The relationship between the features of sparse matrix and the matrix solving status *

Dianwei Han †
Computer Science Dept.
University of Kentucky
Lexington, KY 40506
dianweih@csr.uky.edu

Shuting Xu ‡
Department of CIS
Virginia State University
Petersburg, VA 23806
sxu@vsu.edu

Jun Zhang §
Computer Science Dept.
University of Kentucky
Lexington, KY 40506
jzhang@cs.uky.edu

ABSTRACT

Solving very large sparse linear systems are often encountered in many scientific and engineering applications. Generally there are two classes of methods available to solve the sparse linear systems. The first class is the direct solution methods, represented by the Gauss elimination method. The second class is the iterative solution methods, of which the preconditioned Krylov subspace methods are considered to be the most effective ones currently available in this field. The sparsity structure and the numerical value distribution which are considered as features of the sparse matrices may have important effect on the iterative solution of linear systems. We first extract the matrix features, and then preconditioned iterative methods are used to the linear system. Our experiments show that a few features that may affect, positively or negatively, the solving status of a sparse matrix with the level-based preconditioners.

Categories and Subject Descriptors

G.1.3 [**Linear systems**]: iterative methods

General Terms

Algorithms

Keywords

ACM proceedings, iterative methods, preconditioner, features of matrices

*Technical Report CMIDA-HiPSCCS 001-08, Department of Computer Science, University of Kentucky, Lexington, KY, 2008.
This research was supported in part by the Kentucky Science and Engineering Foundation under grant KSEF-148-502-05-132.
†Graduate student
‡Assistant professor
§Full professor

1. INTRODUCTION

Many important scientific and engineering applications require the use of linear solvers for solving large-scale linear equations, such as the applications to develop the next generation electromagnetics simulators [5], to model and simulate biomedical processes, and to track nerve fibers in human brains [3]. The systems of linear equations from many of these applications can be written abstractly as:

$$A\,x = b \tag{1}$$

where A is a real-valued coefficient matrix of order n, x is the unknown vector and b is the given right-hand side vector. The matrix A will normally be large and sparse. There are two general schemes for solving linear systems: Direct Elimination Methods and Iterative Methods. The direct methods are, in some sense, based on the standard Gauss Elimination technique, which systematically applies row operations to transform the original system of equations into the product of a lower and upper triangular matrices. Among the iterative solution methods the Krylov subspace methods are considered to be the most effective ones currently available [7]. For large sparse linear systems, direct methods are usually avoided because of their potentially large memory requirements and overall efficiency concerns (CPU consumption).

Iterative methods are often considered as the only viable means of dealing with large sparse linear systems [7]. Various such methods have been proposed in the past 50 years [1]. However, the sheer size of the coefficient matrices encountered in practical applications and their ill-conditioning make current iterative solution methods unreliable and difficult to use.

It is widely recognized that preconditioning is very helpful for the iterative methods [11]. Preconditioning is a technique that transforms the original difficult linear system into an equivalent one that is easier to solve with an iterative solver. But choosing a good preconditioner for a specific sparse linear system is a challenging task.

Some researchers have recommended experimental approaches and some have shown that there are mathematical theorems to guarantee the convergence of certain preconditioned iterative methods for special matrices [2]. However, these strategies are not very useful for solving real practical problems because most preconditioning techniques rely on the structural and numerical features of the matrix. For example,

the ILU0, which is the ILU factorization techniques with no fill-in, rely on a fixed sparsity pattern. ILU0 is simple to implement, and is effective for some problems, such as finite difference discretization of PDE's.

We propose a completely new approach to help people efficiently use their preconditioned iterative solvers. First, we extract the special features of general sparse matrices. Second, we conduct experiments to apply preconditioned iterative solvers with these matrices. Finally, we identify the relationships between these matrix features and the performance of preconditioned iterative solvers.

The rest of this paper is organized as follows: In Section 2, we introduce matrix feature extraction. Section 3 discuss sparse linear systems. The experimental results are presented in Section 4, followed by some concluding remarks in Section 5.

2. MATRIX FEATURE EXTRACTION

The features of a matrix are computed according to its sparsity pattern, i.e., the locations and values of its nonzero entries. We extract 69 features in our paper. Here, we just list five major types: structure, value, bandwidth, diagonal, and other related statistics. For more detailed description on the matrix features, please see [9].

2.1 Structure

This group of features describe the distribution of nonzero elements of a matrix. For example, we consider the sparsity rate: the number of nonzero elements divided by the number of all elements of the matrix, the lower nonzero rate: the number of nonzero in the lower part divided by the number of all elements, the diagonal nonzero rate: the number of nonzero in the diagonal part divided by the number of all elements, and the upper nonzero rate: the number of nonzero in the upper part divided by the number of all elements. Sometimes we also want to know the total number of nonzero diagonals, i.e., the number of diagonals which have at least one nonzero element among the $2n-1$ diagonals of the matrix. The attribute *symmetric* measures whether a matrix is symmetric, i.e., $A = A^T$. The attribute *relsymm* describes the relative symmetric rate of a matrix. It is the ratio of the number of elements that matches divided by the number of nonzeros (nnz). An element $a(i,j)$ in the matrix A is *matched* if it satisfies the following condition: if $a(i,j)$ is nonzero then $a(j,i)$ is nonzero.

2.2 Value

This group of features show the value distribution of a matrix. We compute the average value of all nonzero entries and its standard deviation. We take the average of the main diagonal entries and the standard deviation, the average of the upper triangular entries and the standard deviation, as well as the average of the lower triangular entries and the standard deviation. Additionally, the matrix norms are very important attributes as well. For example, the one norm, the infinity norm, and the Frobenius norm of a matrix are computed. This group also includes the minimum of the sum of the columns, and the minimum of the sum of the rows.

2.3 Diagonal

All the features in this category are diagonal related. They include the percentage of weakly diagonally dominant columns and the percentage of weakly diagonally dominant rows. We compute the total number of non-void diagonals, the maximum and the minimum values of the diagonal elements. We include the average distance from each entry to the main diagonal and the standard deviation. Furthermore, we compute the average of the difference from each of the entry to its diagonal value and the standard deviation, etc.

2.4 Bandwidth

This group of features describe the bandwidth of a matrix. For example, the lower bandwidth of a matrix is defined as the largest value of $i-j$, where $a(i,j)$ is nonzero, the upper bandwidth of a matrix is defined as the largest value of $j-i$. The maximum bandwidth is defined as $\max(\max(j) - \min(j))$, where $a(i,j)$ is nonzero. The average bandwidth is defined as the average width of all columns.

3. SPARSE LINEAR SYSTEM

We will discuss some issues concerning the sparse linear systems. First of all, we will briefly introduce iterative methods that are currently most widely used. Then, we will discuss the preconditioners that we used in our research and present the solving status of the sparse linear systems that we used in our experiments.

3.1 Iterative Methods and Preconditioned Krylov Methods

Many iterative linear system solvers are published in literature and used in practice, such as Jacobi method, Gauss-Seidel method, Successive Overrelaxation method (SOR), Symmetric SOR, and Krylov Subspace methods [7]. Among them, the class of Krylov Subspace methods is the most popular and promising general purpose iterative methods. The solver that we used in our experiments is PGMRES (Preconditioned Generalized Minimum Residual Method), which is one of the widely used Krylov Subspace methods.

Krylov subspace methods are based on some projection processes, both orthogonal and oblique, onto Krylov subspaces [7]. A *Krylov subspace* is of the form

$$\mathcal{K}_m(A,v) = \text{span}\{v, Av, A^2v, \ldots, A^{m-1}v\}, \qquad (2)$$

for some nonzero vector v. *Projection method* is to seek an approximate solution x_m to the linear system (1) from an affine subspace $x^{(0)} + \mathcal{K}_m$ of dimension m, such that

$$b - Ax_m \perp \mathcal{L}_m, \qquad (3)$$

where $\mathcal{L}_m$ is another subspace of dimension m. In the implementation of a specific Krylov subspace method, we usually choose $v = r^{(0)}$, where $r^{(0)}$ is the residual of initial guess ($x^{(0)}$). The different Krylov methods use different $\mathcal{L}_m$.

Generalized Minimum Residual Method (GMRES) is a projection method based on the Krylov subspace. It constructs two subspaces $\mathcal{K}_m$ and $\mathcal{L}_m$, where $\mathcal{L}_m = A\mathcal{K}_m$. GMRES uses $v = r^{(0)}/\|r^{(0)}\|_2$. Applying GMRES to a preconditioned system can yield preconditioned GMRES. Three different preconditioning strategies in PGMRES are available: left, split, and right.

In our experiments, GMRES is applied to solve the right-preconditioned system:

$$AM^{-1}u = b, \quad \text{with} \quad u = Mx.$$

Thus, The right Krylov subspace is constructed by:

$$\text{span}\{r_0, AM^{-1}r_0, (AM^{-1})^2 r_0, \ldots, (AM^{-1})^{m-1} r_0\}. \quad (4)$$

3.2 Preconditioning techniques

A preconditioner is any form of modification of an original linear systems which makes it easier to solve by a given iterative method [7]. Consider a linear system,

$$M^{-1}A\,x = M^{-1}b \quad (5)$$

where M is a nonsingular matrix of the same order of the matrix A. The two linear systems of (1) and (5) are equivalent and they have the same solution. If M is close to A, and if $M^{-1}A$ has a smaller condition number than A, the resulting system (5) can be solved by an iterative solver with fewer steps to converge. System (5) is called a preconditioned system, and M is taken as the preconditioner.

There are a variety of preconditioners, such as Jacobi, Successive Overrelaxation, and Symmetric Successive Overrelaxation preconditioners, Incomplete LU Factorization, Approximate Inverse Preconditioners, block preconditioners, and so on [7]. The preconditioners used in this work are based on the incomplete LU factorization.

3.2.1 ILU0

Out of the class of ILU factorizations, the zero-fill incomplete factorization (ILU0) is used most frequently due to its simplicity [4]. The ILU0 has been defined in general terms as "any pair of matrices L (unit lower triangular) and U (upper triangular) so that the elements of $A - LU$ are zero in the location of $NZ(A)$" [7]. Consider one such matrix A as in Figure 1.

The A matrix represented in this figure is a 5-point matrix. Lower triangular matrix L has the same structure as the lower part of A, and the matrix U has the same structure as the upper part of the matrix A.

ILU0 is relatively simple to implement and it has wide applications in many areas, such as finite difference discretization of PDEs. However, the no-fill factorization method is not a very good one in that it can result in too crude an approximation to the original matrix for some real and complicated problems. In this case, the more sophisticated preconditioners, which allow some fill-in, are needed [10].

3.2.2 ILUK

ILUK is the ILU factorization with *level of fill* to be k while ILU0 only allows fill-in in the nonzero position of A. ILU1 allows one more line of fill-in in each of the L and U factors. ILU1 factorization can be illustrated by Figure 2. In this case, the matrix A can be viewed as "augmented pattern"

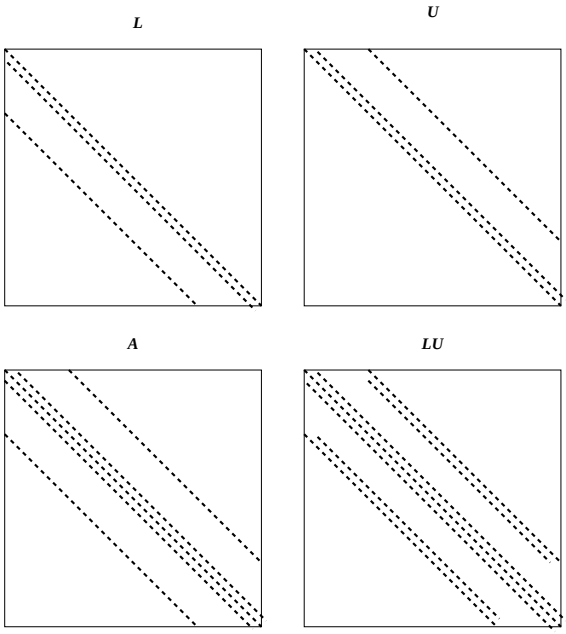

Figure 1: The ILU0 factorization for a five-point matrix.

matrix because the fill-in of blue squares are imaginary and their values are zero. The patterns of L and U are illustrated at the top of Figure 2. The new LU matrix has two additional diagonals, compared to the ILU0 case.

ILU1 is only applicable to structured matrices. The concept of level of fill is introduced for general sparse matrices. A level of fill is attributed to each element that is processed by Gaussian elimination and dropping is based on the value of the level of fill [7]. ϵ^k is attributed to each element whose level of fill is k, where $\epsilon < 1$. It is shown that: the higher the level, the smaller the element. The initial definition is

$$lv_{ij} = \begin{cases} 0 & \text{if } a_{ij} \neq 0, \text{ or } i = j, \\ \infty & \text{otherwise.} \end{cases}$$

In Gaussian elimination, each element a_{ij} is updated by the formula

$$a_{ij} = a_{ij} - a_{ik} \times a_{kj}. \quad (6)$$

The new level of fill with update is obtained by:

$$lv_{ij} = \min\{lv_{ij}, lv_{ik} + lv_{kj} + 1\}. \quad (7)$$

The ILUK algorithm is depicted in Algorithm 3.2.2 [7]. Any element whose level of fill is greater than k is dropped. In our experiments, we only consider $k = 1$, 2, and 3 respectively.

We choose ILUK because they are relatively simple to implement, and they do not have many free parameters used in the building of preconditioners. We will apply more sophisticated preconditioners in our later work.

The ILUK algorithm.

1. For all nonzero elements a_{ij}, define $lev(a_{ij}) = 0$
2. For $i = 2, \cdots, n$, Do:

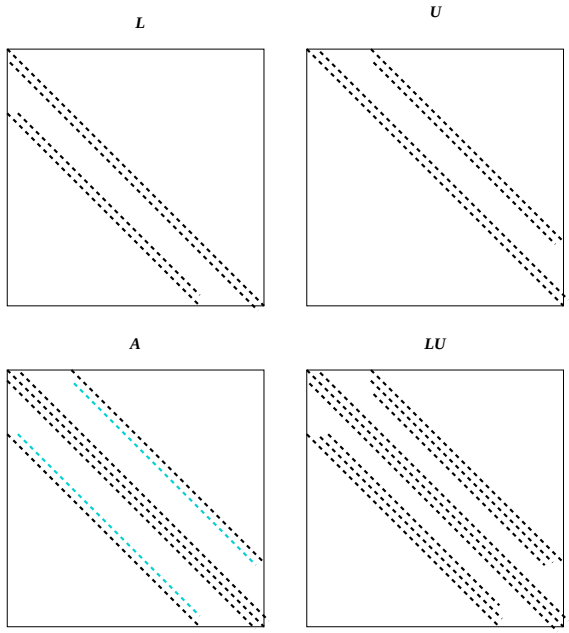

Figure 2: The ILU1 factorization for a five-point matrix.

3. For each $k = 1, \cdots, i-1$ and for $lev(a_{ik}) \leq k$,
4. Do:
5. Compute $a_{ik} := a_{ik}/a_{kk}$
6. Compute $a_{i*} := a_{i*} - a_{ik}a_{k*}$
7. Update the levels of fill of the nonzero $a_{i,j}$'s
8. using (7)
9. EndDo
10. Replace any element in row i with $lev(a_{ij}) > k$
11. with zero
11. EndDo

3.3 Solving status of sparse linear systems

Here we say that a sparse linear system is solved if the preconditioner can be successfully constructed, the preconditioned iterative solver converges within a preset number of iterations, and the relative residual norm is smaller than a preset value. Most general purpose preconditioners for the sparse linear systems are derived from incomplete LU factorization. We just choose ILUK as preconditioners in our experiments. The solvability of a matrix by the preconditioned iterative solver depends mostly on whether we can successfully construct a preconditioner. If a preconditioner can be successfully constructed with a moderate *condest* (condition number estimate), which is defined as $\| (LU)^{-1}e \|$, where e is the vector of all ones, we will be able to continue to execute the preconditioned iterative solver.

We divide the solving statuses into two categories: (1) The preconditioner is NOT successfully constructed. (2) The preconditioner is successfully constructed. When solving the 319 sparse linear systems generated from the 319 sparse matrices, there are four kinds of returning status under the first category. If $solvstat = 95$, it means that the program fails because of zero row. If the solving status is greater than or equal to 100, a zero pivot is encountered in constructing the preconditioner. If $solvstat = 92$, it means that the condest value is very large (larger than 10^{15}) and the preconditioner is not stable. If $solvstat = 93$, it means that the small pivot is too small. The condest value measures the stability of the triangular solvers. Under these four cases, the preconditioner could not be constructed. We only consider two cases under the second category. When $solvstat =$ -1, it means that the preconditioned solver fails to converge within a preset number of iterations (the limit is set to be 500). When $solvstat = 0$, it means that the preconditioned solver (PGMRES in our experiments) successfully solves the problem (sparse linear system).

SS	Meaning
>=100	Failed in constructing preconditioner : zero pivot
95	Failed in constructing preconditioner : zero row
93	Failed in constructing preconditioner : small pivot is too small
92	Failed in constructing preconditioner : large condest, unstable preconditioner
−1	Failed in GMRES: cannot converge in 500 iterations
0	Successful in GMRES

Figure 3: The solving status and the meaning.

4. EXPERIMENTS AND RESULTS

We performed some experiments to test the relationship between the features of sparse matrices and the solving status of preconditioned linear system. The linear systems are constructed by using 319 sparse matrices from the Matrix Market [6]. The right-hand sides (vector b) are constructed by assuming that the solutions are a vector of all ones. The initial guess to the linear system is a vector of all zeros. The maximum number of iterations allowed is set to 500. The stopping criterion is that the 2-norm residual of the approximate solution is reduced by 7 orders of magnitude, relative to the 2-norm residual of the initial guess. The iterative solver used is GMRES and the preconditioners are matrix structure-based incomplete LU factorizations ILUK [10]. We choose K to be 1, 2, 3 respectively and denote them by ILUK1, ILUK2, and ILUK3. 69 matrix attributes has been used in our experiments. The experiments are conducted on a SunBlade 100 workstation.

4.1 Solving status of sparse linear systems

Figure 3 shows the possible solving status of running the constructed preconditioned linear systems and their meanings. SS denotes solving status. If the preconditioner construction process encounters any problem, such as zero pivoting row or large condest value, it will stop and return a value to indicate the error.

The solving statuses related to ILUK1, ILUK2, and ILUK3 are listed in Figure 4. Compared with ILUK1 case, the number of successfully solved linear systems (Solving Status = 0) in ILUK2 and ILUK3 increases a little bit. By using ILUK1

Solving Status	0	-1	95	93	92	Total
ILUK1						
Number of Matrices	193	41	74	2	9	319
ILUK2						
Number of Matrices	195	36	69	3	15	319
ILUK3						
Number of Matrices	199	33	69	2	15	319

Figure 4: The solving status related to ILUK.

	Ill			Good			
features	mean	deviation	median	mean	deviation	median	ratio
diagsparert	0.3547	0.3388	0.0659	0.9141	0.1150	1	39%
nnzdiagrt	0.0572	0.0643	0.0097	0.1486	0.1278	0.0737	34%
diagfillrt	0.3788	0.4101	0.0838	0.937	0.1145	1	40%
relsymm	0.3931	0.4089	0.2267	0.9255	0.1641	1	42%

Figure 5: The significant features that influence solving status in the structure category.

as preconditioner, 41 linear systems fail to converge within 500 iterations when running GMRES algorithm. However, we can see that only 33 GMRES solvers preconditioned with ILUK3 fail to converge within the 500 iterations. The improvement is significant (20%). Furthermore, ILUK2 and LIUK3 work better in the third category (Solving Status = 95), compared with the results of ILUK1 as well. On the other hand, more failures are encountered due to the instability of preconditioners during the construction of preconditioners for ILUK2 and ILUK3 than for ILUK1.

4.2 Significant features

Some features of the sparse matrix have been found to significantly affect the solving status of sparse linear systems. Among the 69 attributes, 16 features play an important role in solving a sparse linear system positively or negatively, which are distributed into four different categories.

4.2.1 Structure

Four features are from this category. We find that almost all the matrices on which no preconditioner could be constructed due to zero row have very small diagonal sparse rate, none zero diagonal rate, diagonal fill rate, and relative symmetry value. So, all these four features are positive in the construction of preconditioner. The detailed information is given in Figure 5.

In Figure 5, (Ill) denotes the bad case that has zero row issue in the construction of preconditioner, (Good) denotes the good case that does not have this problem, the field of "deviation" stands for "average deviation", and the field of "ratio" denotes the ratio of mean(Ill) to mean(Good). If the value of ratio is relatively small, such as 42%($<$ 60%), the

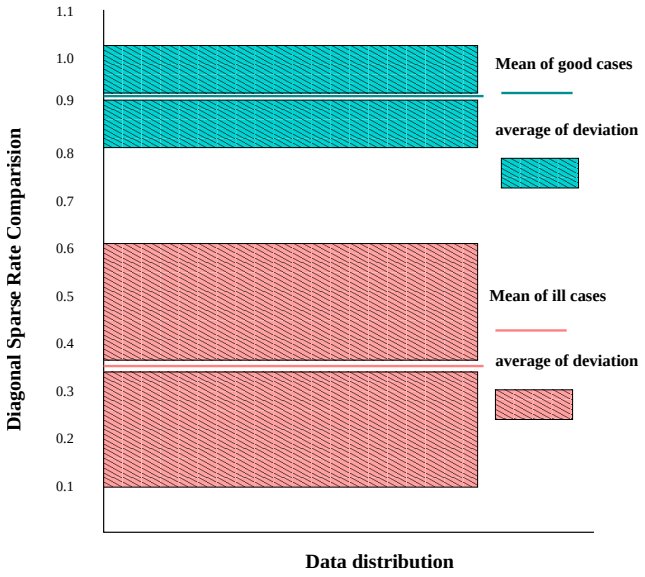

Figure 6: Data area of diagonal sparse rate.

feature's role is positive. On the other hand, if the value of ratio is relatively large, such as 152%($>$ 140%), it shows that the feature's role is negative. Figure 6 displays the data area of ill cases and good cases for diagonal sparse rate feature.

In Figure 6, we can see that the data area of good cases is above the region of ill cases. Here we explain why these features are important. Three features are related to nnzdiag feature (none zero diagonals). If there are very few nonzero diagonals it is more likely to have a zero row. That is the main reason. Furthermore, the statistical data tells us another fact: if a matrix is less symmetric it is more possible that it has a zero row. This conclusion can be obtained from feature relsymm: the mean of ill cases only take 42% of the mean of good cases. In short, all the four features are positive features. In other words, this consideration suggests that large diagonal sparse rate, none zero diagonal rate, diagonal fill rate, and relative symmetry value are desirable.

4.2.2 Diagonal

The Figure 7 shows the four features that play an important role in the diagonal category. The feature diagdomcol denotes the percentage of weakly diagonally dominant columns. The feature diagdomrow denotes the percentage of weakly diagonally dominant rows. The large values of these two features are definitely desirable. A diagonally dominant matrix is unlikely to have a zero row. The feature avvalfdrt is related to the average difference between $a(i,j)$ and diagonal. If the difference is very small, it means that the diagonals are not dominant. The feature mindiagrt denotes the ratio mindiag to frnorm and mindiag denotes the minimum value of the diagonal element. If the value of mindiag is large the matrix is well-conditioned.

4.2.3 Value

Six features that are important in solving a linear system belong to the Value category. The detailed statistical data is displayed in Figure 8. The feature avdiagrt is related to the average value of the main diagonal entries. If the

	Ill			Good			
features	**mean**	**deviation**	**median**	**mean**	**deviation**	**median**	**ratio**
diagdomcol	0.1023	0.2284	0.0028	0.38	0.342	0.2928	27%
diagdomrow	0.1179	0.2231	0.0040	0.3537	0.3528	0.1826	33%
avvalfdrt	0.0066	0.0088	0.0044	0.0122	0.0132	0.0092	54%
mindiagrt	0.0016	0.0058	0	0.0050	0.0098	0.0002	33%

Figure 7: The significant features that influence the solving status in diagonal category.

	Ill			Good			
features	**mean**	**deviation**	**median**	**mean**	**deviation**	**median**	**ratio**
avdiagrt	0.0054	0.0111	0.0004	0.0199	0.0218	0.0135	27%
sdavdiagrt	0.0054	0.0128	0.7e−5	0.0188	0.0201	0.0132	28%
pdiag	0.0569	0.1502	0.0002	0.4237	0.224	0.3956	13%
pup	0.503	0.2694	0.4802	0.2874	0.1462	0.2895	175%
plow	0.4402	0.2651	0.4681	0.2889	0.1613	0.3043	152%
avnnzval	4.3e+11	3.5e+12	5.18	1.1e+14	1.6e+15	1.825	0.40%

Figure 8: The significant features that influence the solving status in value category.

mean of main diagonal entries is small the matrix is not well-conditioned. The feature pdiag denotes the percentage of diagonal value. In reality, we expect to have a large pdiag value. However, the features pup and plow are not desirable because they could make main diagonals insignificant. The feature avnnzval is referred to as the average value of all nonzero entries of the matrix. The small avnnzval would imply that the matrix is more likely to be ill-conditioned.

4.2.4 Others

Two features belong to other category. The number of structural zero pivots [8] is strzpiv. This feature plays a role in solving a linear system negatively. The smaller strzpiv is what we expect. Another feature is minvalcol. The large minvalcol is good.

5. CONCLUDING REMARKS

The experimental results show that some features of sparse matrices may affect the solving status of the sparse matrices with the level-based preconditioners. Thirteen features are positive and three features are negative. This information is very important for people to choose good preconditioners or to solve sparse linear systems. The 16 features are distributed into four different categories. This implementation and the reported experimental results are the part of our work on "Intelligence Preconditioner Recommendation System (IPRS)", which can provide advice on choosing a high performance preconditioner as well as suitable parameters for a given sparse linear system. We will study and define some more features related to diagonal in order to give insight into why these features are so sensitive in solving the linear systems.

6. REFERENCES

[1] G. H. Golub, H. A. van der Vorst. *Closer to the solution: iterative linear solvers.* In I. S. Duff and G. A. Watson, editors, The State of the Art in Numerical Analysis, pp. 63-92, Oxford, 1997.

[2] A. Greenbaum. *Iterative Methods for Solving Linear Systems.* SIAM, Philadelphia, PA, 1997.

[3] N. Kang, J. Zhang, and E. S. Carlson. *Parallel simulation of anisotropic diffusion with human brain DT-MRI data.* Computers and Structures, Vol. 82, pp. 2389-2399 (2004)

[4] G. Karypis, V. Kumar. *Parallel threshold-based ILU factorization.* Supercomputing, ACM/IEEE 1997 Conference , (CDROM). pp. 1-24 (1997)

[5] J. Lee, J. Zhang, and C.-C. Lu. *Performance of preconditioned Krylov iterative methods for solving hybrid integral equations in electromagnetics.* J. of Applied Comput. Electromagnetics Society, Vol. 18, pp. 54-61 (2003)

[6] http://math.nist.gov/MatrixMarket/

[7] Y. Saad. *Iterative Methods for Sparse Linear Systems*, PWS, New York, 1996.

[8] http://www-users.cs.umn.edu/šaad/software/SPARSKIT/sparskit.html

[9] S. Xu, J. Zhang. *A new data mining approach to predicting matrix condition numbers.* Communications in Information and Systems, Vol. 4, pp. 325-340 (2004).

[10] S. Xu, J. Zhang. *A data mining approach to matrix preconditioning problem.* In Proceedings of the Eighth Workshop on Mining Scientific and Engineering Datasets (MSD'05), pp. 49-57 (2005)

[11] Y. Wang, J. Lee, and J. Zhang. *A short survey on preconditioning techniques for large scale dense complex linear systems in electromagnetics.* International Journal of Computer Mathematics, Vol. 84, pp. 1211-1223 (2007)