Divulgaciones Matemáticas Vol. 7 No. 1 (1999), pp. 1–12

Reducción del Ancho de Banda de Matrices en el Algoritmo Go-Away para Mallas Regulares

On Reducing Bandwith of Matrices in the Go-Away Algorithm for Regular Grids

Pedro Ramón Almeida Benítez
Departamento de Matemáticas
Universidad de Las Palmas de Gran Canaria
Las Palmas de Gran Canaria. España

José Ramón Franco Brañas (`jfranco@ull.es`)
Departamento de Análisis Matemático
Universidad de La Laguna
La Laguna. Tenerife. España

Resumen

En este artículo se considera el algoritmo Go-Away [1] para resolver sistemas lineales de ecuaciones con grafo en forma de malla. Este algoritmo fue diseñado originalmente para resolver problemas de aplicaciones de elementos finitos y diferencias finitas. Aquí se compara el modo habitual de reordenar los nodos de los bloques con el orden obtenido al aplicar al algoritmo Go-Away el reordenamiento de Cuthill-McKee [3]. La reducción en el efecto *fill-in* es notable si se compara con los algoritmos de Disección Anidada [5], Disección Unidireccional [9] o Grado Mínimo [5].
Palabras y frases clave: sistemas *sparse*, mallas de elementos finitos, eliminación gaussiana, algoritmo de Cuthill-McKee, algoritmo Go-Away.

Abstract

In this paper we consider the Go-Away algorithm [1] to solve linear systems of equations with graph in grid form. This method is appropriate primarily for matrix problems arising in finite differences and finite

Recibido 1998/12/09. Revisado 1999/02/10. Aceptado 1999/02/18.
MSC (1991): Primary 65F50.

elements applications. Here we compare the usual way of reordering the nodes with the order obtained by applying to the Go-Away algorithm the Cuthill-McKee reordering [3]. The reduction in the fill-in effect is remarkable if we compare with Nested Dissection [5], One-Way [9] or Minimum Degree [5] algorithms.
Key words and phrases: sparse systems, gaussian elimination, finite elements grids, Cuthill-McKee's algorithm, Go-Away algorithm.

1 Introducción

Sea A una matriz simétrica definida positiva de dimensiones $n \times n$. Se quiere resolver el sistema de ecuaciones lineales $Ax = b$, donde x y b son vectores columna de orden n. Se factoriza la matriz A en la forma $A = LL^t$, donde L es una matriz triangular inferior. A continuación, se resuelven los sistemas triangulares $Ly = b$ y $L^t x = y$.

Si A es densa, el número de multiplicaciones/divisiones necesarias para la factorización de Cholesky es $n^3/6 + n^2/2 - 2n/3$ y el total de operaciones necesarias para resolver el sistema, dado el factor de Cholesky L, es igual a $n^2 + n$.

Si A es *sparse*, será posible ahorrar tiempo y almacenamiento manipulando los ceros. Una dificultad que se presenta durante el proceso de factorización de la matriz A es que dicho proceso origina entradas no nulas en posiciones de L que eran nulas en A. Este hecho se conoce con el nombre de efecto *fill-in*, que implica un aumento de la demanda de almacenamiento, mayor número de operaciones y, por tanto, un incremento en los errores de redondeo.

Existen diferentes algoritmos que intentan minimizar el efecto *fill-in* durante la factorización. Unos lo consiguen mediante una minimización local de dicho efecto. Son las estrategias de Markowitz [16], Grado Mínimo [5], etc. Algunos intentan reducir el ancho de banda: Cuthill-McKee [3], King [13], etc. Finalmente, otros intentan dividir el grafo de la matriz A en bloques sin *fill-in* entre ellos: Disección Unidireccional [9], Disección Anidada [5], Go-Away [1].

Por tanto, el proceso total se puede dividir en cuatro etapas:

1. Encontrar un buen ordenamiento para la matriz A.
2. Factorización simbólica. Es interesante conocer las posiciones no nulas de la matriz triangular L antes de comenzar la factorización numérica con el objeto de reducir el coste de dicha factorización.
3. Factorización numérica. Computar L.
4. Resolver los sistemas $Ly = b$ y $L^t x = y$.

En el presente artículo se considera el algoritmo Go-Away para resolver sistemas de ecuaciones lineales con grafo en forma de malla, comparando el modo habitual de reordenar los bloques con el orden obtenido al aplicar a dicho algoritmo el reordenamiento de Cuthill-McKee a la matriz A.

2 Sistemas en banda

Dada una matriz simétrica $A = (a_{ij})$, de dimensiones $n \times n$, se define el *semiancho de banda* de A como:

$$\beta = \max |i - j|, \; a_{ij} \neq 0.$$

En una matriz en banda, la eliminación gaussiana puede ser realizada dentro de la banda, permaneciendo nulas las entradas externas a ella. Entonces, sería deseable encontrar una permutación A' de A tal que el semiancho de banda de A' fuese mínimo. Desgraciadamente, encontrar A' no es tarea fácil. Es un problema NP-completo. Sin embargo, existen buenos esquemas heurísticos de ordenamiento que han sido ampliamente estudiados en las últimas décadas.

Teorema 1. *Sea A una matriz simétrica definida positiva de dimensiones $n \times n$ y semiancho de banda β. El número de operaciones requeridas para factorizar dicha matriz en la forma $A = LL^t$, siendo L una matriz triangular inferior, suponiendo que la banda de $L + L^t$ está llena, es igual a*

$$\beta(\beta + 2)n - 2\beta^3/3 - 3\beta^2/2 - 5\beta/6.$$

Demostración. Al aplicar la eliminación gaussiana, para hacer nulas las entradas subdiagonales de cada una de las $n - \beta$ primeras columnas, se han de realizar β divisiones y $\beta(\beta + 1)$ multiplicaciones (ver fig. 1). En total son: $(n - \beta)[\beta + \beta(\beta + 1)] = (n - \beta)(\beta^2 + 2\beta)$.

$$A = \begin{pmatrix} 1 & x & x & x & & & & & \\ x & 2 & x & x & x & & & & \\ x & x & 3 & x & x & x & & & \\ x & x & x & 4 & x & x & x & & \\ & x & x & x & 5 & x & x & x & \\ & & x & x & x & 6 & x & x & x \\ & & & x & x & x & 7 & x & x \\ & & & & x & x & x & 8 & x \\ & & & & & x & x & x & 9 \end{pmatrix}$$

Figura 1. Matriz con semiancho de banda $\beta = 3$.

Para la columna $n-\beta+1$:

$$(\beta-1)+(\beta-1)\beta=(\beta-1)(\beta+1)$$

Para la columna $n-\beta+2$:

$$(\beta-2)+(\beta-2)(\beta-1)=(\beta-2)\beta$$

Para la columna $n-\beta+3$:

$$(\beta-3)+(\beta-3)(\beta-2)=(\beta-3)(\beta-1)$$

Y así sucesivamente, hasta la última columna. En total:

$$\begin{aligned}\sum_{i=1}^{\beta-1}(\beta-i)(\beta-i+2) &= \sum_{i=1}^{\beta-1}(\beta^2-2\beta i+i^2+2\beta-2i)\\ &= \sum_{i=1}^{\beta-1}[(\beta^2+2\beta)-(2\beta+2)i+i^2]\\ &= (\beta^2+2\beta)(\beta-1)-(2\beta+2)\frac{1+(\beta-1)}{2}(\beta-1)\\ &\quad+\frac{(\beta-1)\beta(2\beta-2+1)}{6}=\beta^3/3+\beta^2/2-5\beta/6.\end{aligned}$$

Sumando con lo anterior:

$$(n-\beta)(\beta^2+2\beta)+\beta^3/3+\beta^2/2-5\beta/6=\beta(\beta+2)n-2\beta^3/3-3\beta^2/2-5\beta/6.$$

□

Si A es densa, entonces $n-1=\beta$. Sustituyendo β en la última expresión, se obtiene un total de operaciones igual a $n^3/6+n^2/2-2n/3$, como era de esperar (ver 1 Introducción).

Teorema 2. *Sea A una matriz simétrica definida positiva de dimensiones $n\times n$ y semiancho de banda β. Dado el factor de Cholesky L, el número de operaciones necesarias para resolver el sistema $Ax=b$, es igual a*

$$(\beta+1)(2n-\beta).$$

Demostración. Para la resolución de $L^tx=y$ hay que realizar un total de $1+2+3+...+\beta+(n-\beta-1)\beta$ multiplicaciones y n divisiones al sustituir los valores de las incógnitas en las ecuaciones precedentes (ver fig. 2). Esto es:

$$\frac{1+\beta}{2}\beta+n\beta-\beta^2-\beta+n=\frac{-\beta^2-\beta+2n\beta+2n}{2}.$$

$$A = LL^t = \begin{pmatrix} l_{11} & & & & & & & & \\ x & l_{22} & & & & & & & \\ x & x & l_{33} & & & & & & \\ x & x & x & l_{44} & & & & & \\ & x & x & x & l_{55} & & & & \\ & & x & x & x & l_{66} & & & \\ & & & x & x & x & l_{77} & & \\ & & & & x & x & x & l_{88} & \\ & & & & & x & x & x & l_{99} \end{pmatrix} \begin{pmatrix} l_{11} & x & x & x & & & & & \\ & l_{22} & x & x & x & & & & \\ & & l_{33} & x & x & x & & & \\ & & & l_{44} & x & x & x & & \\ & & & & l_{55} & x & x & x & \\ & & & & & l_{66} & x & x & x \\ & & & & & & l_{77} & x & x \\ & & & & & & & l_{88} & x \\ & & & & & & & & l_{99} \end{pmatrix}$$

Figura 2. Matriz con semiancho de banda $\beta = 3$ factorizada LL^t.

Para resolver $Ly = b$, hay que realizar el mismo número de operaciones. En total:

$$-\beta^2 - \beta + 2n\beta + 2n = (\beta + 1)(2n - \beta).$$

□

Al igual que antes, si A es densa, entonces $n - 1 = \beta$. Sustituyendo β en la última expresión, se obtiene un total de operaciones igual a $n^2 + n$ (ver 1 Introducción).

3 El algoritmo de Cuthill-McKee (CM)

La teoría de grafos es una herramienta de gran utilidad para el estudio de los sistemas *sparse*. La distribución de las entradas no nulas de una matriz puede ser representada mediante un grafo y ser utilizado éste para visualizar lo que ocurre durante la computación. Dada una matriz simétrica A, si la entrada a_{ij} de A es distinta de cero, los nodos i y j están conectados en el grafo mediante una arista. La eliminación gaussiana en la matriz queda reflejada en el grafo al eliminar el nodo correspondiente y aparecer nuevas conexiones entre los nodos restantes, correspondiendo cada una de ellas a un *fill-in* en A.

El método más ampliamente utilizado para la transformación en banda de una matriz *sparse* es el conocido con el nombre de algoritmo de Cuthill-McKee (CM) [3].

La idea del algoritmo es muy simple: Si a es un vértice ya renumerado y b es un vértice no renumerado aún, conectado al vértice a mediante una arista en el grafo, para minimizar la anchura de la fila asociada a b es evidente que el vértice b se debe renumerar lo antes posible, inmediatamente después del vértice a.

El esquema del algoritmo es el siguiente:

1. Se construye una tabla indicando el número de conexiones (que se suele llamar *grado* y se representa $\delta(x)$) de cada vértice.
2. Se elige un nodo inicial (en un extremo del grafo y con pocas conexiones), que se renumera x_1.
3. Se renumeran a continuación los vértices conectados a x_1, en orden ascendente de grado.

En la figura 3 se representa una matriz simétrica A y su grafo asociado. A la derecha, se representa la matriz A', que ha sido reordenada mediante el algoritmo de Cuthill-McKee (CM). Se puede apreciar el reordenamiento en forma de banda de dicha matriz A'. Se ha elegido como inicial el vértice j (en un extremo del grafo y con pocas conexiones), aunque existen procedimientos más refinados para dicha elección (ver [10,14]).

$$A = \begin{pmatrix} a & & & x & & x & & & & \\ & b & x & & & & x & & & x \\ & x & c & x & x & & & & & \\ x & & x & d & & x & & & x & \\ & & x & & e & & x & & & \\ x & & & x & & f & x & x & & \\ & x & & & x & x & g & x & & \\ & & & & & x & x & h & & \\ & & & x & & & & & i & \\ & x & & & & & & & & j \end{pmatrix} \quad A' = \begin{pmatrix} j & x & & & & & & & & \\ x & b & x & x & & & & & & \\ & x & c & & x & x & & & & \\ & x & & g & x & & x & x & & \\ & & x & x & e & & & & & \\ & & x & & & d & & x & x & x \\ & & & x & & & h & x & & \\ & & & x & & x & x & f & & x \\ & & & & & x & & & i & \\ & & & & & x & & x & & a \end{pmatrix}$$

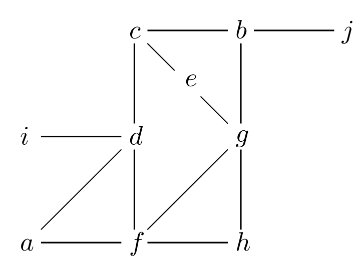

Figura 3. La matriz A, la matriz reordenada con CM y el grafo asociado.

4 Algoritmo para colorear un grafo

El algoritmo expuesto a continuación permite colorear un grafo utilizando un reducido número de colores y sin que haya dos vértices adyacentes con igual color. El objetivo es conseguir una partición del grafo con el menor número posible de clases en dicha partición [1]. La estrategia es la siguiente:

1. Se elige un vértice de máximo grado x. Sea A_1={x}.
2. Entre los vértices no adyacentes al conjunto A_1, se elige un vértice de máximo grado que se añade a A_1.
3. Del mismo modo, se continúa añadiendo vértices a A_1 (no adyacentes y de máximo grado) hasta que no queden más vértices.
4. Con los vértices restantes, se repite el proceso, formando un nuevo conjunto A_2 y así sucesivamente.
5. Se repite el proceso hasta que no queden vértices seleccionables en el grafo. Finalmente, a los nodos de cada conjunto A_k se les asigna el color k.

Intencionadamente, el algoritmo deja para el final los vértices de menor grado. De este modo, los vértices del último A_k tienen pocas conexiones y esto contribuye a la disminución del número de los A_k.

En la figura 4 se puede apreciar la coloración obtenida al aplicar al grafo de la izquierda el algoritmo anterior. En este caso, $A_1 = \{d, g, j\}$, $A_2 = \{f, b, e, i\}$ y $A_3 = \{c, a, h\}$. El resultado es el grafo de la derecha de la figura, en donde cada vértice de A_k ha sido designado con el color k.

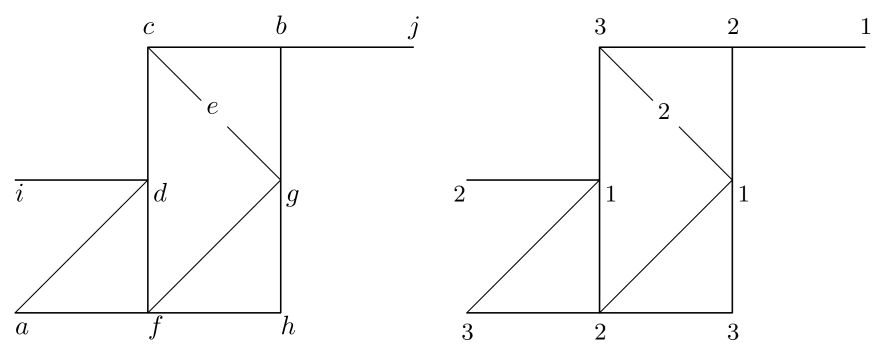

Figura 4. Ejemplo de grafo coloreado.

5 El algoritmo Go-Away para mallas regulares

Sea un grafo en forma de malla con $m \times n$ nodos, como el de la figura:

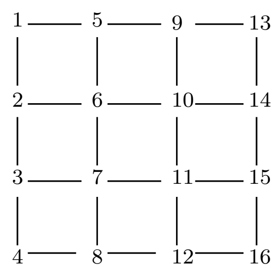

Figura 5. Malla de 4x4.

donde los nodos son las esquinas de los cuadrados de la malla correspondiente a la matriz A de un sistema $Ax = b$. Utilizando el algoritmo anterior, los nodos de la malla pueden ser coloreados con dos colores solamente: $A_1 = \{6, 11, 3, 8, 9, 14, 1, 16\}$, $A_2 = \{7, 10, 2, 5, 12, 15, 4, 13\}$. Entonces, la matriz reordenada $6, 11, 3, 8, 9, 14, 1, 16, 7, 10, 2, 5, 12, 15, 4, 13$, queda dividida en cuatro bloques:

$$\begin{pmatrix} D_{11} & A_{21}^t \\ A_{21} & D_{22} \end{pmatrix}$$

donde las matrices D_{11} y D_{22} son diagonales, correspondiendo a A_1 y A_2, y las entradas no nulas exteriores a la diagonal quedan confinadas en las matrices A_{21} y A_{21}^t. Además, el *fill-in* aparecerá únicamente en la matriz D_{22}:

$$\begin{pmatrix}
6 & & & & & & & & * & * & * & * & & & & \\
& 11 & & & & & & & * & * & & & * & * & & \\
& & 3 & & & & & & * & & * & & & & * & \\
& & & 8 & & & & & * & & & & * & & * & \\
& & & & 9 & & & & & * & & * & & & & * \\
& & & & & 14 & & & & * & & & & * & & * \\
& & & & & & 1 & & & & * & * & & & & \\
& & & & & & & 16 & & & & & * & * & & \\
* & * & * & * & & & & & 7 & & & & & & & \\
* & * & & & * & * & & & & 10 & & & & & & \\
* & & * & & & & * & & & & 2 & & & & & \\
* & & & & * & & * & & & & & 5 & & & & \\
& * & & * & & & & * & & & & & 12 & & & \\
& * & & & & * & & * & & & & & & 15 & & \\
& & * & * & & & & & & & & & & & 4 & \\
& & & & * & * & & & & & & & & & & 13
\end{pmatrix}$$

Figura 6. Matriz de la malla de la figura 5, reordenada con Go-Away.

Pero el efecto *fill-in* se puede reducir aún más aplicando el algoritmo de Cuthill-McKee(CM) a la matriz original A y reordenando los bloques internamente para su transformación en banda y limitar dicho efecto *fill-in*:

$$
\begin{pmatrix}
1 & & & & & & & & \ast & \ast & & & & & & \\
& 3 & & & & & & & \ast & & \ast & \ast & & & & \\
& & 6 & & & & & & \ast & \ast & & \ast & \ast & & & \\
& & & 9 & & & & & & \ast & & & \ast & \ast & & \\
& & & & 8 & & & & & & \ast & \ast & & & \ast & \\
& & & & & 11 & & & & & & \ast & \ast & & \ast & \ast \\
& & & & & & 14 & & & & & & \ast & \ast & & \ast \\
& & & & & & & 16 & & & & & & & \ast & \ast \\
\ast & \ast & \ast & & & & & & 2 & & & & & & & \\
\ast & & \ast & \ast & & & & & & 5 & & & & & & \\
& \ast & & & \ast & & & & & & 4 & & & & & \\
& \ast & \ast & & \ast & \ast & & & & & & 7 & & & & \\
& & \ast & \ast & & \ast & \ast & & & & & & 10 & & & \\
& & & \ast & & & \ast & & & & & & & 13 & & \\
& & & & \ast & \ast & & \ast & & & & & & & 12 & \\
& & & & & \ast & \ast & \ast & & & & & & & & 15
\end{pmatrix}
$$

Figura 7. Matriz de la figura 6, reordenada con Cuthill-McKee.

En efecto, para la matriz anterior, el algoritmo de Cuthill-McKee proporciona el ordenamiento siguiente: 1,2,5,3,6,9,4,7,10,13,8,11,14,12,15,16, tomando como nodo inicial el 1. Posteriormente, se han reordenado internamente los bloques diagonales D_{11} y D_{22} con la secuencia anterior. Esto es, el orden inicial en D_{11} era 6,11,3,8,9,14,1,16 (figura 6). El nuevo ordenamiento será 1,3,6,9,8,11,14,16, ya que en el ordenamiento de Cuthill-McKee, el nodo 1 es anterior al 3, el nodo 3 es anterior al nodo 6, etc. En la figura 7 se aprecia como dicho algoritmo ha transformado en banda las matrices A_{21} y A_{21}^t, donde cada '$\ast$' representa una entrada no nula.

En la figura 8 está representado el factor L^t de la anterior matriz. En ella se puede apreciar como el *fill-in*, indicado con 'o', queda confinado en una banda de la matriz D_{22}.

$$L^t = \begin{pmatrix}
1 & & & & & & & & * & * & & & & & & \\
& 3 & & & & & & & * & & * & * & & & & \\
& & 6 & & & & & & * & * & & * & * & & & \\
& & & 9 & & & & & & * & & & * & * & & \\
& & & & 8 & & & & & & * & * & & & * & \\
& & & & & 11 & & & & & & * & * & & * & * \\
& & & & & & 14 & & & & & & * & * & & * \\
& & & & & & & 16 & & & & & & & * & * \\
& & & & & & & & 2 & o & o & o & o & & & \\
& & & & & & & & & 5 & o & o & o & o & & \\
& & & & & & & & & & 4 & o & o & o & o & \\
& & & & & & & & & & & 7 & o & o & o & o \\
& & & & & & & & & & & & 10 & o & o & o \\
& & & & & & & & & & & & & 13 & o & o \\
& & & & & & & & & & & & & & 12 & o \\
& & & & & & & & & & & & & & & 15
\end{pmatrix}$$

Figura 8. Factor L^t de la matriz de la figura 7.

Teorema 3. *Dada una matriz A, con grafo en forma de malla de dimensiones $n \times n$, reordenada mediante el algoritmo Go-Away con Cuthill-McKee, el* fill-in *en el factor de Cholesky L es igual a*

$$\frac{(n-1)(n+4)(2n+1)}{6}.$$

Demostración. El *fill-in* introducido en la mitad triangular superior de D_{22}, al eliminar las entradas no nulas de A_{21}, viene dado por

$$1 + (3+4+2) + (3+3.4+2) + ... + [3 + (n-3)4 + 2]$$
$$+[22 + (n-3)4 + 1] + ... + (3+4+1) = 2n^2 - 4n + 1.$$

A continuación, se ha de añadir el *fill-in* interno del bloque inferior derecho D_{22}, esto es, el *fill-in* producido por la eliminación de las nuevas entradas subdiagonales no nulas de D_{22}:

$$[1^2 + 2^2 + 3^2 + ... + (n-3)^2] + (n-3)^2 + (n-3) = \frac{(n-3)(n-2)(2n+1)}{6}.$$

Por último, el *fill-in* total será igual a la suma de ambos:

$$2n^2 - 4n + 1 + \frac{(n-3)(n-2)(2n+1)}{6} = \frac{(n-1)(n+4)(2n+1)}{6}.$$

□

6 Resultados numéricos

Mediante la fórmula anterior, se va a calcular el efecto *fill-in* en L^t para distintas mallas utilizando el algoritmo Go-Away con CM y comparando los resultados obtenidos con los algoritmos de Grado Mínimo, One-Way, Nested Dissection y Go-Away (con CM):

Malla	num.ec.	Grad. Min.	One-Way	Nest.Diss.	Go-Away con CM
4×4	16	26	24	18	22
5×5	25	81	59	51	42
6×6	36	191	117	98	75
7×7	49	356	198	159	121
8×8	64	634	394	253	182

En la tabla se puede apreciar la importante reducción del *fill-in* para distintas mallas regulares al aplicar el algoritmo Go-Away con CM y comparar los resultados con los algoritmos de Grado Mínimo, One-Way y Nested Dissection. Además, se puede apreciar que el algoritmo de Grado Mínimo no es un buen algoritmo para este tipo de mallas.

Por otra parte, la implementación del algoritmo es sencilla como se vió anteriormente. El descenso del efecto *fill-in* implica una importante reducción del coste computacional y una disminución de los errores de redondeo.

Referencias

[1] Almeida Benítez, P. R., Franco Brañas, J. R. *The Go-Away algorithm for Block Factorization of a Sparse Matrix*, Course on algorithms for Sparse Large Scale Linear Algebraic Systems, NATO ASI SERIES, Vol. 508, Kluwer, Londres, 1998, 107–117.

[2] Almeida Benítez, P. R., Franco Brañas, J. R. *El Algoritmo de Disección Unidireccional para Mallas Regulares*, Rev. Acad. Canar. Cienc., VIII (Núm. 1), (1996), 135–142.

[3] Cuthill, E. *Several Strategies for Reducing the Bandwith of Matrices*, Papers of the Symposium on Sparse Matrices and their Applications, IBM Thomas J. Watson Research Center, New York, 1971.

[4] Dongarra, J. J. et al., *Solving Linear Systems on Vector and Shared Memory Computers*, SIAM, Philadelphia, 1991.

[5] Duff, I. S., Erisman, A. M., Reid, J. K. *Direct Methods for Sparse Matrices*, Monographs on Numerical Analysis, Oxford Press, Oxford, 1989.

[6] De la Fuente O'connor, J. L. *Tecnologías Computacionales para Sistemas de Ecuaciones, Optimización Lineal y Entera*, Reverté, Barcelona, 1993.

[7] Gallivan, K. A., et al., *Parallel Algorithms for Matrix Computations*, SIAM, Philadelphia, 1991.

[8] George, A. *Block Elimination of Finite Elements Systems of Equations*, The IBM Symposia Series, Plenum Press, New York, 1972, 101–114.

[9] George, A. *An Automatic One-Way Dissection Algorithm for Irregular Finite Elements Problems*, SIAM J. Num. Anal. **17**(1980), 740–751.

[10] George, A., et al. *Computer Solution of Large Sparse Positive Defined Systems*, Prentice-Hall, London, 1981.

[11] George, A., Ng, E. *Waterloo Sparse Matrix Package. User guide for SPARSPAK-B*, Research Report CS-84-37, Departament of Computer Science, University of Waterloo, Ontario, Canada, 1984.

[12] George, A., Ng, E. *An Implementation of Gaussian Elimination with Partial Pivoting for Sparse Systems*, SIAM J. Sci. Stat. Com. **6** (1985), 390–409.

[13] Jennings, A., McKeown, J. J. *Matrix Computation*, John Wiley and Sons, Chichester, 1992.

[14] Paulino, G. H., et al. *A New Algorithm for Finding a Pseudoperipheral Vertex or the Endpoints of a Pseudodiameter in a Graph*, John Wiley and Sons, Chichester, 1994.

[15] Rose, D. J., Tarjan, R. E., Lueker, J. S. *Algorithmic Aspects of Vertex Elimination on Graphs*, SIAM J. Comput., 1976, 266–283.

[16] Wijshoff, H. A. G. *Direct Methods for Solving Linear Systems*, Papers of the Large-Scale Scientific Parallel Computing Course, Abingdon, 1992.

[17] Zlatev, Z. *Computational Methods for General Sparse Matrices*, Kluwer Academic Publishers, London, 1991.