Progress In Electromagnetics Research, PIER 90, 121–136, 2009

A GENERALIZED GPS ALGORITHM FOR REDUCING THE BANDWIDTH AND PROFILE OF A SPARSE MATRIX

Q. Wang, Y. C. Guo, and X. W. Shi

National Key Laboratory of Antenna and Microwave Technology
Xidian University
No. 2 South Taibai Road, Xi'an, Shaanxi, China

Abstract—A generalized GPS (GGPS) algorithm is proposed for the problem of reducing the bandwidth and profile of the stiffness matrix in finite element problems. The algorithm has two key-points. Firstly and most importantly, more pseudo-peripheral nodes are found, used as the origins for generating more level structures, rather than only two level structures in the GPS (Gibbs-Poole-Stockmeyer) algorithm. A new level structure is constructed with all the level structures rooted at the pseudo-peripheral nodes, leading to a smaller level width than the level width of any level structure's in general. Secondly, renumbering by degree is changed to be sum of the adjacent nodes codes to make a better renumbering in each level. Simulation results show that the GGPS algorithm can reduce the bandwidth by about 37.63% and 8.91% and the profiles by 0.17% and 2.29% in average for solid models and plane models, respectively, compared with the outcomes of GPS algorithm. The execution time is close to the GPS algorithm. Empirical results show that the GGPS is superior to the GPS in reducing bandwidth and profile.

1. INTRODUCTION

In recent years the greatest progress in computational electromagnetics has been in the field of partial differential equation methods such as the finite-element method (FEM) [1–6]. Nowadays FEM is one of the most popular numerical methods for solving partial differential equations [7, 8]. The FEM involves the solution of a large sparse system

Corresponding author: Q. Wang (wangqing@mail.xidian.edu.cn).

of linear equations, which is of the form [6]

$$Ax = B \tag{1}$$

where the $n \times n$ matrix A is the coefficient matrix, called the stiffness matrix. B is right-hand side vector, and x is the unknown vector. The matrix A is generally an extremely sparse positive definite matrix. There is a direct correspondence between the structure of the coefficient matrix A and the structure of the topology graph delineating the matrix element layout. Both direct and iterative methods can be used for solving such a system.

Direct methods produce an exact solution using a finite number of operations, however, direct methods require large memory resources and consume a large solution time. For the efficient solution of the large sparse system of linear equations in Equation (1) and to compress the memory space, it is desirable to have a nodal renumbering scheme to ensure that the corresponding coefficient matrix A will have a narrow bandwidth and profile, called the matrix bandwidth minimization problem. The renumbering operation is applied before the construction of the stiffness matrix A, aimed at finding a permutation of the rows and the columns of a matrix that keeps all the non-zero elements in a band, or, as close as possible to the main diagonal. This problem has generated considerable interest over at least 32 years because of its significance in the solution of sparse matrix.

A variety of methods has been proposed for the matrix bandwidth minimization problem. The first extensive study of bandwidth minimization problems was done by Cuthill and McKee in 1969 as the Cuthill-McKee (CM) algorithm [18]. Following the publication of the CM algorithm, graph theory and level sets became standard approaches for both bandwidth and profile reduction [5]. Methods proposed in the late 1970s and early 1980s include the Reverse Cuthill-McKee algorithm [13], the Gibbs-King algorithm [17], and the Gibbs-Poole-Stockmeyer algorithm [15]. Gibbs, Poole and Stockmeyer used the concept of maximum eccentricity of graphs to develop a method that finds a good starting node. The performances of this method were superior to the CM algorithm, and its computation time became much faster than the CM algorithm [14].

The GPS uses a heuristic starting nodes finding algorithm to get a pair of nodes located at nearly maximal distance apart [16, 17], called pseudo-peripheral nodes. However, in some problems of finite element analysis, namely, in graph associated with some more complex shapes there may be several candidates for pseudo-peripheral nodes. The selection of the pair of nodes may make a difference. Consequently, if additional criteria were taken into account to enlarge the amount of the eligible nodes, more pseudo-peripheral nodes would be found.

In this paper we introduce a generalized modification to the GPS algorithm, aiming at the improvement of its extension in the selection of pseudo-peripheral nodes. It modifies the GPS pseudo-peripheral nodes finding algorithm to get more pseudo-peripheral nodes. In the third algorithm of GPS method, the rule for renumbering is also modified. Renumbered by degree seems to be instable, so we change the rule to be by the numbers of the sum of the adjacent nodes codes.

Section 2 introduces the necessary terminology and background. Section 3 reviews the GPS algorithm and gives the procedure of the new GGPS approach. Section 4 compares the simulation results and performances of GGPS algorithm against GPS algorithm with some classical models. Section 5 draws up our conclusions.

2. BASIC CONCEPTS AND TERMINOLOGIES

2.1. Matrix Terminologies

Let the coefficient matrix A in Equation (1) be an n by n symmetric positive definite matrix, with entries a_{ij}

$$\beta_i(A) = i - \min\left\{j \middle| a_{ij} \neq 0\right\} \tag{2}$$

where $\beta_i(A)$ is called the ith bandwidth of A.

The bandwidth of A is defined by [18]

$$\beta = \beta(A) = \max\left\{\beta_i(A) \middle| 1 \leq i \leq n\right\} = \max\left\{|i-j| |a_{ij} \neq 0\right\} \tag{3}$$

For the Cholesky resolution, it has been shown that, if $\beta \ll n$, the number of operations N_{op} required is a function of the square of β [2]

$$N_{\text{op}} = O\left(n\beta^2\right) \tag{4}$$

Hence, reducing β is an important factor to decrease the resolution time.

The vector that contains all the bandwidth lines is called the envelope of A and defined by

$$Env(A) = \left\{(i,j) \middle| 0 < i-j \leq \beta_i(A), i = 1, \ldots, n\right\} \tag{5}$$

The quantity $|Env(A)|$ is called the profile or the envelope size of A, defined by

$$P(A) = |Env(A)| = \sum_{i=1}^{n} \beta_i(A) \tag{6}$$

where $P(A)$ represents the profile of A.

By minimizing the profile, we minimize the number of stored zero values in the stiffness matrix, which can reduce the processor constraints and accelerate the resolution process.

2.2. Graph Theory

If V is a finite nonempty set and $E \in \{\{(a,b)\} : a \neq b \,\text{and}\, a,b \in V\}$ is a collection of unordered pairs of elements of V, $G = (V,E)$ is a finite undirect graph without loops or multiple edges. For the stiffness matrix $A = (a_{ij})_{n\times n}$ in Equation (1), we can define a graph $G = (V,E)$ where V has n nodes, $\{v_1, v_2, \ldots, v_n\}$, and $\{v_i, v_j\} \in E$ if $a_{ij} \neq 0$ and $i \neq j$. The elements of $V(G)$ and $E(G)$ are called nodes and edges, respectively.

An important concept in bandwidth and profile reduction algorithms is level structure [15]. A level structure of graph G, $L(G)$, is a partition of the elements in $V(G)$ into levels $L_1, L_2, \ldots, L_k$. The depth of level structure $L(G)$ is also k, the number of levels. The essential properties of $L(G)$ are that all nodes adjacent to nodes in L_1 are in either L_1 or L_2; all nodes adjacent to nodes in L_k are in either level L_k or L_{k-1}; for $1 < i < k$, all nodes adjacent to nodes in L_i are in either L_{i-1}, L_i or L_{i+1}. To each node $v \in V(G)$ there corresponds a particular level structure $L_v(G)$, called the level structure rooted at v. In $L_v(G)$, $L_1 = \{v\}$. In any level structure $L(G)$, rooted or not, $w_i(L) = |L_i|$ (the number of the nodes in L_i) is called the width of L_i, and $w(L) = \max\{(w_i)_{i=1,\ldots,k}\}$ is the width of the level structure $L(G)$. It is easily observed that for any level structure, $L(G)$, a numbering f_L of G that assigns consecutive integers level by level, from the nodes in L_1, then to those in L_k, yields a bandwidth, β_{f_L}, satisfying [15]

$$\beta_{f_L} \leq 2w(L) - 1 \tag{7}$$

If the level structure $L(G)$ is rooted, then we also have

$$\beta_{f_L} \geq w(L) \tag{8}$$

By renumbering the nodes of corresponding graph G, we can change the structure of A to reduce both bandwidth and profile.

3. DESCRIPTION OF THE GGPS ALGORITHM

3.1. Review of the GPS Algorithm

The main steps of GPS process are [14]:

(i) Finding two pseudo-peripheral nodes of graph G;

(ii) Minimizing the level width;

(iii) Renumbering the nodes level by level.

3.2. The GGPS Algorithm

The description of the proposed GGPS algorithm is divided to three algorithms the same as the GPS algorithm, which are algorithm I, II and III as follow.

3.2.1. Algorithm I. Finding Pseudo-peripheral Nodes

(i) Pick an arbitrary node of minimal degree and call it v.

(ii) Generate a level structure L_v rooted at v. Let S be the set of nodes which are in the last level of L_v.

(iii) Generate level structures rooted at nodes $s \in S$ selected in the order of increasing degree. If for some $s \in S$ the depth of L_s is greater than the depth of L_v, then set $v \leftarrow s$ and return to step 2.

(iv) Let u be the node of S whose associated level structure has the smallest width, with ties broken arbitrarily. Let k be the depth of L_v or L_u.

(v) If for some $s \in S$ the depth of L_s is k, the nodes are picked up with u as the set of pseudo-peripheral nodes at the "u" end.

(vi) For any node $s' \in G$ with the same degree as v, if the depth of $L_{s'}$ is also k, and it has not been picked up yet, then picked it up to the set of pseudo-peripheral nodes at the "v" end.

(vii) Let m be the sum of pseudo-peripheral nodes at the "u" end and "v" end. Let p be the number of the pseudo-peripheral nodes at the "v" end, and q is the number of the pseudo-peripheral nodes at the "u" end, so $p + q = m$

Step (v), (vi) and (vii) are the modifications introduced. The set of pseudo-peripheral nodes at the "u" end are the nodes in the last level of L_v, which all have the same level depth of each rooted level structure, including node "u". The set of pseudo-peripheral nodes at the "v" are the nodes with the same degrees as v and also the same level depth compared with the level structure rooted from v. $L_{v_1}, \ldots, L_{v_{m_1}}, \ldots, L_{v_m}$ are defined as all the level structures constructed in algorithm I.

3.2.2. Algorithm II. Minimizing Level Width

From Equation (7), we know whether the level structure $L(G)$ is rooted or not, the bandwidth of A, β, and the level width of $L(G)$, $w(L)$, satisfy the relationship

$$\beta \leq 2w(L) - 1 \tag{9}$$

Equation (9) shows that smaller level width leads to a smaller matrix bandwidth and profile. The new level structure is a combination of all the level structures constructed in algorithm I, and more level structures tends to increase the possibility to make a new level structure with a smaller level width compared with the level width constructed in GPS method.

(i) Using the rooted level structures $L_{v_1} = \{L_1^1, L_2^1, \ldots, L_k^1\}$, $L_{v_2} = \{L_1^2, L_2^2, \ldots, L_k^2\}$, …, and $L_{v_m} = \{L_1^m, L_2^m, \ldots, L_k^m\}$ obtained from algorithm I, we associate each node w of G with the ordered set $(i_1, \ldots, i_p, i_{p+1}, \ldots, i_m)$ (p is the number of the nodes at the "v" end), called the associated level set, where $i_t (t = 1, \ldots, p)$ is the index of the level in level structure L_{v_t} that contains w, and $k + 1 - i_t$ $(t = p + 1, \ldots, m)$ is the index of the level in level structure L_{v_t} that contains w. Assign the nodes of G to levels in a new level structure $L = \{L_1, L_2, \ldots, L_k\}$ as follows:

(a) If the associated level set of a node w is of the form $(i, i, \ldots, i)$, w is placed in L_i of level structure L. Node w and all edges incident to w are removed from the graph. If $V(G) = \emptyset$ ($V(G)$ is the number of nodes in G), stop.

(b) In the above step, we have removed some nodes and the entire edges incident from G, so it is possible that the whole graph has been divided to be separate parts. We define those separate parts as "subgraphs", such as C_1, C_2, …, C_t. Graph G now consists of a set of more than one subgraph $C_1, C_2, \ldots, C_t (t \geq 2)$ ordered so that $|V(C_1)| \geq |V(C_2)| \geq \cdots \geq |V(C_t)|$.

(c) For the nodes in each subgraph $C_x (x = 1, 2, \ldots t)$, do the following:

1. Compute the vector $(n_1, n_2, \cdots n_k)$ where $n_i = |L_i|$ (n_i is the number of nodes which have been placed in L_i).
2. Compute the vectors $(h_1^1, h_2^1, \ldots, h_k^1)$, $(h_1^2, h_2^2, \ldots, h_k^2)$, …, and $(h_1^m, h_2^m, \cdots, h_k^m)$ where $h_i^j = n_i +$ (the number of nodes which would be placed in L_j if the jth element of the associated level set were used).
3. Find $h_0^j = \max_i \{h_i^j : h_i^j - n_i > 0\} (j = 1, \ldots, m)$, then $h_0 = \min\{h_0^1, \ldots, h_0^m\}$ and the corresponding index j $(j = 1, \ldots, m)$ (if there are more than one indices, pick up the former one), and place all the nodes of the subgraph in the levels indicated by the jth elements of the associated level set.

3.2.3. Algorithm III. Renumbering

Unlike GPS algorithm renumbering the nodes by the order of increasing degree, in GGPS the nodes are renumbered in the order of increasing sum of the adjacent nodes codes (in the following "sum" for short).

(i) If the sum of v is not the smallest among all the pseudo-peripheral nodes, interchange v and the node with the smallest sum, and reverse the level structure obtained in algorithm II by setting L_i to L_{k-i+1}.

(ii) Assign consecutive positive integers to the nodes in L_1 in the following order:
  - (a) Assign the number 1 to node v (if this is not the first component of the original graph, then assign the smallest unassigned positive integer to v).
  - (b) Let w be the lowest numbered node in L_1 which has unnumbered nodes in L_1 adjacent to it. Number the nodes in L_1 adjacent to w, in the order of increasing sum. Repeat this step until all nodes in L_1 adjacent to numbered nodes are themselves numbered.
  - (c) If any unnumbered nodes remain in L_1, number the one of minimal sum, then go to step (ii). Otherwise proceed to step (iii).

(iii) Number the nodes of level L_i $(i = 2, 3, \ldots, k)$, as follows:
  - (a) Let w be the lowest numbered node in L_{i-1} that has unnumbered nodes in L_i adjacent to it. Number the nodes in L_i adjacent to w in the order of increasing sum. Repeat this step until all nodes in L_i adjacent to nodes in L_{i-1} are numbered.
  - (b) Repeat steps (ii)(b) and (ii)(c), replacing 1 with i.

(iv) The numbering is reversed by setting i to $n-i+1$, for $i = 1, 2, \ldots n$ if either of the two following conditions holds:
  - (a) Step (i) interchanged v and the node with the smallest sum and algorithm II selected the "u" end element (the element before the pth element) of the level set for subgraph C_1.
  - (b) Step (i) did not interchange v and other node and algorithm II selected the "v" end elements of the level set for subgraph C_1.

3.2.4. An Example

This example explains the procedure of GGPS algorithm. Fig. 1 is the example with 58 nodes. Fig. 1(a) is the mesh and Fig. 1(b) is its

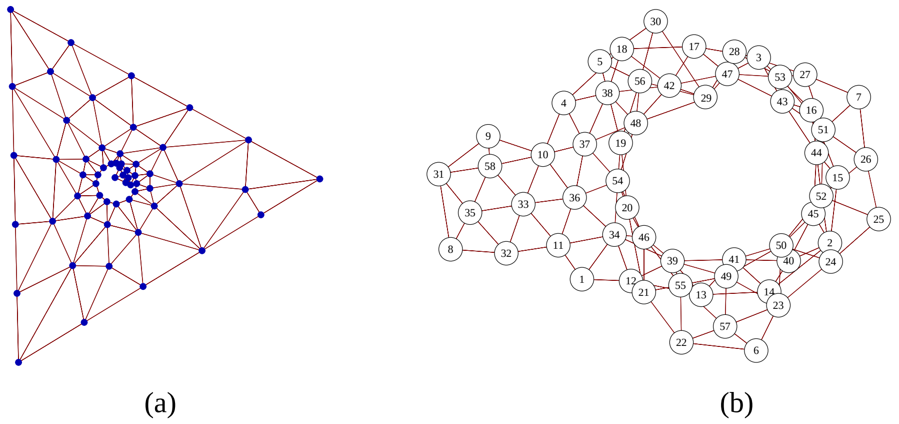

Figure 1. Mesh and graph with 58 nodes (a) Mesh (b) Graph.

corresponding topology graph. Its initial bandwidth is 51, and initial profile is 1134.

From Fig. 1(b), we first find two pseudo-peripheral nodes 7 and 8, which are also the two pseudo-peripheral nodes in GPS method. Their rooted level structures are shown in Fig. 2(a) and Fig. 2(b). Call node 7 as the "v" node and node 8 as the "u" node. In the last level of $L_7(G)$, there is only one node, node 8, so the pseudo-peripheral nodes at the "u" end only include node 8. In the graph with the rest nodes, we can still find nodes 1, 5, 6, 9 and 25 with the same degree as node 7, and the level depths of $L_1(G)$, $L_5(G)$, $L_6(G)$, $L_9(G)$ and $L_{25}(G)$ are 8, 8, 9, 9 and 10, respectively. Only $L_{25}(G)$ has the same level depth as the level depth of $L_7(G)$. So the pseudo-peripheral nodes at the "v" end include node 7 and 25. The associated level set in GGPS procedure is (i_1, i_2, i_3); i_1 and i_2 is the index of the level in $L_7(G)$ and $L_{25}(G)$ that contains w, and $k + 1 - i_3$ is the index of the level in $L_8(G)$ that contains w. For example, the associated level set for node 7 is (1,3,1), which means node 7 is in L_1 of $L_7(G)$, L_3 of $L_{25}(G)$, and L_{9+1-1}, or L_9, of $L_8(G)$, for node 25 is (3,1,2) and for node 8 is (10,9,10). The associated level sets in our method and the associated level pairs in GPS method for all nodes in the example are given in Table 1.

The nodes with the associated level set of the form (i, i, i) are node 26, 33, 35 and 36. The graph can not be divided to separate parts without these nodes, while the level depths of $L_7(G)$, $L_{25}(G)$ and $L_8(G)$ are 10, 10 and 9, so we compare the level widths of $L_7(G)$ and $L_{25}(G)$ to pick the one with the smallest level width. The level width of $L_{25}(G)$ is 11, and the level width of $L_7(G)$ is 12. So we

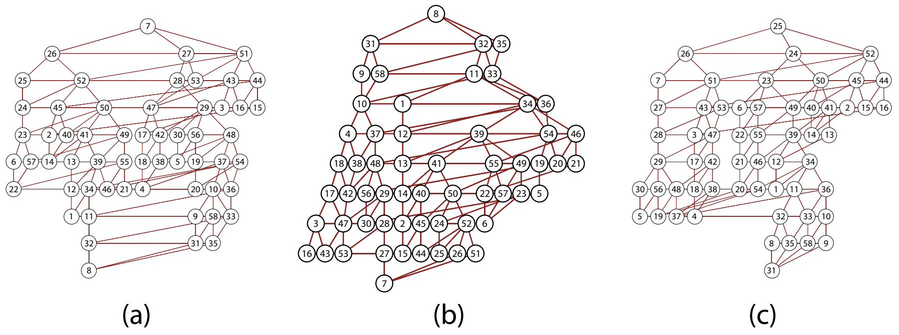

Figure 2. Level structures rooted from node 7, 8, and 25 (a) $L_7(G)$ (b) $L_8(G)$ (c) $L_{25}(G)$.

Table 1. Associated level pairs/sets in GPS/GGPS algorithm.

node	GPS	GGPS	node	GPS	GGPS	node	GPS	GGPS
1	(8,7)	(8,7,7)	21	(7,5)	(7,6,5)	41	(5,5)	(5,4,5)
2	(5,3)	(5,4,3)	22	(7,4)	(7,5,4)	42	(5,4)	(5,6,4)
3	(4,3)	(4,5,3)	23	(5,4)	(5,3,4)	43	(3,2)	(3,4,2)
4	(7,6)	(7,8,6)	24	(4,3)	(4,2,3)	44	(3,2)	(3,3,2)
5	(6,4)	(6,8,4)	25	(3,2)	(3,1,2)	45	(4,3)	(4,3,3)
6	(6,3)	(6,4,3)	26	(2,2)	(2,2,2)	46	(7,6)	(7,6,6)
7	(1,1)	(1,3,1)	27	(2,2)	(2,4,2)	47	(4,3)	(4,5,3)
8	(10,10)	(10,9,10)	28	(3,3)	(3,5,3)	48	(5,5)	(5,7,5)
9	(8,8)	(8,9,8)	29	(4,4)	(4,6,4)	49	(5,5)	(5,4,5)
10	(7,7)	(7,8,7)	30	(5,3)	(5,7,3)	50	(4,4)	(4,3,4)
11	(8,8)	(8,7,8)	31	(9,9)	(9,10,9)	51	(2,2)	(2,3,2)
12	(7,6)	(7,6,6)	32	(9,9)	(9,8,9)	52	(3,3)	(3,2,3)
13	(6,5)	(6,5,5)	33	(8,8)	(8,8,8)	53	(3,2)	(3,4,2)
14	(6,4)	(6,5,4)	34	(7,7)	(7,6,7)	54	(6,6)	(6,7,6)
15	(4,2)	(4,4,2)	35	(9,9)	(9,9,9)	55	(6,5)	(6,5,5)
16	(4,2)	(4,4,2)	36	(7,7)	(7,7,7)	56	(5,4)	(5,7,4)
17	(5,4)	(5,6,4)	37	(6,6)	(6,8,6)	57	(6,4)	(6,4,4)
18	(6,5)	(6,7,5)	38	(6,5)	(6,7,5)	58	(8,8)	(8,9,8)
19	(6,5)	(6,8,5)	39	(6,6)	(6,5,6)			
20	(7,5)	(7,7,5)	40	(5,4)	(5,4,4)			

choose $L_{25}(G)$ to be the "new" level structure, and the level width of our "new" level structure is 11.

For GPS method, by the associated level pairs, the new level structure $L(G)$ is constructed by $L_7(G)$ and $L_{25}(G)$ which as follows:

L_1: 7;
L_2: 25, 26, 27, 51;
L_3: 6, 24, 28, 30, 43, 44, 52, 53;
L_4: 3, 5, 15, 16, 22, 23, 29, 45, 47, 50, 56, 57;
L_5: 2, 17, 19, 20, 21, 40, 41, 42, 48, 49, 55;
L_6: 13, 14, 18, 37, 38, 39, 46, 54;
L_7: 4, 10, 12, 34, 36;

The level width of the GPS new level structure is 12. Compared with the level width of the "new" level structure in GGPS method, the level width of the GPS new level structure is larger than the former one by 1.

By renumbering the nodes in the GGPS new level structure level by level, using the sum of the numbers of the adjacent nodes for each sum of the adjacent nodes codes, for example, tthe nodes in L_2 of $L_{25}(G)$ in the order of increasing sum are 26, 24 and 52, so we renumber 26 to be 2 (node 25 has been renumbered to be 1). The nodes adjacent to 26 in L_2 are 24, so we renumber 24 to be 3. The last one 52 is renumbered to be 4. From L_3 to L_{10} using the same renumbering rule as in L_2, we obtain a new numbering of the graph with a bandwidth of 11 and a profile of 401, while the bandwidth and profile in GPS method is 15 and 429, respectively. The reduction ration of bandwidth and profile compared with GPS method

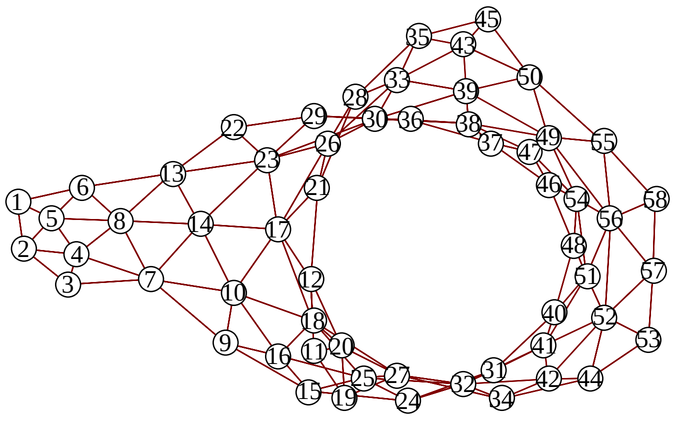

Figure 3. Graph with GGPS numbering.

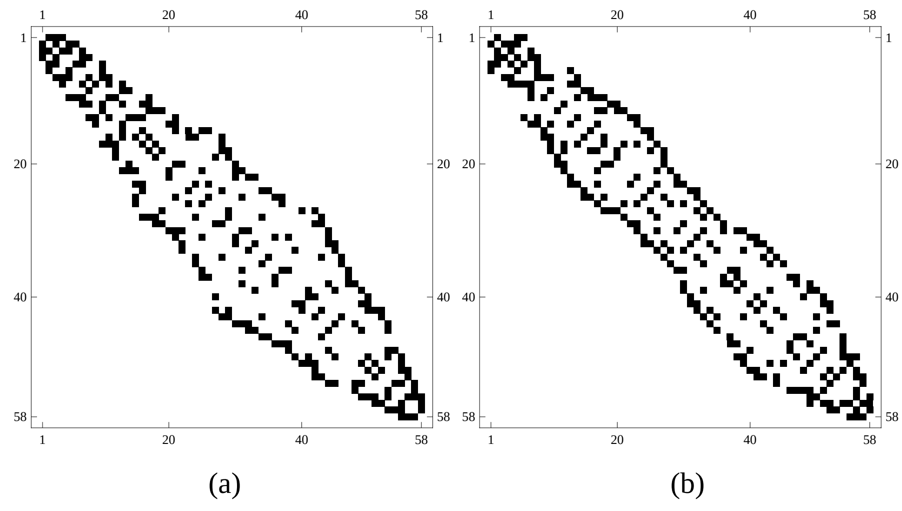
Figure 4. Adjacency matrices of GPS and GGPS numbering (a) GPS (b) GGPS.

is 26.67% and 6.53%. The graph with the new numbering is shown in Fig. 3. The sparse matrices structures of GPS renumbering and GGPS renumbering are shown in Fig. 4. The sparse matrix structure of GGPS renumbering is apparently slim and uniform compared with the sparse matrix structure of GPS renumbering, which means our renumbering is better than the GPS renumbering.

4. DESCRIPTION OF TEST RESULTS

The test problems usefully answering two-dimensional questions may not be useful for answering the three-dimensional questions. In order for the problems to be representative of a fairly large class of applications, and also in order to assess the time complexity carefully, the authors chose 21 plane models for two-dimensional problems and 21 solid models for three-dimensional problems. The proposed method is applied to different geometries, and both GGPS and GPS algorithm are coded using FORTRAN 95 programming language.

Table 2 gives the test results and comparisons for 21 volumes with tetrahedron meshes. Table 3 gives the results and comparisons for 21 plane models with triangular meshes. Table 4 gives the computation time ratio between GPS procedure and GGPS procedure.

From the results of solid models we get the means which are 37.63% in bandwidth reduction, 0.17% in profile reduction, and 1.03

Table 2. Test results for solid models (bandwidth and profile) (β_g —— bandwidth of GPS, P_g —— profile of GPS; β_{gg} —— bandwidth of GGPS, P_{gg} —— profile of GGPS; m —— the number of pseudo-peripheral nodes).

nodes	GPS		GGPS			$1-\beta_{gg}/\beta_g$	$1-P_{gg}/P_g$
	β_g	P_g	β_{gg}	P_{gg}	m	%	%
23	12	163	10	154	8	16.67	5.52
194	119	3889	35	4072	9	70.59	−4.71
446	106	30185	101	29619	4	4.72	1.88
2454	2440	183318	159	191055	3	93.48	−4.22
2457	229	318135	212	316032	9	7.42	0.66
2576	219	339831	207	335349	24	5.48	1.32
2803	381	621545	367	604018	3	3.67	2.82
3678	136	321034	141	320035	2	−3.68	0.31
4184	354	792644	327	807905	12	7.63	−1.93
4331	422	1101917	413	1114717	9	2.13	−1.16
7519	551	2902026	542	2943995	17	1.63	−1.45
7914	4100	920858	179	924153	5	95.63	−0.36
7984	592	3070727	577	3177287	33	2.53	−3.47
11319	8775	1628056	216	1624966	2	97.54	0.19
11658	11634	6306696	913	5942830	2	92.15	5.77
13190	12950	5827201	787	5986765	3	93.92	−2.74
19262	17870	10392439	815	9909131	2	95.44	4.65
22298	22261	15323169	1090	15381846	6	95.10	−0.38
24058	1128	7538107	1058	7557882	25	6.21	−0.26
31798	1352	24106621	1368	24065426	9	−1.18	0.17
149610	864	94013839	836	93156094	13	3.24	0.91

in time ratio, compared with the results of GPS method. In some cases the bandwidth can be reduced by more than 90%. For plane models the means are 8.91% in bandwidth reduction, 2.29% in profile reduction, and 1.04 in time ratio. For bandwidth reduction there are still some large percentages such as 99.51%. Based on the analysis, we can infer that the GPS method is less stable in reducing bandwidth compared with the GGPS method. In most cases for both solid and

Table 3. Test results for plane models (bandwidth and profile) (β_g —— bandwidth of GPS, P_g —— profile of GPS; β_{gg} —— bandwidth of GGPS, P_{gg} —— profile of GGPS; m —— the number of pseudo-peripheral nodes).

nodes	GPS		GGPS			$1-\beta_{gg}/\beta_g$	$1-P_{gg}/P_g$
	β_g	P_g	β_{gg}	P_{gg}	m	%	%
41	9	233	9	237	6	0	−1.46
43	10	268	10	264	13	0	1.29
44	7	229	7	228	4	0	0.37
49	13	349	11	327	16	14.29	5.53
73	12	599	12	587	9	0	1.79
92	16	861	14	838	3	11.76	2.41
148	19	1862	18	1815	13	5.00	2.34
199	21	2871	21	2851	9	0	0.65
677	39	18476	39	18598	8	0	−0.64
721	31	19300	32	19524	6	20.75	15.28
738	24	14709	23	14695	2	4.00	0.09
738	41	20838	40	20969	34	2.38	−0.61
741	40	18352	39	19013	31	2.44	−3.46
750	41	21655	39	21377	11	4.76	1.24
837	52	27505	50	25986	48	3.77	5.36
1074	50	37891	47	38130	11	6.25	6.48
3683	82	204288	81	201899	2	1.20	1.15
9338	135	856854	127	814643	4	5.88	4.87
14556	168	1662348	168	1662614	2	0	−0.02
37129	269	6797501	253	6485421	4	5.22	4.57
72610	72473	20743713	357	20543390	6	99.51	0.96

plane models the profiles are slightly reduced compared with GPS method. The number of pseudo-peripheral nodes is increased in most cases, and some number is as great as 48. The results show that the increase in pseudo-peripheral nodes does contribute to the reduction in bandwidths and profiles. Further, in some cases, though the number of pseudo-peripheral nodes is also the same as in GPS method, bandwidth can still be reduced by more than 90%, proving the GGPS method

Table 4. Test results for solid models (bandwidth and profile) (T_g —— time of GPS, T_{gg} —— time of GGPS).

plane				solid			
nodes	T_{gg}/T_g	nodes	T_{gg}/T_g	nodes	T_{gg}/T_g	nodes	T_{gg}/T_g
23	1.500	7914	1.012	41	1.000	738	1.000
194	1.000	7984	1.008	43	1.000	741	1.170
446	1.000	11319	1.005	44	1.000	750	1.170
2454	1.024	11658	1.004	49	1.000	837	1.000
2457	1.021	13190	1.005	73	1.000	1074	1.080
2576	1.023	19262	1.002	92	1.000	3683	1.026
2803	1.013	22298	1.002	148	1.000	9338	1.010
3678	1.027	24058	1.001	199	1.000	14556	1.008
4184	1.011	31798	1.003	677	1.000	37129	1.003
4331	1.010	149610	1.043	721	1.170	72610	1.002
7519	1.007			738	1.174		

could find more suitable pseudo-peripheral nodes. The results in Table 4 shows the increased complexity in the proposed method makes little effect in time increasing.

For a banded solver using Cholesky resolution, a 38% bandwidth reduction leads to a decrease of needed operations by about 75% compared with the GPS method. Indeed, the number of operations is reduced by square value of bandwidth for such direct solvers.

5. CONCLUSIONS

Based on the GPS algorithm, we have developed a more generalized algorithm to provide high quality solutions to the problem of minimizing the matrix bandwidth and profile. Overall experiments with 42 instances were performed to assess the merit of the proposed GGPS method, which are more competitive by a set of data reported in the paper. The reduction in bandwidth is impressive, as the means are 37.63% for solid models and 8.91% for plane models. The profiles are reduced in a small range in average, while the means are 0.17% for solid models and 2.29% for plane models.

The modifications embody the generalization of the original GPS algorithm. Compared with the GPS method, in most cases the GGPS produces more pseudo-peripheral nodes, leading to better numberings, only slightly worse in computation time. The reduction in bandwidth compared with the GPS method leads to a squared ratio reduction in the number of operations needed for LU or a complete Cholesky's resolution scheme. The profile of stiffness matrix is also reduced, offering significant savings of memory storage which accelerate the resolution scheme. It was concluded that the GGPS has a more generalized rules, hence, the reordering of the matrix can be always better than the GPS reordering. The GGPS algorithm can be widely used in the reordering of sparse matrix in the finite element method.

ACKNOWLEDGMENT

This work was funded by the 9140A070113, the 51307060101 and the Aviation Science 20070181002.

REFERENCES

1. Wang, Q., Y. C. Guo, and X. W. Shi, "An improved matrix bandwidth and profile reduction algorithm in FEM problems," *Progress In Electromagnetics Research Symposium*, Hangzhou, China, 2008.
2. Amjadi, S. M. and M. Soleimani, "Design of band-pass waveguide filter using frequency selective surfaces loaded with surface mount capacitors based on split-field update FDTD method," *Progress In Electromagnetics Research B*, Vol. 3, 271–281, 2008.
3. Zhou, X. and G. W. Pan, "Application of physical spline finite element method (PSFEM) to fullwave analysis of waveguides," *Progress In Electromagnetics Research*, PIER 60, 19–41, 2006.
4. Wei, X. C., E. P. Li, and Y. J. Zhang, "Application of the improved finite element-fast multipole method on large scattering problems," *Progress In Electromagnetics Research*, PIER 47, 49–60, 2004.
5. Doncker, P. D., "The use of tansfinite elements in the methods of moments applied to electromagnetic scattering by dielectric cylinders," *Progress In Electromagnetics Research*, PIER 25, 77–94, 2000.
6. Bedrosian, G., "High-performance computing for finite element methods in low-frequency electromagnetics," *Progress In Electromagnetics Research*, PIER 07, 57–110, 1993.
7. Vaish, A. and H. Parthasarathy, "Analysis of a rectangular waveg-

uide using finite element method," *Progress In Electromagnetics Research C*, Vol. 2, 117–125, 2008.

8. Hernandez, L. M. A. and M. G. Quintillan, "A finite element method code to analyse waveguide dispersion," *Journal of Electromagnetic Waves and Applications*, Vol. 21, No. 3, 397–408, 2007.
9. Jin, J. M., *The Finite Element Method in Electromagnetics*, Wiley, New York, 2002.
10. Cuthill, E. and J. Mckee, "Reducing the bandwidth of sparse symmetric matrices," *Proceedings of the 1969 24th National Conference*, 157–172, 1969.
11. Reid, J. K. and J. A. Scott, "Implementing Hager's exchange methods for matrix profile reduction," *ACM Transcations on Mathematical Software*, Vol. 28, No. 4, 377–391, 2002.
12. Chan, W. M. and A. George, "A lineartime implementation of the reverse Cuthill-McKee algorithm," *BIT Numerical Mathematics*, Vol. 20, No. 1, 8–14, 1980.
13. Gibbs, N. E., "Algorithm 509: A hybrid profile reduction algorithm [F1]," *ACM Transactions on Mathematical Software*, Vol. 2, No. 4, 378–387, 1976.
14. Gibbs, N. E., W. G. Poole, Jr., and P. K. Stockmeyer, "An algorithm for reducing the bandwidth and profile of a sparse matrix," *SIAM Journal on Numerical Analysis*, Vol. 13, No. 2, 236–250, 1976.
15. George, A. and J. W. H. Liu, "An implementation of a pseudoperipheral node finder," *ACM Transactions on Mathematical Software*, Vol. 5, No. 3, 284–295, 1979.
16. Souza, L. T. and D. W. Murray, "An alternative pseudoperipheral node finder for resequencing schemes," *International Journal for Numerical Methods in Engineering*, Vol. 36, No. 19, 3351–3379, 1993.
17. Tai, C. C. and Y. L. Pan, "Finite element method simulationof photoinductive imaging for cracks," *Progress In Electromagnetics Research Letters*, Vol. 2, 53–61, 2008.
18. Boutora, Y., N. Takorabet, R. Ibtiouen, and S. Mezani, "A new method for minimizing the bandwidth and profile of square matrices for triangular finite elements mesh," *IEEE Transactions on Magnetics*, Vol. 43, No. 4, 1513–1516, 2007.