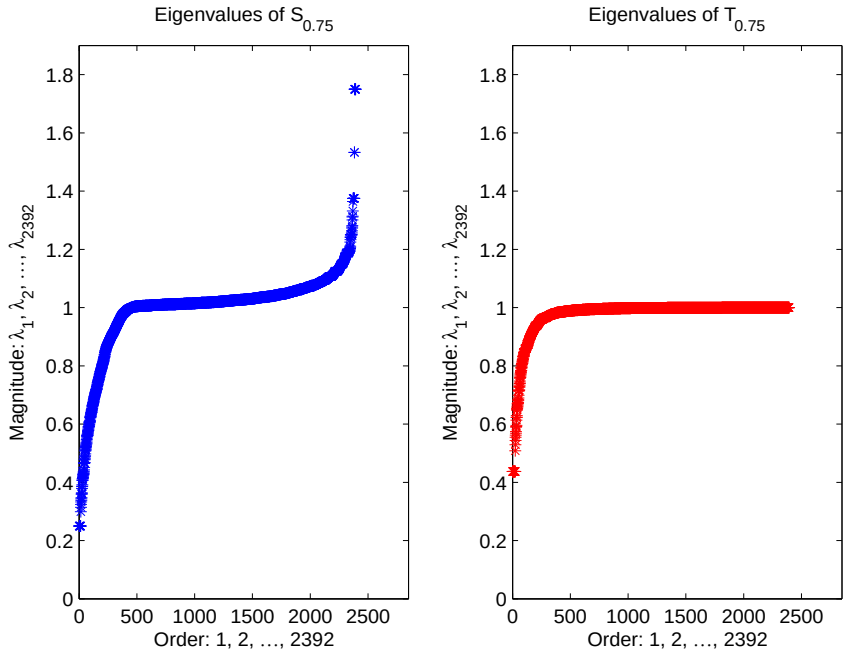

FIG. 4.2. *Eigenvalue distribution of $S_{0.75}$ and $T_{0.75}$ for the matrix w_Up.*

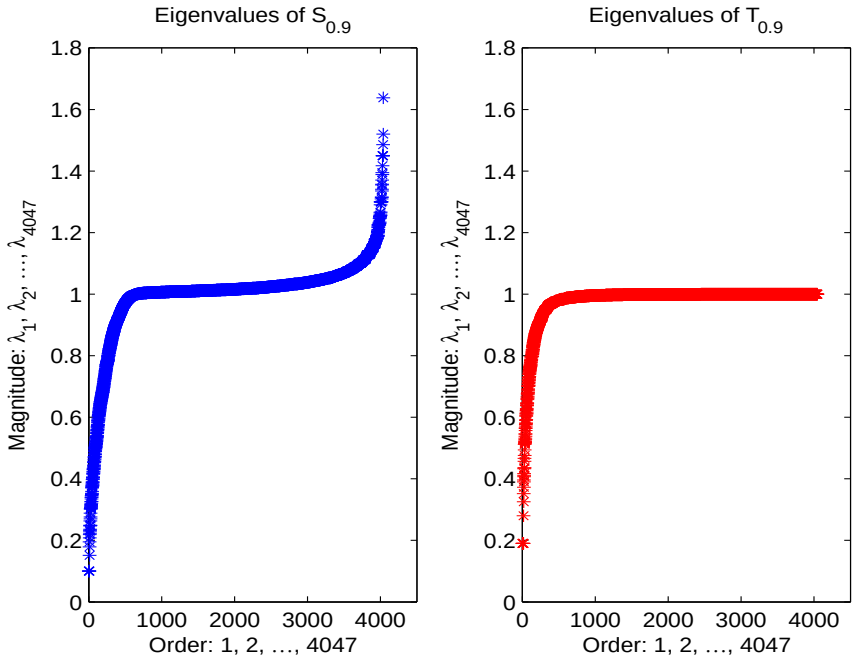

FIG. 4.3. *Eigenvalue distribution of $S_{0.9}$ and $T_{0.9}$ for the matrix w_All.*

TABLE 4.3
Results for the w_Down matrix in Example 4.1. Here ex = extr_dataDown.

α	0.50	0.75	0.80	0.99
CG	267(0.087)	310(0.091)	322(0.094)	427(0.131)
PCG	27(0.010)	40(0.013)	44(0.014)	76(0.024)
Chebyshev	22(0.007)	35(0.008)	40(0.009)	173(0.035)
Chebyshev-M_α	14(0.016)	24(0.012)	28(0.013)	125(0.041)
Chebyshev-$\bar{M}_\alpha$	11(0.010)	19(0.013)	22(0.014)	103(0.046)
CG-M_α	11(0.006)	17(0.006)	18(0.006)	32(0.013)
CG-$\bar{M}_\alpha$	10(0.004)	16(0.006)	17(0.008)	32(0.020)
CG-$\tilde{M}_\alpha$	7(0.007)	12(0.012)	13(0.011)	23(0.014)

TABLE 4.4
Results for the SNPa matrix in Example 4.2. Here ex = $(\frac{1}{n})e$, where e is the vector of all ones.

α	0.50	0.75	0.80	0.99
CG	86(2.420)	116(3.321)	128(3.663)	469(13.321)
PCG	17(0.514)	27(0.818)	30(0.922)	91(2.738)
Chebyshev	17(0.208)	28(0.359)	31(0.384)	130(1.402)
Chebyshev-M_α	11(0.244)	19(0.352)	22(0.382)	95(1.246)
Chebyshev-$\bar{M}_\alpha$	9(0.298)	15(0.346)	18(0.401)	79(1.464)
CG-M_α	9(0.164)	13(0.213)	15(0.236)	44(0.704)
CG-$\bar{M}_\alpha$	8(0.163)	14(0.287)	16(0.342)	52(1.086)
CG-$\tilde{M}_\alpha$	6(0.195)	9(0.269)	10(0.262)	33(0.963)

5. Conclusion. In this paper, a simple polynomial preconditioner has been implemented and tested for the GeneRank problem. Then some of its properties have been given. Finally some numerical experiments have been presented to show the effectiveness of the proposed preconditioner. As seen it does not need any CPU time to set up the preconditioner and the preconditioner is explicitly in hand. Our numerical results show that the proposed preconditioner is more effective than the ones presented in the literature.

Acknowledgements. We would like to thank Prof. Yimin Wei from Fudan University for providing us the SNPa data and Prof. Michele Benzi from Emory University to provide us the code of Chebyshv acceleration method. The authors are also grateful to the anonymous referee for the valuable comments and suggestions which substantially improved the quality of this paper.

REFERENCES

[1] O. Axelsson, A survey of preconditioned iterative methods for linear systems of algebraic equations, BIT 25 (1985) 166–187.
[2] O. Axelsson, Iterative solution methods, Cambridge University Press, Cambridge, 1996
[3] S. Agarwal and S. Sengupta, Ranking genes by relevance to a disease, Proc. LSS Comput. Syst. Bioinform. Conf. 8 (2009) 37-46.
[4] M. Benzi and V. Kuhlemann, Chebyshev acceleration of the GeneRank algorithm, Electronic Transactions on Numerical Analysis 40 (2013) 311-320.
[5] A. Berman and R.J. Plemmons, Nonnegative Matrices in the Mathematical Sciences, Academic Press, New York, 1979, SIAM, Philadelphia, PA, 1994.

TABLE 4.5
Results for the SNPa matrix in Example 4.2. Here $ex = p$, where p is a random probability vector.

α	0.50	0.75	0.80	0.99
CG	127(3.649)	174(5.011)	193(5.607)	721(20.411)
PCG	26(0.789)	41(1.242)	46(1.375)	139(4.203)
Chebyshev	17(0.211)	27(0.354)	30(0.352)	125(1.368)
Chebyshev-M_α	11(0.259)	19(0.382)	21(0.367)	92(1.222)
Chebyshev-$\bar{M}_\alpha$	9(0.267)	15(0.396)	17(0.414)	77(1.349)
CG-M_α	8(0.170)	13(0.209)	14(0.232)	43(0.713)
CG-$\bar{M}_\alpha$	8(0.175)	14(0.279)	16(0.355)	51(1.069)
CG-$\tilde{M}_\alpha$	6(0.171)	9(0.245)	10(0.261)	32(0.908)

TABLE 4.6
Results for the RENGA matrices in Example 4.2. Here $ex = (\frac{1}{n})e$, where e is the vector of all ones.

		$n = 100000$		
α	0.50	0.75	0.80	0.99
CG	43(0.789)	47(0.840)	48(0.863)	129(2.305)
PCG	14(0.262)	22(0.431)	24(0.459)	95(1.824)
Chebyshev	17(0.183)	27(0.245)	31(0.259)	127(0.905)
Chebyshev-M_α	11(0.244)	19(0.352)	22(0.382)	95(1.246)
Chebyshev-$\bar{M}_\alpha$	9(0.238)	15(0.331)	17(0.358)	78(1.083)
CG-M_α	8(0.092)	12(0.141)	14(0.168)	53(0.673)
CG-$\bar{M}_\alpha$	7(0.120)	11(0.192)	12(0.234)	51(0.885)
CG-$\tilde{M}_\alpha$	5(0.118)	9(0.201)	10(0.226)	41(0.935)
		$n = 500000$		
CG	23(2.612)	27(3.006)	30(3.398)	106(12.019)
PCG	13(1.578)	20(2.453)	22(2.661)	86(10.497)
Chebyshev	17(0.901)	27(1.371)	30(1.512)	125(6.359)
Chebyshev-M_α	11(1.655)	19(2.231)	21(2.364)	91(7.240)
Chebyshev-$\bar{M}_\alpha$	8(1.802)	15(2.482)	17(2.542)	76(8.082)
CG-M_α	5(0.645)	12(1.100)	14(1.234)	50(4.671)
CG-$\bar{M}_\alpha$	6(0.743)	10(1.223)	11(1.384)	44(5.555)
CG-$\tilde{M}_\alpha$	5(0.768)	8(1.214)	9(1.436)	38(6.098)

[6] G.H. Golub and C.F. van Loan, Matrix Computations, 3rd edition, Johns Hopkins University Press, Baltimore, 1996.

[7] M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, Journal of Research of the National Bureau of Standards 49 (1952) 409-436.

[8] J. Morrison, R. Breitling, D. Higham and D. Gilbert, GeneRank: using search engine for the analysis of microarray experiments, BMC Bioinform, 6 (2005) 233-246.

[9] Y. Saad, Iterative methods for sparse linear systems, 2nd Edition. SIAM, Philadelphia, 2003.

[10] G. Wu, Y. Zhang, and Y.Wei. Krylov subspace algorithms for computing GeneRank for the analysis of microarray data mining, Journal of Computational Biology 17 (2010) 631646.

[11] G. Wu, W. Xu, Y. Zhang, and Y. Wei, A preconditioned conjugate gradient algorithm for GeneRank with application to microarray data mining, Data Min. Knowl. Disc. 26 (2013) 27-56.

[12] B. Yue, H. Liang and F. Bai, Understanding the GeneRank model, IEEE 1st Int Conf Bioinform Biomed Eng. 6-8 (2007) 248251.

TABLE 4.7
Results for the RENGA matrices in Example 4.2. Here $ex = p$, where p is a random probability vector.

		$n = 100000$		
α	0.50	0.75	0.80	0.99
CG	68(1.254)	73(1.299)	75(1.337)	222(4.021)
PCG	22(0.418)	34(0.667)	39(0.732)	165(3.286)
Chebyshev	17(0.167)	26(0.255)	30(0.269)	122(0.883)
Chebyshev-M_α	11(0.257)	19(0.289)	22(0.335)	93(0.949)
Chebyshev-$\bar{M}_\alpha$	8(0.237)	15(0.306)	17(0.335)	75(0.998)
CG-M_α	8(0.093)	12(0.145)	14(0.170)	53(0.668)
CG-$\bar{M}_\alpha$	7(0.123)	11(0.185)	12(0.216)	51(0.903)
CG-$\tilde{M}_\alpha$	5(0.105)	9(0.185)	10(0.212)	40(0.877)
		$n = 500000$		
CG	36(4.101)	45(5.109)	50(5.637)	200(22.676)
PCG	21(2.509)	34(4.168)	38(4.585)	163(19.841)
Chebyshev	16(0.838)	26(1.360)	29(1.455)	120(6.062)
Chebyshev-M_α	11(1.702)	18(2.153)	21(2.409)	88(6.712)
Chebyshev-$\bar{M}_\alpha$	8(1.750)	14(2.464)	17(2.556)	73(7.948)
CG-M_α	8(0.733)	12(1.094)	13(1.174)	51(4.761)
CG-$\bar{M}_\alpha$	6(0.807)	10(1.244)	11(1.391)	46(5.854)
CG-$\tilde{M}_\alpha$	5(0.796)	8(0.236)	10(1.586)	39(6.050)