

CONTENTS

1	Introduction	1
1.1	Introduction	1
1.2	Graph theory	2
1.3	Example of a sparse matrix	5
1.4	Modern computer architectures	10
1.5	Computational performance	12
1.6	Problem formulation	13
1.7	Sparse matrix test collections	14
2	Sparse matrices: storage schemes and simple operations	18
2.1	Introduction	18
2.2	Sparse vector storage	18
2.3	Inner product of two packed vectors	20
2.4	Adding packed vectors	20
2.5	Use of full-sized arrays	22
2.6	Coordinate scheme for storing sparse matrices	22
2.7	Sparse matrix as a collection of sparse vectors	23
2.8	Sherman's compressed index scheme	25
2.9	Linked lists	26
2.10	Sparse matrix in column-linked list	29
2.11	Sorting algorithms	30
	2.11.1 The counting sort	30
	2.11.2 Heap sort	31
2.12	Transforming the coordinate scheme to other forms	32
2.13	Access by rows and columns	34
2.14	Supervariables	35
2.15	Matrix by vector products	36
2.16	Matrix by matrix products	36
2.17	Permutation matrices	37
2.18	Clique (or finite-element) storage	39
2.19	Comparisons between sparse matrix structures	40
3	Gaussian elimination for dense matrices: the algebraic problem	43
3.1	Introduction	43
3.2	Solution of triangular systems	43

3.3	Gaussian elimination	44
3.4	Required row interchanges	46
3.5	Relationship with LU factorization	47
3.6	Dealing with interchanges	49
3.7	LU factorization of a rectangular matrix	49
3.8	Computational sequences, including blocking	50
3.9	Symmetric matrices	52
3.10	Multiple right-hand sides and inverses	54
3.11	Computational cost	55
3.12	Partitioned factorization	57
3.13	Solution of block triangular systems	59
4	Gaussian elimination for dense matrices: numerical considerations	62
4.1	Introduction	62
4.2	Computer arithmetic error	63
4.3	Algorithm instability	65
4.4	Controlling algorithm stability through pivoting	67
4.4.1	Partial pivoting	67
4.4.2	Threshold pivoting	68
4.4.3	Rook pivoting	68
4.4.4	Full pivoting	69
4.4.5	The choice of pivoting strategy	70
4.5	Orthogonal factorization	70
4.6	Partitioned factorization	70
4.7	Monitoring the stability	71
4.8	Special stability considerations	72
4.9	Solving indefinite symmetric systems	74
4.10	Ill-conditioning: introduction	74
4.11	Ill-conditioning: theoretical discussion	75
4.12	Ill-conditioning: automatic detection	79
4.12.1	The LINPACK condition estimator	79
4.12.2	Hager's method	80
4.13	Iterative refinement	80
4.14	Scaling	81
4.15	Automatic scaling	83
4.15.1	Scaling so that all entries are close to one	84
4.15.2	Scaling norms	84
4.15.3	I-matrix scaling	85
5	Gaussian elimination for sparse matrices: an introduction	89
5.1	Introduction	89
5.2	Numerical stability in sparse Gaussian elimination	90

5.2.1	Trade-offs between numerical stability and sparsity	90
5.2.2	Incorporating rook pivoting	92
5.2.3	2×2 pivoting	93
5.2.4	Other stability considerations	93
5.2.5	Estimating condition numbers in sparse computation	94
5.3	Orderings	94
5.3.1	Block triangular matrix	95
5.3.2	Local pivot strategies	96
5.3.3	Band and variable band ordering	96
5.3.4	Dissection	98
5.4	Features of a code for the solution of sparse equations	99
5.4.1	Input of data	101
5.4.2	The ANALYSE phase	101
5.4.3	The FACTORIZE phase	101
5.4.4	The SOLVE phase	102
5.4.5	Output of data and analysis of results	103
5.5	Relative work required by each phase	103
5.6	Multiple right-hand sides	104
5.7	Computation of entries of the inverse	105
5.8	Matrices with complex entries	106
5.9	Writing compared with using sparse matrix software	106
6	Reduction to block triangular form	108
6.1	Introduction	108
6.2	Finding the block triangular form in three stages	109
6.3	Looking for row and column singletons	110
6.4	Finding a transversal	111
6.4.1	Background	111
6.4.2	Transversal extension by depth-first search	113
6.4.3	Analysis of the depth-first search transversal algorithm	115
6.4.4	Implementation of the transversal algorithm	116
6.5	Symmetric permutations to block triangular form	118
6.5.1	Background	118
6.5.2	The algorithm of Sargent and Westerberg	119
6.5.3	Tarjan's algorithm	122
6.5.4	Implementation of Tarjan's algorithm	125
6.6	Essential uniqueness of the block triangular form	125
6.7	Experience with block triangular forms	127
6.8	Maximum transversals	128
6.9	Weighted matchings	130
6.10	The Dulmage–Mendelsohn decomposition	133

7	Local pivotal strategies for sparse matrices	137
7.1	Introduction	137
7.2	The Markowitz criterion	138
7.3	Minimum degree (Tinney scheme 2)	138
7.4	A priori column ordering	140
7.5	Simpler strategies	142
7.6	A more ambitious strategy: minimum fill-in	143
7.7	Effect of tie-breaking on the minimum degree algorithm	145
7.8	Numerical pivoting	147
7.9	Sparsity in the right-hand side and partial solution	149
7.10	Variability-type ordering	153
7.11	The symmetric indefinite case	153
8	Ordering sparse matrices for band solution	156
8.1	Introduction	156
8.2	Band and variable-band matrices	156
8.3	Small bandwidth and profile: Cuthill–McKee algorithm	158
8.4	Small bandwidth and profile: the starting node	162
8.5	Small bandwidth and profile: Sloan algorithm	162
8.6	Spectral ordering for small profile	163
8.7	Calculating the Fiedler vector	166
8.8	Hybrid orderings for small bandwidth and profile	167
8.9	Hager’s exchange methods for profile reduction	168
8.10	Blocking the entries of a symmetric variable-band matrix	169
8.11	Refined quotient trees	170
8.12	Incorporating numerical pivoting	173
	8.12.1 The fixed bandwidth case	173
	8.12.2 The variable bandwidth case	174
8.13	Conclusion	175
9	Orderings based on dissection	177
9.1	Introduction	177
9.2	One-way dissection	177
	9.2.1 Finding the dissection cuts for one-way dissection	179
9.3	Nested dissection	180
9.4	Introduction to finding dissection cuts	182
9.5	Multisection	182
9.6	Comparing nested dissection with minimum degree	185
9.7	Edge and vertex separators	186
9.8	Methods for obtaining dissection sets	188

9.8.1	Obtaining an initial separator set	188
9.8.2	Refining the separator set	189
9.9	Graph partitioning algorithms and software	191
9.10	Dissection techniques for unsymmetric systems	193
9.10.1	Background	193
9.10.2	Graphs for unsymmetric matrices	194
9.10.3	Ordering to singly bordered block diagonal form	197
9.10.4	The performance of the ordering	200
9.11	Some concluding remarks	201
10	Implementing Gaussian elimination without symbolic factorize	204
10.1	Introduction	204
10.2	Markowitz ANALYSE	205
10.3	FACTORIZE without pivoting	210
10.4	FACTORIZE with pivoting	213
10.5	SOLVE	215
10.6	Hyper-sparsity and linear programming	218
10.7	Switching to full form	219
10.8	Loop-free code	221
10.9	Interpretative code	222
10.10	The use of drop tolerances to preserve sparsity	225
10.11	Exploitation of parallelism	228
10.11.1	Various parallelization opportunities	228
10.11.2	Parallelizing the local ordering and sparse factorization steps	228
11	Implementing Gaussian elimination with symbolic FACTORIZE	232
11.1	Introduction	232
11.2	Minimum degree ordering	233
11.3	Approximate minimum degree ordering	236
11.4	Dissection orderings	238
11.5	Numerical FACTORIZE using static data structures	239
11.6	Numerical pivoting within static data structures	240
11.7	Band methods	241
11.8	Variable-band (profile) methods	244
11.9	Frontal methods: introduction	245
11.10	Frontal methods: SPD finite-element problems	246
11.11	Frontal methods: general finite-element problems	250
11.12	Frontal methods for non-element problems	251
11.13	Exploitation of parallelism	255

12 Gaussian elimination using trees	258
12.1 Introduction	258
12.2 Multifrontal methods for finite-element problems	259
12.3 Elimination and assembly trees	263
12.3.1 The elimination tree	263
12.3.2 Using the assembly tree for factorization	266
12.4 The efficient generation of elimination trees	266
12.5 Constructing the sparsity pattern of \mathbf{U}	269
12.6 The patterns of data movement	270
12.7 Manipulations on assembly trees	271
12.7.1 Ordering of children	271
12.7.2 Tree rotations	273
12.7.3 Node amalgamation	276
12.8 Multifrontal methods: symmetric indefinite problems	277
13 Graphs for symmetric and unsymmetric matrices	281
13.1 Introduction	281
13.2 Symbolic analysis on unsymmetric systems	282
13.3 Numerical pivoting using dynamic data structures	283
13.4 Static pivoting	284
13.5 Scaling and reordering	287
13.5.1 The aims of scaling	287
13.5.2 Scaling and reordering a symmetric matrix	287
13.5.3 The effect of scaling	288
13.5.4 Discussion of scaling strategies	289
13.6 Supernodal techniques using assembly trees	290
13.7 Directed acyclic graphs	292
13.8 Parallel issues	294
13.9 Parallel factorization	295
13.9.1 Parallelization levels	295
13.9.2 The balance between tree and node parallelism	297
13.9.3 Use of memory	299
13.9.4 Static and dynamic mapping	300
13.9.5 Static mapping and scheduling	300
13.9.6 Dynamic scheduling	302
13.9.7 Codes for shared and distributed memory computers	303
13.10 The use of low-rank matrices in the factorization	304
13.11 Using rectangular frontal matrices with local pivoting	306
13.12 Rectangular frontal matrices with structural pivoting	310
13.13 Trees for unsymmetric matrices	312

14 The SOLVE phase	315
14.1 Introduction	315
14.2 SOLVE at the node level	316
14.3 Use of the tree by the SOLVE phase	318
14.4 Sparse right-hand sides	318
14.5 Multiple right-hand sides	319
14.6 Computation of null-space basis	319
14.7 Parallelization of SOLVE	320
14.7.1 Parallelization of dense solve	321
14.7.2 Order of access to the tree nodes	321
14.7.3 Experimental results	322
15 Other sparsity-oriented issues	325
15.1 Introduction	325
15.2 The matrix modification formula	326
15.2.1 The basic formula	326
15.2.2 The stability of the matrix modification formula	327
15.3 Applications of the matrix modification formula	328
15.3.1 Application to stability corrections	328
15.3.2 Building a large problem from subproblems	328
15.3.3 Comparison with partitioning	329
15.3.4 Application to sensitivity analysis	330
15.4 The model and the matrix	331
15.4.1 Model reduction	331
15.4.2 Model reduction with a regular submodel	333
15.5 Sparsity constrained backward error analysis	334
15.6 Why the inverse of a sparse irreducible matrix is dense	335
15.7 Computing entries of the inverse of a sparse matrix	337
15.8 Sparsity in nonlinear computations	339
15.9 Estimating a sparse Jacobian matrix	341
15.10 Updating a sparse Hessian matrix	343
15.11 Approximating a sparse matrix by a positive-definite one	344
15.12 Solution methods based on orthogonalization	346
15.13 Hybrid methods	348
15.13.1 Domain decomposition	348
15.13.2 Block iterative methods	350

A Matrix and vector norms	355
B Pictures of sparse matrices	359
C Solutions to selected exercises	367
References	390
AUTHOR INDEX	412
SUBJECT INDEX	418