Ordering Givens Transformations for Sparse QR Factorization

by

Mary Irene Gillespie
B. Sc., University of Saskatchewan, Regina, Saskatchewan, 1974

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science
in the Department
of
Computer Science

We accept this thesis as conforming
to the required standard

[redacted]
Dr. D. D. Olesky, Supervisor (Department of Computer Science)

[redacted]
Dr. F. D. Roberts, Departmental Member (Department of Computer Science)

[redacted]
Dr. P. van den Driessche, Outside Member (Department of Mathematics)

[redacted]
Dr. D. Stanford, External Examiner (College of William and Mary, Math Dept.)

Supervisor: Dr. D. D. Olesky

Abstract

The QR factorization of a matrix A is commonly used to solve least squares problems. One way to compute such a factorization is by using Givens transformations. When A is sparse, the precise order in which the transformations are applied can affect the amount of storage required. In this work we present an ordering for the Givens transformations that is optimal with regard to storage (a so-called "tight" ordering) and that preserves sparsity by restricting fill to those locations in R that are necessarily nonzero when A has the Hall property. This ordering is of particular interest when A does not have the strong Hall property and is not permuted into block upper trapezoidal form.

We describe a bipartite graph model of sparse matrix structures, and summarize the characterization of the structures of the factors Q and R. We define the product of structures of matrices, determine the product of the structures of a sequence of Givens transformations, and define a tight ordering for the Givens transformations. We then present a family of tight orderings for a given matrix structure.

[redacted]

Dr. D. D. Olesky, Supervisor (Department of Computer Science)

[redacted]

Dr. F. D. Roberts, Departmental Member (Department of Computer Science)

[redacted]

Dr. P. van den Driessche, Outside Member (Department of Mathematics)

[redacted]

Dr. D. Stanford, External Examiner (College of William and Mary, Math Dept.)

Contents

Chapter 1

Introduction

1.1 Sparse Matrices

Many problems in diverse fields including science, engineering, surveying, and economics give rise to systems of linear equations that are frequently very large. Problems with thousands of equations and thousands of variables are not unknown. Indeed, in some applications, such as climate modelling, the sizes of the systems are limited by the availability of computing resources, rather than by the size of the underlying problem.

Often in such problems, each equation involves only a few of the variables so that only a small percentage of the entries of the coefficient matrix are nonzero. A typical example might have thousands of rows and thousands of columns, with perhaps only 10 percent of the entries nonzero. If the problem is large enough and the percentage of nonzeros small enough, it becomes advantageous to avoid storing or doing computation on the zero entries. In

this case the matrix is said to be *sparse*

In order to take advantage of sparsity, it is necessary to create data structures for the matrix, and for any factors of the matrix that may be computed during the solution of the problem Ideally, the data structures will be large enough to accommodate all the nonzero entries, but no larger In addition, we would like the data structures to permit easy access to the matrix entries with as little overhead as possible Algorithms that utilize these special data structures are also required This implies an intimate connection between the data structures and the algorithm

In general, given a problem that can be expressed as

$$Ax = b$$

where A is a sparse matrix, x and b are vectors, A and b are known, and it is desired to compute x, the approach is to find factors C and D of A so that $A = CD$ with C and D in some special form that makes solution of the system easy Gaussian elimination is a well-known example that is often applied to square matrices, in which A is factored into a product of L and U, where L is lower triangular and contains the multipliers computed during the forward elimination, and U is an upper triangular matrix Once these matrices are determined, the solution x may be found by forward and back substitution Another example is orthogonal factorization (also known as QR factorization) which may be applied to any matrix having at least as many rows as columns In this factorization, Q has orthonormal columns and R is upper triangular The algorithms used for computing the QR factorization are different from those used to compute the LU factorization, and hence

place different constraints on the data structures used

The basic problem in designing data structures for sparse matrix operations is accommodating *fill* This is the phenomenon in which there is a nonzero at a specified position in one of the factors of A while the corresponding entry in A is zero, so that the factors have more nonzeros than A has In fact, some algorithms may cause so much fill that sparsity is lost entirely, so that the matrix may as well be treated as a full matrix Therefore, it is important to choose algorithms to maintain sparsity as much as possible However, it is rarely possible to eliminate fill entirely, and so the data structures used must be able to handle it

One general approach is to use dynamic data structures such as linked lists, which has the advantage of being quite flexible For example, it allows row and column pivoting operations to be determined during the numerical factorization Dynamic data structures can deal with fill by allocating space and setting pointers for new nonzeros as they are encountered One disadvantage is that for each nonzero, storage is required for the value of the entry, the row index, the column index, pointers to the next entry in the row and the next entry in the column, and perhaps pointers to the previous entries as well Another disadvantage is that it incurs a high overhead in accessing the entries of the matrices since the list must be traversed to find a particular entry

Another approach is to compute a static data structure for each of the resulting matrices ahead of time, before the numerical computation begins The creation of the data structures is called *symbolic factorization* Static data structures have the advantage of requiring less space than dynamic

structures, since fewer pointers are required. Instead, much of the indexing information associated with a particular nonzero entry may be inferred from its location in the data structure and vice versa. Static data structures also require less time to access entries of the matrix, and vectorizing the algorithm is easier. They have the disadvantage of reduced flexibility in the subsequent numerical phase. For instance, it may be impossible to do row or column interchanges to maintain numerical stability within the precomputed data structure, or the precise sequence of row or column interchanges may be fixed in advance, thus prohibiting alternate pivoting sequences. In addition, a preprocessing step is required to compute the data structure. Nevertheless, the advantages often outweigh the disadvantages and static data structures are frequently used (see, for example, [5] [15]).

1.2 Outline of the Thesis

In this thesis we will examine a way to compute a static data structure that is suitable for orthogonal factorization (without column interchanges) of full rank matrices that have at least as many rows as columns. Structures for both the factors Q and R are determined, and in addition, if Givens transformations are used to compute the factorization, then the ordering of the transformations is specified so that the storage required is limited to that determined recently by Hare et al. [11].

In Chapter 2 we give a brief overview of the theory of orthogonal factorization, its application to least squares problems and the use of Givens transformations to compute an orthogonal factorization. We also define the

concept of structure of a matrix, multiplication of structures, and a bipartite graph representation of a structure. Finally, we review the recent work of Hare et al. on the structures of the factors of a QR factorization.

In Chapter 3 we briefly outline previous work done on data structures for sparse QR factorization.

In Chapter 4 we determine the product of the structures of a sequence of Givens transformations. We present an ordering for applying Givens transformations to a sparse matrix in order to compute a QR factorization and, finally, we use the results of Hare et al. to show that if this ordering is used, then the product of the structures of the resulting sequence of Givens transformations is identical to the structure of Q. We also show that the structure for R obtained in this way is correct.

We summarize our results in Chapter 5.

Chapter 2

Preliminary Theory

In this chapter we briefly review the theory of orthogonal factorization, its application to least squares problems, and the use of Givens transformations. A more detailed treatment may be found in [16, Chapter 3].

In Section 2.3 we define what is meant by the structure of a matrix and discuss some of the problems that arise in the computation of the structures of the factors during a QR factorization. Having done that, we are able to state precisely the purpose of this thesis.

In Section 2.4 we describe a bipartite graph model of the structures of the factors Q and R of a sparse matrix, and review the recent work of Hare et al [11] that characterizes these structures. These ideas will be used extensively in Chapter 4.

2.1 The QR Factorization

If $m \geq n$ and A is an $m \times n$ matrix over the complex field, then there is an $m \times m$ unitary matrix $\hat{Q}$ and an $m \times n$ upper trapezoidal matrix $\hat{R}$ such that

$$A = \hat{Q}\hat{R}$$

This is called a QR factorization of A. A proof of its existence may be found in Watkins [16]. If $\text{rank}(A) = n$, then the diagonal entries of R may be chosen to be positive, and in this case, the factorization is unique.

We can rewrite a QR factorization of A given by $A = \hat{Q}\hat{R}$ as

$$A = \begin{pmatrix} Q & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix},$$

where

$$\begin{aligned} Q &= (q_1, q_2, \ldots, q_n), \\ Q_2 &= (q_{n+1}, q_{n+2}, \ldots, q_m), \\ q_i &= \text{the } i^{th} \text{ column of } \hat{Q}, \end{aligned}$$

and R is $n \times n$. Thus

$$A = QR$$

where Q is an $m \times n$ matrix with orthonormal columns and R is an $n \times n$ upper triangular matrix. This is an alternate formulation of the QR factorization of a matrix.

There are a number of ways to compute a QR factorization of a matrix. The classical Gram-Schmidt orthogonalization is one. The Modified Gram-Schmidt algorithm is a more stable method. Another is the reduction of

A to upper trapezoidal form by Householder transformations, and it is this method that is generally preferred for full matrices. Still another method is the reduction of A to upper trapezoidal form by Givens transformations, and this method is frequently used for sparse matrices.

The QR factorization is usually used to solve least squares problems and to compute eigenvalues, and may also be used to solve square nonsingular systems of linear equations. In the latter case, where it is desired to solve $Ax = b$ for x, we replace A by its factors to get $\hat{Q}\hat{R}x = b$. Since $\hat{Q}$ is unitary, $\hat{Q}^{-1} = \hat{Q}^*$, where $\hat{Q}^*$ is the conjugate transpose of $\hat{Q}$, so that

$$\hat{R}x = \hat{Q}^*b,$$

which can be solved for x by back-substitution. This is generally a stable method of solution, although the classical Gram-Schmidt algorithm may be unstable if A is ill-conditioned. The Modified Gram-Schmidt, Householder and Givens methods are stable [16].

In the case of a rectangular $m \times n$ matrix A with $m \geq n$, there may be no solution to the problem

$$Ax = b,$$

but it may be desired to find the vector x for which

$$\| Ax - b \|_2$$

is minimized. This is the least-squares problem. The vector $r = Ax - b$ is called the *residual* and $\| r \|_2$ is the Euclidean length of r defined by $(r^*r)^{1/2}$. Since $\hat{Q}$ is unitary,

$$\| \hat{Q}^*(Ax - b) \|_2^2 = [\hat{Q}^*(Ax - b)]^*[\hat{Q}^*(Ax - b)]$$

$$\begin{aligned} &= (Ax-b)^*\hat{Q}\hat{Q}^*(Ax-b) \\ &= (Ax-b)^*I_m(Ax-b) \\ &= \| Ax-b \|_2^2 \end{aligned}$$

where I_m is the identity matrix of order m. Thus, multiplication by $\hat{Q}^*$ does not change the Euclidean length of r so that the solution of $\min_x \| Ax-b \|_2$ is equal to the solution of $\min_x \| \hat{R}x - \hat{Q}^*b \|_2$. We partition $\hat{R}x - \hat{Q}^*b$ as

$$\begin{pmatrix} R \\ 0 \end{pmatrix} x - \begin{pmatrix} Q^* \\ Q_2^* \end{pmatrix} b = \begin{pmatrix} Rx - Q^*b \\ -Q_2^*b \end{pmatrix}$$

This clearly has minimal length when $Rx - Q^*b = 0$ or $Rx = Q^*b$.

If A has full rank, then R also has full rank (and thus all its diagonal entries are nonzero) and the square system $Rx = Q^*b$ has a unique solution. Since R is upper triangular, this system is easily solved by back substitution.

If A is rank deficient, then R is singular and either the QR factorization with column pivoting or the singular-value decomposition is used to compute a solution. In the QR factorization with column pivoting, a permutation P of the columns of A is determined so that $AP = \hat{Q}\hat{R}$ where

$$\hat{R} = \begin{pmatrix} R_1 & R_2 \\ 0 & 0 \end{pmatrix}$$

and R_1 is nonsingular and upper triangular. That is, $Ax - b$ is transformed to

$$\begin{pmatrix} R_1 & R_2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \hat{Q}^*b \quad \text{where} \quad y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = P^{\mathsf{T}}x$$

This is equivalent to

$$\begin{pmatrix} R_1 y_1 + R_2 y_2 - Q^* b \\ -Q_2^* b \end{pmatrix}$$

which has minimal length when $R_1 y_1 + R_2 y_2 - Q^* b = 0$. There are an infinite number of solutions to this system and one may be obtained by arbitrarily fixing y_2 and solving the nonsingular system

$$R_1 y_1 = Q^* b - R_2 y_2$$

In the remainder of this thesis we will deal only with full-rank matrices.

Orthogonalization methods for solving least squares problems are often preferred to the classical method of solving the normal equations

$$A^* A x = A^* b$$

because they are more stable. The solution of the normal equations may be subject to serious losses of accuracy if A is ill-conditioned [16]. One reason is that critical information may be lost because of round-off error in the computation of A^*A. Furthermore, the condition number of A^*A is the square of the condition number of A, so that if A is even mildly ill-conditioned, then A^*A can be very badly conditioned. In this case, the solution of the system $A^*Ax = A^*b$ can be very inaccurate. The normal equations have the advantage of using less computer space and time than the QR factorization if m is much greater than n. The better stability performance of the orthogonalization methods motivates the effort to find more efficient data structures and algorithms for implementing the QR factorization for large sparse matrices.

2.2 Givens Transformations

This section describes the use of Givens transformations to compute a QR factorization of an $m \times n$ matrix A with $m \geq n$ and rank $A = n$.

Given a nonzero vector $v = (z_1, z_2)^T$ over the complex field, we define the transformation

$$T = \frac{1}{r} \begin{pmatrix} \overline{z_1} & \overline{z_2} \\ -z_2 & z_1 \end{pmatrix}$$

where $r = \|v\|_2 = (v^* v)^{1/2}$. It is easily verified that T is unitary. Applying the transformation T to v gives

$$Tv = \frac{1}{r} \begin{pmatrix} \overline{z_1} & \overline{z_2} \\ -z_2 & z_1 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \frac{1}{r} \begin{pmatrix} \| v \|_2^2 \\ 0 \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}$$

Suppose that v is the first column of a 2×2 matrix

$$A = \begin{pmatrix} z_1 & z_3 \\ z_2 & z_4 \end{pmatrix}$$

Now

$$TA = \begin{pmatrix} r & (\overline{z_1} z_3 + \overline{z_2} z_4)/r \\ 0 & (-z_2 z_3 + z_1 z_4)/r \end{pmatrix} \equiv R$$

is upper triangular. Thus $A = T^* R$ is a QR factorization, since $T^* = T^{-1}$. T is a *Givens transformation*, or a *plane rotation*, since in the real case, it represents a rotation of the plane by an angle θ where

$$\theta = \arccos(z_1/r)$$

Extending T to an n-dimensional vector space, suppose that the 2×2 principal submatrix in rows and columns j and i of the $n \times n$ matrix A is

$$\begin{pmatrix} z_1 & z_3 \\ z_2 & z_4 \end{pmatrix}$$

Define G_{ij}, with $i > j$, by

$$G_{ij} = \begin{pmatrix} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ & & \overline{z_1}/r & & & & \overline{z_2}/r & & \leftarrow j \\ & & & 1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & 1 & & & \\ & & -z_2/r & & & & z_1/r & & \leftarrow i \\ & & & & & & & 1 & \\ & & & & & & & & \ddots \\ & & & & & & & & 1 \end{pmatrix} \quad (2.1)$$

$$\begin{matrix} & & \uparrow & & \uparrow & \\ & & j & & i & \end{matrix}$$

where the principal submatrix of G_{ij} in rows and columns j and i is the Givens transformation T defined for the vector $(z_1, z_2)^T$, and the rest of the matrix is the identity. The (j, j) entry of A is sometimes called the *pivot* for the Givens transformation and the (i, j) entry is the target entry that is to be set to zero. Clearly G_{ij} is a unitary matrix. Applying G_{ij} to A changes only rows j and i of A, leaving the rest of A untouched. Each of rows j and

i of $G_{ij}A$ is a linear combination of rows j and i of A, and the (i,j) entry of $G_{ij}A$ is 0.

Diagonal pivoting refers to the situation in which all of the pivots for the Givens transformations are diagonal entries. This can be generalized so that the pivot element is not a diagonal entry of A (i.e., *variable pivoting*). If $i > j \neq k, a_{jk} = z_1, a_{ik} = z_2$ and G_{ij} is defined as above, then the (i,k) entry of $G_{ij}A$ is 0, and the (j,k) entry is called the pivot.

A QR factorization of an $n \times n$ matrix A can be obtained by using Givens transformations to reduce A to upper triangular form. Let G_{ij} denote the Givens transformation that sets to 0 the (i,j) entry of the matrix that it multiplies, and consider the sequence of Givens transformations

$$G_{21}, G_{31}, \quad , G_{n1}, G_{32}, G_{42}, \quad , G_{n2}, \quad , G_{n,n-1}$$

that produces zeros sequentially in the below-diagonal positions of the matrix A. The transformations can actually be applied in any order that does not destroy previously created zeros. For the sequence above, we obtain

$$G_{n,n-1} \cdots G_{31}G_{21}A = R$$

where R is upper triangular. Since each of the transformations G_{ij} is unitary,

$$A = G_{21}^* G_{31}^* \cdots G_{n,n-1}^* R,$$

and

$$Q = G_{21}^* G_{31}^* \cdots G_{n,n-1}^*$$

is the required unitary factor.

This result can be extended to the rectangular case where A is $m \times n$ and $m > n$. We apply Givens transformations as before to produce

$$A = G_{21}^* G_{31}^* \cdots G_{m1}^* G_{32}^* \cdots G_{m,n}^* \hat{R}$$

where

$$\hat{Q} = G_{21}^* G_{31}^* \cdots G_{m1}^* G_{32}^* \cdots G_{m\,n}^*$$

is $m \times m$ and $\hat{R}$ is $m \times n$. As discussed in Section 2.1, we take Q to be the first n columns of $\hat{Q}$, and R to be the upper n rows of $\hat{R}$ to get the QR factorization in the alternate form.

If the matrix R that is obtained from this procedure does not have all diagonal entries positive, we can multiply R by a diagonal matrix D whose diagonal entries have magnitude 1, and are chosen so that $d_{ii} r_{ii}$ is real and positive (since all $r_{ii} \neq 0$ in the full rank case). Then D is unitary, DR is upper triangular with positive diagonal entries and QD^* is the required factor with orthonormal columns.

It should be noted that there is usually no need to multiply the Givens transformations together to obtain the matrix Q. Normally, it is sufficient to store Q in factored form. Golub and van Loan [10, Section 3.4] give an efficient storage method in which each Givens transformation may be represented by a single parameter, so that the amount of storage required for Q is equal to the number of Givens rotations that must be applied.

2.3 The Algebra of Structures

When working with sparse matrices, the first step is a 'symbolic' phase in which the locations of the nonzero entries in the computed solution are determined so that suitable storage may be allocated. The set of locations corresponding to the nonzero entries is called the *structure* of a sparse matrix $A = (a_{ij})$, which we define by

$$structure(A) = \{(i,j) \mid a_{ij} \neq 0\}$$

We define the *product of the structures* of an $m \times n$ matrix A and an $n \times p$ matrix B as

$$\begin{aligned} structure(A)structure(B) \;=\; & \{(i,j) \mid 1 \leq i \leq m, 1 \leq j \leq p, \\ & \text{and there is some } k\ 1 \leq k \leq n, \text{ such that} \\ & (i,k) \in structure(A) \text{ and } (k,j) \in structure(B)\} \end{aligned}$$

The *transpose* of $structure(A)$ is defined by

$$(structure(A))^* = structure(A^*)$$

The following lemmas follow readily from the definitions, and we state them without proof.

Lemma 2.1 *Multiplication of structures is associative.*

Lemma 2.2 *If*

$$\mathbf{S} = \prod_{t=1}^{k} structure(G^*_{i_t j_t})$$

then

$$\mathbf{S}^* = \prod_{t=k}^{1} structure(G_{i_t j_t})$$

Lemma 2 3 *Let A be an $m \times n$ matrix and B be an $n \times p$ matrix Then*

$$structure(AB) \subseteq structure(A)structure(B)$$

An example suffices to show that it is not necessarily true that

$$structure(A)structure(B) \subseteq structure(AB)$$

Consider

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} -1 & 0 \\ 1 & 1 \end{pmatrix} \qquad \text{and} \qquad AB = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

Now $(1,1) \notin structure(AB)$ but $(1,1) \in structure(A)structure(B)$

This phenomenon, in which an element of $structure(A)structure(B)$ is not an element of $structure(AB)$, is called *cancellation* Its occurrence depends on the precise values of the nonzero entries of the matrices For instance, if the nonzero values in the preceding example are changed to

$$A = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \text{ and } B = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \text{ then } AB = \begin{pmatrix} 3 & 2 \\ 1 & 1 \end{pmatrix}$$

and cancellation does not occur

However, in the computation of a QR factorization of a matrix A by Givens transformations, for some orderings of the transformations cancellation must occur regardless of the numerical values of the nonzero entries of A (assuming a fixed structure for A)

Example 2 1 Let

$$\begin{pmatrix} \star & 0 & 0 & 0 \\ 0 & \star & \star & 0 \\ \star & 0 & \star & 0 \\ \star & 0 & 0 & \star \end{pmatrix} \tag{2 2}$$

represent $structure(A) = \{(1,1),(3,1),(4,1),(2,2),(2,3),(3,3),(4,4)\}$ There are two ways in which any matrix A with this structure can be reduced to triangular form by Givens transformations (using diagonal pivoting) One way is to apply the rotation G_{31} before G_{41}, in which case the $(4,3)$ entry will become nonzero (that is, there is fill at the $(4,3)$ position) and thus G_{43} must be applied as well The resultant factorization is

$$G_{43}G_{41}G_{31}A = R$$

or

$$A = G_{31}^{*}G_{41}^{*}G_{43}^{*}R \tag{2 3}$$

On the other hand, if G_{41} is applied before G_{31}, then there is no fill in positions of A below the main diagonal and the resultant factorization is

$$\tilde{G}_{31}\tilde{G}_{41}A = R$$

or

$$A = \tilde{G}_{41}^{*}\tilde{G}_{31}^{*}R \tag{2 4}$$

(Note that the matrix G_{31} in (2 3) is not equal to $\tilde{G}_{31}$ in (2 4), although their structures are the same This is also true for the matrices G_{41} and $\tilde{G}_{41}$)

Both of (2 3) and (2 4) determine the unique QR factorization of any full rank matrix A having the structure in (2 2) (if the diagonal entries of R are

normalized), that is,

$$Q = G_{31}^* G_{41}^* G_{43}^* = \tilde{G}_{41}^* \tilde{G}_{31}^*$$

However,

$$structure(G_{31}^*)structure(G_{41}^*)structure(G_{43}^*) = \begin{pmatrix} \star & 0 & \star & \star \\ 0 & \star & 0 & 0 \\ \star & 0 & \star & \star \\ \star & 0 & \star & \star \end{pmatrix} \equiv \mathbf{S}_1$$

whereas

$$structure(\tilde{G}_{41}^*)structure(\tilde{G}_{31}^*) = \begin{pmatrix} \star & 0 & \star & \star \\ 0 & \star & 0 & 0 \\ \star & 0 & \star & 0 \\ \star & 0 & \star & \star \end{pmatrix} \equiv \mathbf{S}_2$$

Using results of Hare et al [11], it can be shown that for any full rank matrix A having the structure in (2 2), if $A = QR$ then $structure(Q) \subseteq \mathbf{S}_2$ Moreover, there are such matrices A for which $structure(Q) = \mathbf{S}_2$ Note, in particular, that $q_{34} = 0$ for all matrices Q of such a QR factorization Thus the $(3,4)$ entry of $G_{31}^* G_{41}^* G_{43}^*$ in (2 3) must be 0, so that cancellation occurs in this product regardless of the actual values of the nonzero entries of A

□

The significance of Example 2 1 is that the product of the structures of the rotations in the QR factorization (2 4) can be used to determine the "best" structure for Q, whereas the analogous product using the QR factorization

(2 3) produces a structure that is too large We call an ordering for the Givens transformations a *tight ordering* if the product of their structures gives this "best" structure for Q It is the purpose of this thesis to demonstrate a tight ordering for the application of the Givens rotations A numeric computation using such an ordering will limit the storage required by the QR factorization to that determined in [11]

2.4 Bipartite Graph Representation

In this thesis we use the terminology and notation that appears in *Graph Theory with Applications* by J A Bondy and U S R Murty [1]

Recall the definition of the structure of an $m \times n$ matrix A $structure(A) = \{(i,j) \mid a_{ij} \neq 0 \text{ for } 1 \leq i \leq m \text{ and } 1 \leq j \leq n\}$ A bipartite graph corresponding to $\mathbf{A} \equiv structure(A)$ is defined as $H(\mathbf{A}) = (R(\mathbf{A}), C(\mathbf{A}), E(\mathbf{A}))$ where $R(\mathbf{A})$ is the vertex set $\{r_i = i \mid 1 \leq i \leq m\}$ corresponding to the rows of the matrix A, $C(\mathbf{A})$ is the vertex set $\{c_j = j \mid 1 \leq j \leq n\}$ corresponding to the columns of the matrix A, and $E(\mathbf{A})$ is the set of edges $r_i c_j$ such that $r_i c_j \in E(\mathbf{A})$ if and only if $a_{ij} \neq 0$ Note that $r_i c_j$ and $c_j r_i$ denote the same edge If $a_{ij} \neq 0$, then r_i and c_j are said to be *adjacent*

A *walk* in $H(\mathbf{A})$ is a finite sequence of vertices and edges

$$W = v_0 e_1 v_1 e_2 v_2 \ldots e_k v_k$$

where $k \geq 1$, and for $1 \leq i \leq k$, the $v_i \in R(\mathbf{A})$ or $C(\mathbf{A})$, the $e_i \in E(\mathbf{A})$ and the ends of e_i are v_{i-1} and v_i, and $v_{i-1} \in R(\mathbf{A})$ if and only if $v_i \in C(\mathbf{A})$ W is called a walk *from* v_0 *to* v_k, or a (v_0, v_k)-*walk* This may be expressed

more compactly as $W = v_0v_1v_2 \ldots v_k$, where it is understood that $v_{i-1}v_i$, $i = 1, \ldots, k$, is an edge in $E(\mathbf{A})$. If there is a (v_i, v_j)-walk, then v_j is said to be *reachable* from v_i.

A walk $W = v_0v_1 \ldots v_k$ in $H(\mathbf{A})$ is called a *path* if the vertices $v_0, v_1, \ldots, v_k$ are distinct. Note that if there is a (u, v)-walk in $H(\mathbf{A})$, then there is also a (u, v)-path in $H(\mathbf{A})$. In addition, if there is a (u, v)-path, then there is also a (v, u)-path.

If W_1 is a (u, v)-path and W_2 is a (v, w)-path, then W_1W_2 represents a (u, w)-walk in $H(\mathbf{A})$. If W_3 is a (y, z)-path and if the edge vy exists in $E(\mathbf{A})$, then W_1W_3 is a (u, z)-walk in $H(\mathbf{A})$.

The number of vertices in a set S is called the *cardinality* of S and is denoted by $|S|$. A bipartite graph with $|\, C(\mathbf{A})\,| \leq |\, R(\mathbf{A})\,|$ is said to have the *Hall property* (with respect to $C(\mathbf{A})$) if every subset S of $C(\mathbf{A})$ is adjacent to at least $|S|$ vertices in $R(\mathbf{A})$. It is said to have the *strong Hall property* [11] if S is adjacent to more than $|S|$ vertices in $R(\mathbf{A})$ for all subsets S of $C(\mathbf{A})$ such that

(i) $1 \leq |S| \leq |C(\mathbf{A})| - 1$ if $|C(\mathbf{A})| = |R(\mathbf{A})| > 1$, or

(ii) $1 \leq |S| \leq |C(\mathbf{A})|$ if $|C(\mathbf{A})| < |R(\mathbf{A})|$

Analogously, a matrix A with m rows and n columns, $m \geq n$, has the Hall property if every set of k columns, $1 \leq k \leq n$, has nonzeros in at least k rows, and has the strong Hall property if

(i) $m = n > 1$ and every set of k columns, $1 \leq k < n$, has nonzeros in more than k rows, or

(ii) $m > n$ and every set of k columns, $1 \leq k \leq n$, has nonzeros in more than k rows.

An $m \times n$ matrix with $m \geq n$ must have the Hall property if it has full rank.

Given an $m \times n$ matrix A with the Hall property, let $\mathbf{A} = structure(A)$. If $(i,j) \in \mathbf{A}$ we write $\mathbf{a}_{ij} = \star$ and if $(i,j) \notin \mathbf{A}$ we write $\mathbf{a}_{ij} = 0$. $\mathbf{A}$ represents the structure of all matrices with nonzeros in precisely the same positions in which A has nonzeros. Thus $\mathbf{A}$ determines a set of matrices $\alpha = \{B \mid B \text{ is } m \times n \text{ and } structure(B) = \mathbf{A}\}$. Now consider the QR factorization of each full rank matrix $B \in \alpha$ and define

$$\mathbf{Q} \equiv \bigcup structure(Q)$$

and

$$\mathbf{R} \equiv \bigcup structure(R),$$

where the unions are over all matrices Q and R, respectively, such that $B = QR$, $B \in \alpha$ and rank $B = n$. The structures $\mathbf{Q}$ and $\mathbf{R}$ are the smallest possible that can be guaranteed to accommodate all the nonzeros of the factors Q and R, respectively, of any matrix with structure equal to $\mathbf{A}$. We write $\mathbf{q}_{ij} = \star$ if $(i,j) \in \mathbf{Q}$, $\mathbf{q}_{ij} = 0$ if $(i,j) \notin \mathbf{Q}$ and similarly for $\mathbf{R}$. Hare et al. [11] have determined a characterization of the structures $\mathbf{Q}$ and $\mathbf{R}$. We now summarize a number of concepts that they introduced and that are of importance here.

Let A be an $m \times n$ matrix with the Hall property, and let $H(\mathbf{A})$ be the bipartite graph associated with A.

- By a *Hall set* of $\mathbf{A}$ (or A), we mean a subset S of $C(\mathbf{A})$ such that the

corresponding columns of A have nonzeros in exactly $|S|$ rows of A. It is clear that the union of two Hall sets of $\mathbf{A}$ is itself a Hall set.

- Let $\mathbf{A}_j$ be the structure of the submatrix formed by the first j columns of A.

- Let S_j be the (possibly empty) Hall set of maximum cardinality in $\mathbf{A}_j$, and define $S_0 = \emptyset$.

- Let s_j be the subset of $R(\mathbf{A})$ of all vertices adjacent to vertices in S_j, and define $s_0 = \emptyset$. Note that $S_j = s_j, 1 \leq j \leq n-1$, if $a_{ii} = \star$ for $1 \leq i \leq n$ (since $c_i = i$ and $r_i = i$).

- For $1 \leq j \leq n$, define the auxiliary bipartite graph $B_j(\mathbf{A}) = (R_j(\mathbf{A}), C_j(\mathbf{A}), E_j(\mathbf{A}))$ to be the bipartite graph of $\mathbf{A}_j$, with the sets s_{j-1} and S_{j-1} removed. That is, $C_j(\mathbf{A}) \equiv \{c_k = k \mid 1 \leq k \leq j,\ c_k \notin S_{j-1}\}$, $R_j(\mathbf{A}) \equiv \{r_l = l \mid 1 \leq l \leq m,\ r_l \notin s_{j-1}$, and there exists $c_v \in C_j(\mathbf{A})$ such that $\mathbf{a}_{lv} \neq 0\}$, and there is an edge between $r_i \in R_j(\mathbf{A})$ and $c_k \in C_j(\mathbf{A})$ if and only if $\mathbf{a}_{ik} \neq 0$.

- Let

  t_j denote the subset of $R(\mathbf{A})$ not in $R_j(\mathbf{A})$ or s_{j-1},

  u_j denote the subset of $R_j(\mathbf{A})$ of vertices that are not reachable from c_j,

  p_j denote the subset of $R_j(\mathbf{A})$ of vertices that are reachable from c_j.

  Then t_j, u_j, p_j and s_{j-1} form a partition of $R(\mathbf{A})$, while u_j and p_j form a partition of $R_j(\mathbf{A})$, where $1 \leq j \leq n$.

The main result of [11] is the following, which characterizes Q in terms of path conditions in $B_j(\mathbf{A}), 1 \leq j \leq n$.

Theorem 2.4 [11, Theorem 4.7] *Let $1 \leq i \leq m$ and $1 \leq j \leq n$. Then $(i,j) \in \mathbf{Q}$ if and only if $r_i \in p_j$.*

We also need the following result.

Theorem 2.5 [11, Theorem 5.1] *Let $\mathbf{A}$, $\mathbf{Q}$ and $\mathbf{R}$ be as defined above. Then $\mathbf{R}$ is identical to the upper trapezoidal part of $\mathbf{Q}^*\mathbf{A}$.*

We end this section with an example to illustrate these concepts.

Example 2.2 Let

$$\mathbf{A} = \begin{pmatrix} \star & 0 & 0 & 0 \\ \star & \star & 0 & \star \\ 0 & \star & \star & \star \\ 0 & 0 & 0 & \star \\ \star & 0 & 0 & 0 \\ 0 & 0 & 0 & \star \end{pmatrix}$$

The bipartite graph $H(\mathbf{A})$ is shown in Figure 2.1.

The Hall sets S_j of $\mathbf{A}$ are

$$\begin{aligned} S_0 &= S_1 = S_2 = \emptyset \\ S_3 &= \{c_2, c_3\} \\ S_4 &= S_3 \end{aligned}$$

so that

$$s_0 = s_1 = s_2 = \emptyset$$

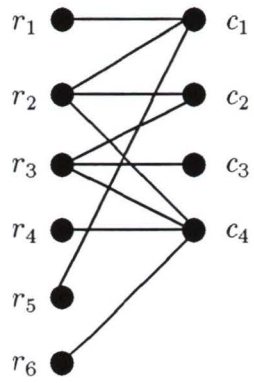


Figure 2.1: Bipartite graph $H(\mathbf{A})$ for the structure $\mathbf{A}$ in Example 2.2

$$s_3 = \{r_2, r_3\}$$

$$s_4 = s_3$$

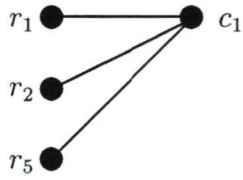
The bipartite graphs $B_1(\mathbf{A})$, $B_2(\mathbf{A})$, $B_3(\mathbf{A})$ and $B_4(\mathbf{A})$ are shown in Figure 2.2. The structures $\mathbf{Q}$ and $\mathbf{R}$ are given by

$$\mathbf{Q} = \begin{pmatrix} \star & \star & \star & 0 \\ \star & \star & \star & 0 \\ 0 & \star & \star & 0 \\ 0 & 0 & 0 & \star \\ \star & \star & \star & 0 \\ 0 & 0 & 0 & \star \end{pmatrix}$$

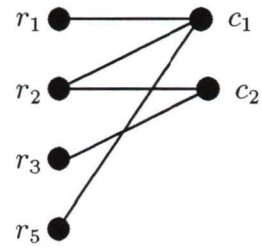
and

$$\mathbf{R} = \begin{pmatrix} \star & \star & 0 & \star \\ 0 & \star & \star & \star \\ 0 & 0 & \star & \star \\ 0 & 0 & 0 & \star \end{pmatrix}$$

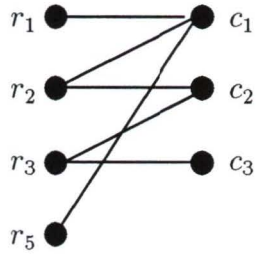
□



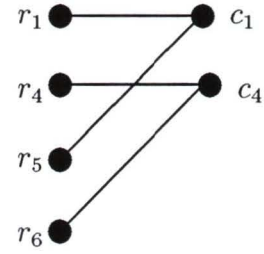
$B_1(\mathbf{A})$
$t_1 = \{r_3, r_4, r_6\}$
$p_1 = \{r_1, r_2, r_5\}$
$u_1 = \emptyset$



$B_2(\mathbf{A})$
$t_2 = \{r_4, r_6\}$
$p_2 = \{r_1, r_2, r_3, r_5\}$
$u_2 = \emptyset$



$B_3(\mathbf{A})$
$t_3 = \{r_4, r_6\}$
$p_3 = \{r_1, r_2, r_3, r_5\}$
$u_3 = \emptyset$



$B_4(\mathbf{A})$
$t_4 = \emptyset$
$p_4 = \{r_4, r_6\}$
$u_4 = \{r_1, r_5\}$

Figure 2.2: Auxiliary bipartite graphs $B_j(\mathbf{A})$ for $\mathbf{A}$ in Example 2.2

Chapter 3

Previous Work on Sparse Matrix Data Structures

Much of the early work on data structures for sparse matrices was done for positive definite systems using Cholesky factorization (which is a variant of the well-known Gaussian elimination algorithm). See, for example, George and Liu [5]. In such problems, it is desired to solve $Ax = b$, where A is positive definite, by factoring

$$A = LL^*,$$

where L is a lower triangular matrix. The solution x is determined by solving

$$Ly = b$$

by forward substitution and then solving

$$L^*x = y$$

by back substitution. Since pivoting is not required to maintain stability in this case, the emphasis was on finding a permutation matrix P such that in the Cholesky factorization

$$P^*AP = LL^*,$$

L is as sparse as possible. It has been found that a suitable choice of P can have a dramatic effect on the cost of the factorization and subsequent solving for x.

Unfortunately, the problem of finding the optimal ordering P for maintaining sparsity is a difficult problem that might require more computation than does solving the original problem. See, for example, [5, p. 115] and [17]. Therefore, heuristic methods are employed that do an adequate job of maintaining sparsity without taking a lot of computer time or space. This has resulted in the development of many different strategies, each of which performs well for some circumstances, but perhaps not for others.

If A is $m \times n$ and $A = QR$, then

$$A^*A = R^*Q^*QR = R^*R,$$

so if A has full rank n, then A^*A is positive definite and the factor R^* is (by uniqueness of the Cholesky factorization) identical to the Cholesky factor of A^*A. Thus the results on preserving sparsity in positive definite linear systems can be applied to the factor R of the QR factorization.

Much of the work on sparse QR factorization has been done in the context of least squares problems where it is possible to reformulate the problem using column interchanges. If A is $m \times n$, P is an $n \times n$ permutation matrix, Q is

an $m \times m$ unitary matrix and $AP = QR$, then

$$\begin{aligned} \|Ax - b\|_2 &= \|APP^{\mathsf{T}}x - b\|_2 \\ &= \|(AP)y - b\|_2, \text{ where } y \equiv P^{\mathsf{T}}x \\ &= \|Q^*(AP)y - Q^*b\|_2 \\ &= \|Ry - Q^*b\|_2 \end{aligned}$$

As

$$\min_x \|Ax - b\|_2 = \min_y \|Ry - Q^*b\|_2$$

and R is upper trapezoidal, the least squares solution y is easily determined (see Section 2.1), and then $x = Py$.

Applying the results of the last two paragraphs to a least squares problem

$$\min_x \|Ax - b\|_2$$

where A is a large sparse matrix with full rank, we want to determine a permutation matrix P_c such that in the Cholesky factorization

$$P_c^{\mathsf{T}} A^* A P_c = R^* R,$$

the matrix R is as sparse as possible. The heuristic strategies developed for large sparse positive definite systems may be applied in order to find a suitable permutation matrix P_c, and to determine the structure of R.

Reordering the rows of the matrix A that is to be factored is equivalent to multiplying A by a permutation matrix P_r, so that if

$$P_r A = QR$$

is the QR factorization of $P_r A$, then

$$A = (P_r^T Q)R$$

is the QR factorization of A since $P_r^T Q$ has orthonormal columns. Thus we see that this reordering does not affect R, and affects Q only by a reordering of its rows. However, Example 2.1 shows that the number of Givens transformations that must be applied depends on the order in which they are applied, and this order may be linked to the ordering of the rows of the matrix (as, for instance, in many implementations where the nonzeros are eliminated sequentially row by row using diagonal pivoting [4]). That example also demonstrates how the intermediate fill may vary depending on the ordering of the Givens transformations.

A good overview of ordering strategies appears in Duff [3]. George, Liu and Ng [7], Liu [12], and Osterby and Zlatev [13] are other references. Many of the strategies are variations on the minimum degree algorithm [6]. Duff [3] and Osterby and Zlatev [13], using dynamic data structures, recommend reordering the rows and columns during the numerical factorization, choosing the next pivot at a particular stage based on criteria dependent on the structure that has been computed at that point. This strategy attempts to reduce the number of Givens transformations that are required as well as to maintain sparsity in the factor R.

George et al. [5, 4, 7, 8] and Liu [12] deal with static data structures. Their strategy for orthogonalization problems is to first apply a column permutation to maintain sparsity in R using methods developed for Cholesky factorization, and then to try to find "good" row orderings based on this

column ordering. After a static data structure for R is created, the QR factorization using Givens transformations proceeds row by row. In addition to the space required for A and R, one vector of length n is required for intermediate results. Ostrouchov [14] adapts the row ordering strategies of Duff in a symbolic factorization algorithm to create a static data structure for R and to determine row orderings suitable for use with the Givens reduction method of George and Heath [4]. Most of these methods use diagonal pivoting, but Liu's method [12] uses a form of variable pivoting.

Less work has been done on computing a data structure for Q since, in least squares applications, the computation may be arranged so that there is no need to store Q. However, there are applications in which Q is required [16, p. 296]; therefore, it is of interest to know how to compute the structure of Q. George et al. [8, 9] have determined a data structure for Q suitable for QR factorization using Householder transformations, where Q is stored in factored form. This data structure can also be used with Givens transformations, but in this case the structure may be larger than necessary.

The accuracy with which the structures of the factors Q and R are determined is also a matter of interest. Coleman et al. [2] have shown that if A has the strong Hall property, then the structure for R that is determined using the methods developed for Cholesky factorization of A^*A is equal to $\mathbf{R}$. However, if A does not have the strong Hall property, then the structure for R computed by this method may be too large. They also proved that if a matrix A has the strong Hall property, and if a sequence of Givens transformations $\{G_i\}$ is applied to reduce it to upper trapezoidal form using diagonal pivoting and eliminating nonzeros row by row, then the upper trapezoidal

part of

$$\mathbf{G} \equiv \left(\prod_{i=1}^{t} structure(G_i) \right) structure(A)$$

is equal to $\mathbf{R}$, and this result holds regardless of the row ordering used. Once again, if A does not have the strong Hall property, then the upper trapezoidal part of $\mathbf{G}$ may be larger than $\mathbf{R}$. If a matrix does not have the strong Hall property, it is possible to reorder its rows and columns so that the permuted matrix is block upper trapezoidal with each diagonal block having the strong Hall property. This is called the Dulmage-Mendelsohn decomposition (see [2]). A least-squares problem can then be decomposed into a set of smaller problems where each of the subproblems involves a matrix with the strong Hall property. However, this procedure may fail if A is rank deficient. For matrices that are ill-conditioned or numerically rank deficient, transforming the matrix into block trapezoidal form may be less desirable than permuting for stability, or for determining the numerical rank, and in this case care must be taken that the computed data structures are not too large.

Hare et al. [11] have developed an algorithm that accurately determines the structures $\mathbf{R}$ and $\mathbf{Q}$ of any matrix A that has the Hall property (but not necessarily the strong Hall property). This algorithm is useful in any situation in which it is not desirable to reorder the matrix into block trapezoidal form. However, these structures $\mathbf{R}$ and $\mathbf{Q}$ are not large enough to accommodate "intermediate fill" that results from an ordering strategy for the Givens transformations such as that of (2,3) in Example 2.1, where the matrix has the Hall property but not the strong Hall property. In this thesis we show that for any matrix having the Hall property, there is a tight ordering for the

Givens transformations so that all computed values can be accommodated in **Q** and **R**. This ordering scheme for the Givens transformations also provides an alternate algorithm (to that given in [11]) for computing the structures **Q** and **R**.

Chapter 4

Determination of R and Q Using Givens Transformations

In this chapter we describe a tight ordering for the Givens transformations. In Section 4.1 we characterize the product of the structures of an arbitrary sequence of Givens transformations. In Section 4.2 we present an algorithm for computing a symbolic QR factorization of a structure $\mathbf{A}$ with the Hall property. This algorithm imposes an ordering on the Givens transformations used to reduce a matrix A with $structure(A) = \mathbf{A}$ to upper trapezoidal form. In Section 4.3 we prove that the reduction using this ordering may be carried out within the structures $\mathbf{Q}$ and $\mathbf{R}$. That is, we show that the appropriate product of the structures of the Givens transformations applied by the algorithm is equal to $\mathbf{Q}$ and that the resulting structure for the upper triangular factor is equal to $\mathbf{R}$. Thus the algorithm produces a tight ordering for the Givens transformations.

4.1 Products of Structures of Givens Transformations

If a sequence of Givens transformations $G_1, G_2, \ldots, G_k$ is applied to a matrix A in order to reduce it to upper trapezoidal form, how can we determine

$$\prod_{i=1}^{k} structure(G_i) \ ?$$

To answer this question, let A be an $m \times n$ matrix and consider an $m \times m$ Givens transformation G_{ij} that produces a zero at the (i, j) position in A. Then $G_{ij} = (g_{kl})$ where

$$\begin{aligned} g_{kl} &\neq 0 \text{ if } k = l, \\ g_{ij} &\neq 0, \\ g_{ji} &\neq 0, \\ g_{kl} &= 0 \text{ otherwise} \end{aligned}$$

We restate this in terms of structures as

$$\begin{aligned} (k,l) &\in structure(G_{ij}) \text{ if } k = l, \\ (i,j) &\in structure(G_{ij}), \\ (j,i) &\in structure(G_{ij}), \\ (k,l) &\notin structure(G_{ij}) \text{ otherwise} \end{aligned}$$

In the following lemmas, we consider the product of the structures of $m \times m$ Givens transformations

Lemma 4 1 *Let i, j, k and l be distinct and let*

$$\mathbf{H} = structure(G_{ij})structure(G_{kl})$$

Then $\mathbf{H} = \{(i,j),(j,i),(k,l),(l,k)\} \bigcup \{(r,r) \mid 1 \leq r \leq m\}$

Proof This result follows immediately from the definition of the product of structures □

Lemma 4 2 *Let i, j, and k be distinct, and let* **H** *be any one of*

$structure(G_{ij})structure(G_{ik})$,

$structure(G_{ji})structure(G_{ki})$,

$structure(G_{ji})structure(G_{ik})$ *or*

$structure(G_{ij})structure(G_{ki})$

Then

$$\mathbf{H} = \{(i,j),(j,i),(i,k),(k,i),(j,k)\} \bigcup \{(r,r) \mid 1 \leq r \leq m\}$$

Proof In the first case, $(j,k) \in \mathbf{H}$ since $(j,i) \in structure(G_{ij})$ and $(i,k) \in structure(G_{ik})$ For this case, the rest of **H** is clear from the definitions The other three cases follow similarly □

We note that $(k,j) \notin \mathbf{H}$ in Lemma 4 2 since (k,k) is the only ordered pair in $structure(G_{ij})$ or $structure(G_{ji})$ that has k as its first element and (j,j) is the only ordered pair in $structure(G_{ik})$ or $structure(G_{ki})$ that has j as its second element

If A is reduced to upper trapezoidal form by a sequence of Givens transformations

$$G_{i_1 j_1}, G_{i_2 j_2}, \ldots, G_{i_r j_r}$$

then

$$G_{i_r j_r} \cdots G_{i_2 j_2} G_{i_1 j_1} A = R$$

and

$$Q = G^*_{i_1 j_1} G^*_{i_2 j_2} \cdots G^*_{i_r j_r}$$

In the following result, the structure of an arbitrary product of Givens transformations is specified

Lemma 4 3 *If* $r \geq 1$ *and* $Q = G^*_{i_1 j_1} G^*_{i_2 j_2} \cdots G^*_{i_r j_r}$, *then*

$$structure(Q) \subseteq \prod_{t=1}^{r} structure(G^*_{i_t j_t}) \equiv \mathbf{G}, \tag{4 1}$$

and $(a, b) \in \mathbf{G}$ *if and only if*

- $a = b$, *or*
- *there exists* s *such that*

  1 $i_s = a$ *and* $j_s = b$, *or*

  2 $i_s = b$ *and* $j_s = a$

  or

- *there is an ordered subsequence of the applied rotations*

$$G^*_{i_{s_1} j_{s_1}}, G^*_{i_{s_2} j_{s_2}}, \ldots, G^*_{i_{s_k} j_{s_k}}$$

with $2 \leq k \leq r$ *and*

$$s_1 < s_2 < s_3 < \quad < s_k$$

such that $a = \imath_{s_1}$ *or* $a = \jmath_{s_1}$, *and* $b = \imath_{s_k}$ *or* $b = \jmath_{s_k}$, *and (at least) one of*

$$\begin{aligned} \imath_{s_t} &= \imath_{s_{t+1}}, \\ \imath_{s_t} &= \jmath_{s_{t+1}}, \\ \jmath_{s_t} &= \imath_{s_{t+1}}, \\ \jmath_{s_t} &= \jmath_{s_{t+1}} \end{aligned}$$

holds for each $t = 1, 2, \quad , k-1$

Proof Firstly, (4 1) follows from the definition of Q and as *structure*$(G_{\imath\jmath}) =$ *structure*$(G^*_{\imath\jmath})$ for any $\imath, \jmath$ The remainder of the lemma follows using Lemmas 4 1 and 4 2 and induction on k □

4.2 An Algorithm for Symbolic Factorization

In this section we present an algorithm for determining the structures of the upper triangular factor R and the factor Q of a matrix A that has the Hall property (but not necessarily the strong Hall property). We claim that the computed structures are correct in the sense that they are identical to the structures that are determined by the method of Hare et al [11].

Our strategy is to compute a QR factorization of A by applying Givens transformations in a specified order that depends upon $structure(A)$. The ordering is restricted to the case where A has a nonzero diagonal and diagonal pivoting is used. The restriction on the diagonal is reasonable since we can reorder the rows of any matrix with the Hall property so that it has a nonzero diagonal. This reordering does not affect the structure of R and affects the structure of Q only by reordering its rows. Nonzeros below the diagonal are eliminated column by column, and within each column, nonzeros in rows that are not in a set s_k are eliminated before nonzeros in rows in s_k. The resulting structures are

$$\bar{\mathbf{R}} \equiv \left(\prod_{k=t}^{1} structure(G_{i_k j_k}) \right) structure(A) \backslash \{(i,j) \mid i > j\}$$

and

$$\bar{\mathbf{Q}} \equiv \left(\prod_{k=1}^{t} structure(G^{*}_{i_k j_k}) \right) \backslash \{(i,j) \mid j > n\},$$

where the ordering for the Givens transformations is determined by Algorithm 1.

Algorithm 1 *Determine* $\bar{\mathbf{R}}$ *and* $\bar{\mathbf{Q}}$

Input m, n, $\mathbf{A}$, *the sets* $s_1, s_2, \ldots, s_{n-1}$ *(as defined in Section 2.4)*

{ $m \geq n$ *and* $\mathbf{A}$ *is the structure of an* $m \times n$ *matrix with a nonzero diagonal* }

Output $\bar{\mathbf{Q}}, \bar{\mathbf{R}}$

Step 1 *Initialize* $\bar{\mathbf{Q}} = \{(i,i) \mid 1 \leq i \leq m\}$

Step 2 *Initialize* $\hat{\mathbf{A}} = \mathbf{A}$

Step 3 *Iterate for* $j = 1, 2, 3, \ldots, n-1$

{ *Elimination on* c_j }

Initialize $rows \leftarrow \{r_i \mid i > j \text{ and } (i,j) \in \hat{\mathbf{A}}\}$

For $k = n-1, n-2, \ldots, j$

for each $i \in rows \backslash s_k$

set $\bar{\mathbf{Q}} \leftarrow \bar{\mathbf{Q}}\, structure(G_{ij})$

{ G_{ij} *is an* $m \times m$ *Givens transformation that*

produces a zero entry at the (i,j) *position, see (2.1)* }

set $\hat{\mathbf{A}} \leftarrow (structure(G_{ij})\hat{\mathbf{A}}) \backslash \{(i,j)\}$

set $rows \leftarrow rows \backslash \{r_i\}$

Step 4 *For* $j = n$

{ *Elimination on* c_n }

Initialize $rows \leftarrow \{r_i \mid i > j \text{ and } (i,j) \in \hat{\mathbf{A}}\}$

for each $i \in rows$

set $\bar{\mathbf{Q}} \leftarrow \bar{\mathbf{Q}}\, structure(G_{ij})$

set $\hat{\mathbf{A}} \leftarrow (structure(G_{ij})\hat{\mathbf{A}}) \backslash \{(i,j)\}$

set $rows \leftarrow rows \backslash \{r_i\}$

Step 5 *Set* $\bar{\mathbf{R}} = \hat{\mathbf{A}}$

Set $\bar{\mathbf{Q}} = \bar{\mathbf{Q}} \backslash \{(i,j) \mid j > n\}$

Step 6 *Output* $\bar{\mathbf{R}}$ *and* $\bar{\mathbf{Q}}$

Some notation that will be used throughout the rest of this chapter follows. For $1 \leq j \leq n$, let $\hat{\mathbf{A}}_\mathbf{j}$ denote the structure $\hat{\mathbf{A}}$ after elimination on c_j in Algorithm 1 (that is, after the jth iteration of Step 3 or Step 4). By 'fill' at the (i,k) position we mean that $(i,k) \notin \mathbf{A}$ and for some $t \geq 1, (i,k) \in \hat{\mathbf{A}}_\mathbf{t}$. At any fixed point in Algorithm 1 we say that $\hat{\mathbf{a}}_{ij} = \star$ if $(i,j) \in \hat{\mathbf{A}}$ and $\hat{\mathbf{a}}_{ij} = 0$ if $(i,j) \notin \hat{\mathbf{A}}$. (Note that $\hat{\mathbf{A}}$ is a dynamic data structure, so that at different points in Algorithm 1, any $\hat{\mathbf{a}}_{ij}$ may be 0 or $\star$.) Similarly, $\hat{\mathbf{a}}_{ij}^{(t)} = \star$ if $(i,j) \in \hat{\mathbf{A}}_\mathbf{t}$ and $\hat{\mathbf{a}}_{ij}^{(t)} = 0$ if $(i,j) \notin \hat{\mathbf{A}}_\mathbf{t}$.

We illustrate the application of this algorithm to the matrix in Example 2.2. In the array representation of the evolving structure of $\hat{\mathbf{A}}$, we use the symbol f to denote fill, and the symbol $\bigcirc$ to denote an eliminated entry. At the first iteration of Step 3, G_{51} is applied first since $r_5 \notin s_3$ but $r_2 \in s_3$. Thus

$$\hat{\mathbf{A}} \rightarrow \begin{pmatrix} \star & 0 & 0 & 0 \\ \star & \star & 0 & \star \\ 0 & \star & \star & \star \\ 0 & 0 & 0 & \star \\ \bigcirc & 0 & 0 & 0 \\ 0 & 0 & 0 & \star \end{pmatrix} \rightarrow \begin{pmatrix} \star & f & 0 & f \\ \bigcirc & \star & 0 & \star \\ 0 & \star & \star & \star \\ 0 & 0 & 0 & \star \\ \bigcirc & 0 & 0 & 0 \\ 0 & 0 & 0 & \star \end{pmatrix},$$

that is, $\hat{A} = \{(1,1),(1,2),(1,4),(2,2),(2,4),(3,2),\quad\}$

At the same time,

$$\bar{\mathbf{Q}} \rightarrow \begin{pmatrix} \star & 0 & 0 & 0 & \star & 0 \\ 0 & \star & 0 & 0 & 0 & 0 \\ 0 & 0 & \star & 0 & 0 & 0 \\ 0 & 0 & 0 & \star & 0 & 0 \\ \star & 0 & 0 & 0 & \star & 0 \\ 0 & 0 & 0 & 0 & 0 & \star \end{pmatrix} \rightarrow \begin{pmatrix} \star & \star & 0 & 0 & \star & 0 \\ \star & \star & 0 & 0 & 0 & 0 \\ 0 & 0 & \star & 0 & 0 & 0 \\ 0 & 0 & 0 & \star & 0 & 0 \\ \star & \star & 0 & 0 & \star & 0 \\ 0 & 0 & 0 & 0 & 0 & \star \end{pmatrix}$$

In the second iteration of Step 3, only G_{32} is applied so that

$$\hat{\mathbf{A}} \rightarrow \begin{pmatrix} \star & f & 0 & f \\ \odot & \star & f & \star \\ 0 & \odot & \star & \star \\ 0 & 0 & 0 & \star \\ \odot & 0 & 0 & 0 \\ 0 & 0 & 0 & \star \end{pmatrix}$$

and

$$\bar{\mathbf{Q}} \rightarrow \begin{pmatrix} \star & \star & \star & 0 & \star & 0 \\ \star & \star & \star & 0 & 0 & 0 \\ 0 & \star & \star & 0 & 0 & 0 \\ 0 & 0 & 0 & \star & 0 & 0 \\ \star & \star & \star & 0 & \star & 0 \\ 0 & 0 & 0 & 0 & 0 & \star \end{pmatrix}$$

No computation occurs during the third iteration of Step 3, and only G_{64} is

applied at Step 4, so that

$$\hat{\mathbf{A}} \rightarrow \begin{pmatrix} \star & f & 0 & f \\ \bigcirc & \star & f & \star \\ 0 & \bigcirc & \star & \star \\ 0 & 0 & 0 & \star \\ \bigcirc & 0 & 0 & 0 \\ 0 & 0 & 0 & \bigcirc \end{pmatrix}$$

and

$$\bar{\mathbf{Q}} \rightarrow \begin{pmatrix} \star & \star & \star & 0 & \star & 0 \\ \star & \star & \star & 0 & 0 & 0 \\ 0 & \star & \star & 0 & 0 & 0 \\ 0 & 0 & 0 & \star & 0 & \star \\ \star & \star & \star & 0 & \star & 0 \\ 0 & 0 & 0 & \star & 0 & \star \end{pmatrix}$$

Finally, at Step 5, the last two columns of $\bar{\mathbf{Q}}$ are deleted, and the resulting structures returned are

$$\bar{\mathbf{R}} \rightarrow \begin{pmatrix} \star & \star & 0 & \star \\ 0 & \star & \star & \star \\ 0 & 0 & \star & \star \\ 0 & 0 & 0 & \star \end{pmatrix} \quad \text{and} \quad \bar{\mathbf{Q}} \rightarrow \begin{pmatrix} \star & \star & \star & 0 \\ \star & \star & \star & 0 \\ 0 & \star & \star & 0 \\ 0 & 0 & 0 & \star \\ \star & \star & \star & 0 \\ 0 & 0 & 0 & \star \end{pmatrix}$$

4.3 Proof of Correctness for Algorithm 1

In this section we prove that $\bar{\mathbf{Q}} = \mathbf{Q}$ and $\bar{\mathbf{R}} = \mathbf{R}$. Application of Algorithm 1 to a structure $\mathbf{A}$ determines a sequence

$$structure(G_{i_1 j_1}), structure(G_{i_2 j_2}), \quad , structure(G_{i_r j_r})$$

such that if A is any full rank $m \times n$ matrix with $structure(A) = \mathbf{A}$, then

$$A = (G_{i_r j_r} \quad G_{i_2 j_2} G_{i_1 j_1})^* R$$

is the QR factorization of A. (The precise values of the nonzero entries of the matrices G_{ij} depend on the values of the nonzero entries of A, also, as mentioned in Section 2.2, some normalization may be necessary in order that the diagonal entries of R are positive.)

From Lemma 4.3 it follows that if $G_{i_t j_t}$ is applied, and if $i_t \leq n$, then (i_t, j_t) and (j_t, i_t) are both elements of

$$\bar{\mathbf{Q}} \equiv \left(\prod_{k=1}^{t} structure(G^*_{i_k j_k}) \right) \backslash \{(i, j) \mid j > n\}$$

Thus, the first step in proving that $\bar{\mathbf{Q}}$ is identical to $\mathbf{Q}$ is to show that if $G_{i_t j_t}$ is applied, then $(i_t, j_t) \in \mathbf{Q}$ and if $i_t \leq n$, then $(j_t, i_t) \in \mathbf{Q}$. To do this, we will make use of the bipartite graph characterization of $\mathbf{Q}$ described in Section 2.4. A number of lemmas are required. We note that while the next two lemmas are never specifically cited in the succeeding proofs, they will be used implicitly in most of them to determine membership (or nonmembership) of vertices in Hall sets, and thus in the auxiliary bipartite graphs $B_j(\mathbf{A})$.

Lemma 4.4 *Let $m \geq n$ and let A be an $m \times n$ matrix with the Hall property and $a_{ii} \neq 0, 1 \leq i \leq n$. Let $1 \leq j \leq n$ and $1 \leq i \leq j$. Then $c_i \in S_j$ if and only if $r_i \in s_j$.*

Proof The result follows since the diagonal entries $a_{r_i c_i} \equiv a_{ii}$ are nonzero. □

Lemma 4.5 *Let S_j and S_k be Hall sets of a matrix A. If $j \leq k$, then $S_j \subseteq S_k$.*

Proof $S_j \cup S_k$ is a Hall set on the first k columns. But $|S_j \cup S_k| \leq |S_k|$ since S_k is the Hall set of maximum cardinality on the first k columns. Therefore $S_j \subseteq S_k$. □

Lemma 4.6 *Let $2 \leq j \leq n-1$ and $i < j$, and suppose $c_j \in S_k$ for some k such that $j \leq k \leq n-1$. Then during elimination on c_i using Algorithm 1, fill in c_j below r_i is restricted to rows in s_k.*

Proof We use induction on i.

Base Case Elimination on c_1.

We consider 2 cases.

Case 1. $r_1 \in s_k$. Then $c_1 \in S_k$ and hence only rows in s_k are involved in the elimination at this stage so that fill is necessarily restricted to rows in s_k.

Case 2. $r_1 \notin s_k$. If $c_j \in S_k$, then c_j must have a zero in row 1, and in all other rows that are not in s_k. Since nonzeros in all rows not in s_k are eliminated **before** nonzeros in rows in s_k, it follows that $\hat{\mathbf{a}}_{1j} = 0$ until after

elimination on rows in s_k begins. At that point, only rows in s_k remain to be processed, so that fill in c_j below r_1 is restricted to these rows.

Induction Hypothesis Suppose $v < j - 1$ and that during elimination of nonzeros in column $i, 1 \leq i \leq v$, fill below r_i in c_j is restricted to rows in s_k.

Induction Step We now consider elimination of nonzeros from column $v + 1$. The pivot row is row $v + 1$.

Case 1. $r_{v+1} \in s_k$. In this case $c_{v+1} \in S_k$ and by the Induction Hypothesis, c_{v+1} has nonzeros below the diagonal only in rows in s_k. Therefore, fill during this stage of the elimination is restricted to rows in s_k.

Case 2. $r_{v+1} \notin s_k$. In this case $\mathbf{a}_{v+1,j} = 0$ and by the Induction Hypothesis, $\hat{\mathbf{a}}_{v+1,j} = 0$ before elimination on c_{v+1} begins. Furthermore, all nonzeros in c_j below r_v are restricted to rows in s_k before processing of column c_{v+1} begins. The order of elimination guarantees that all the zeros in c_j below r_v are preserved until elimination of rows in s_k begins. At that point, $\hat{\mathbf{a}}_{v+1,j}$ may become $\star$ but fill in c_j below r_{v+1} is restricted to rows in s_k.

Hence, during elimination on c_{v+1} fill in c_j below r_{v+1} is restricted to rows in s_k. □

Corollary 4.6.1 *Let j and k be defined as in Lemma 4.6. Then using Algorithm 1, fill in c_j below r_j is restricted to rows in s_k.*

Lemma 4.7 *Let $1 \leq v \leq n - 1$, $v < j \leq n$, $v < i \leq m$, and $j \leq k \leq n$. If $r_i \notin s_{k-1}$ and fill occurs at the (i, j) position during elimination of the*

nonzero entries of column v below the diagonal in Algorithm 1, then there is a (c_j, r_i)-path in $B_k(\mathbf{A})$

Proof We use induction on v

Base Case Consider elimination of the nonzero entries below the diagonal in column $v = 1$

If fill occurs at the (i, j) position, then $\mathbf{a}_{i1} = \star$ and either

1. $\mathbf{a}_{1j} = \star$, or
2. there exists w such that G_{w1} is applied before G_{i1} , $\mathbf{a}_{w1} = \star$ and $\mathbf{a}_{wj} = \star$

Now $\mathbf{a}_{i1} = \star$ implies that $c_1 \notin S_{k-1}$ and $r_1 \notin s_{k-1}$

1. If $\mathbf{a}_{1j} = \star$, then $c_j \notin S_{k-1}$ and $r_j \notin s_{k-1}$ Therefore $B_k(\mathbf{A})$ contains the path $r_i c_1 r_1 c_j$ since the $(i, 1)$, $(1, 1)$ and $(1, j)$ entries are all nonzero
2. If there exists w such that G_{w1} is applied before G_{i1}, $\mathbf{a}_{w1} = \star$ and $\mathbf{a}_{wj} = \star$, then $r_w \notin s_{k-1}$ (by the order of elimination of entries in Algorithm 1) which implies that $c_j \notin S_{k-1}$ Thus we have the path $r_i c_1 r_w c_j$ in $B_k(\mathbf{A})$

In either case there is a path in $B_k(\mathbf{A})$ from c_j to r_i

Induction Hypothesis Suppose that during elimination on column p where $1 \leq p \leq v$, if fill occurs at the (i, j) position, where $p < i \leq m$, $p < j \leq n$, $j \leq k \leq n$ and $r_i \notin s_{k-1}$, then there is a (c_j, r_i)-path in $B_k(\mathbf{A})$

Induction Step We consider elimination on column $v+1$ and suppose that fill occurs at the (i,j) position, where $v+1 < i \leq m$, $v+1 < j \leq n$, $j \leq k \leq n$ and $r_i \notin s_{k-1}$. For this to happen, we must have $\hat{\mathbf{a}}_{i,v+1}^{(v)} = \star$ which means either $\mathbf{a}_{i,v+1} = \star$ or, by the induction hypothesis, there is a path in $B_k(\mathbf{A})$ from c_{v+1} to r_i. In either case, $c_{v+1} \notin S_{k-1}$ which implies that $r_{v+1} \notin s_{k-1}$. We must also have one of the following two cases.

1. $\hat{\mathbf{a}}_{v+1,j}^{(v)} = \star$, in which case there is a (c_j, r_{v+1})-path in $B_k(\mathbf{A})$. Call this path P_1 and denote by P_2 the (c_{v+1}, r_i)-path in $B_k(\mathbf{A})$. The nonzero diagonal of A implies that the $r_{v+1}c_{v+1}$ edge also exists, so that P_1P_2 is a (c_j, r_i)-walk in $B_k(\mathbf{A})$ and hence there is a (c_j, r_i)-path in $B_k(\mathbf{A})$.

2. There exists w such that $G_{w,v+1}$ is applied before $G_{i,v+1}$, $\hat{\mathbf{a}}_{w,v+1}^{(v)} = \star$ and $\hat{\mathbf{a}}_{wj}^{(v)} = \star$. Since $G_{w,v+1}$ is applied before $G_{i,v+1}$, r_w cannot be in s_{k-1} (by the order of elimination of entries in Algorithm 1). Thus, $\hat{\mathbf{a}}_{w,v+1}^{(v)} = \star$ and the induction hypothesis together imply that there is a (c_{v+1}, r_w)-path in $B_k(\mathbf{A})$. Similarly, $\hat{\mathbf{a}}_{wj}^{(v)} = \star$ implies that $B_k(\mathbf{A})$ contains a (c_j, r_w)-path. Since $B_k(\mathbf{A})$ also contains the (c_{v+1}, r_i)-path, together these paths give a (c_j, r_i)-walk which implies that there is a (c_j, r_i)-path in $B_k(\mathbf{A})$.

In either case there is a (c_j, r_i)-path in $B_k(\mathbf{A})$. □

Corollary 4.7.1 *If, in Algorithm 1, fill occurs at the (i,j) position where $i > j$, then $\mathbf{q}_{ij} = \star$.*

Proof If $i > j$ then $r_i \notin s_{j-1}$. Applying Lemma 4.7 with $k = j$, there

is a (c_j, r_i)-path in $B_j(\mathbf{A})$, that is, $r_i \in p_j$ The result now follows from Theorem 2 4 □

Corollary 4 7 2 *Let* $1 \leq j \leq n$ *and* $j < i \leq m$, *and suppose that* G_{ij} *is applied in Algorithm 1 If there exists* k, *where* $j \leq k \leq n$, *such that* $r_i \notin s_{k-1}$, *then the* (c_j, r_i)*-path exists in* $B_k(\mathbf{A})$

Proof This is clear when $a_{ij} = \star$ since the edge $r_i c_j$ exists in $B_k(\mathbf{A})$, and follows from Lemma 4 7 otherwise □

Corollary 4 7 3 *If* G_{ij} *is applied in Algorithm 1, then* $r_i \in p_j$ *and* $(i, j) \in \mathbf{Q}$

Proof Apply Corollary 4 7 2 with $k = j$ to obtain $r_i \in p_j$ It follows from Theorem 2 4 that $(i, j) \in \mathbf{Q}$ □

Lemma 4 8 *Let* $1 \leq j \leq n$ *and* $i > j$ *If* G_{ij} *is applied in Algorithm 1, then* $r_j \notin s_l$, $l = min\{i - 1, n - 1\}$

Proof Suppose $r_j \in s_l$ Then by Corollary 4 6 1, fill in c_j below the diagonal is restricted to rows in s_l, and since $r_i \notin s_l$ we must have $\hat{\mathbf{a}}_{ij}^{(j-1)} = 0$ But then G_{ij} would not be applied in Algorithm 1 Therefore, if G_{ij} is applied, then $r_j \notin s_l$ □

Lemma 4 9 *Let* $1 \leq j < i \leq n$ *If* G_{ij} *is applied in Algorithm 1, then* $r_j \in p_i$

Proof We know that $r_j \notin s_{i-1}$ by Lemma 4 8 since $i > j$ Application of G_{ij} implies that $\hat{\mathbf{a}}_{ij}^{(j-1)} = \star$ Since $\mathbf{a}_{ii} = \star$ and $\mathbf{a}_{jj} = \star$, $\hat{\mathbf{a}}_{ji}^{(j)} = \star$ Either $\mathbf{a}_{ji} = \star$ with $r_j c_i \in B_i(\mathbf{A})$, or fill occurs at the (j,i) position during elimination on columns $1, \ldots, j-1$, in which case $r_j \in p_i$ by Lemma 4 7 □

Corollary 4 7 3 and Lemma 4 9 together prove that if G_{ij} is applied in Algorithm 1, then $(i,j) \in \mathbf{Q}$ and if $i \leq n$ then $(j,i) \in \mathbf{Q}$ This is the second condition in Lemma 4 3 We must also show that if there is an ordered subsequence of the G_{ij}'s as described in the third condition in Lemma 4 3, then the appropriate entry of **Q** is $\star$ Before doing this, we first examine the ordering permitted by Algorithm 1

We define a *chain* of *length* $k \geq 2$ that links $G_{i_a j_a}$ and $G_{i_b j_b}$ to be an ordered subsequence

$$G_{i_{s_1} j_{s_1}}, G_{i_{s_2} j_{s_2}}, \ldots, G_{i_{s_k} j_{s_k}} \tag{4 2}$$

of the rotations $G_{i_1 j_1}, \ldots, G_{i_r j_r}$ with $s_1 < s_2 < s_3 < \ldots < s_k$ such that $s_1 = a$, $s_k = b$ and one of

(i) $i_{s_t} = i_{s_{t+1}}$,

(ii) $i_{s_t} = j_{s_{t+1}}$,

(iii) $j_{s_t} = j_{s_{t+1}}$, or

(iv) $j_{s_t} = i_{s_{t+1}}$

holds for each $1 \leq t \leq k-1$

We first show that (iv) is not possible in the ordering for the transformations imposed by Algorithm 1

Lemma 4.10 *Let $1 \leq i \leq m$, $1 \leq j \leq n$ and $1 \leq k \leq n$. If G_{ij} and G_{jk} are both applied in Algorithm 1, then G_{jk} must be applied before G_{ij}.*

Proof If G_{ij} is applied, then $i > j$. If G_{jk} is applied, then $j > k$, so that $k < j < i$. Since the columns are processed in increasing order, G_{jk} is applied before G_{ij}. □

Suppose that for a fixed t in (4.2), (i) holds, that is, (4.2) contains an adjacent pair of rotations of the form

$$G_{i_{s_t} j_{s_t}}, G_{i_{s_t} j_{s_t+1}}$$

If (i) also holds for $t+1$, then (4.2) contains

$$G_{i_{s_t} j_{s_t}}, G_{i_{s_t} j_{s_t+1}}, G_{i_{s_t} j_{s_t+2}}$$

and we can eliminate $G_{i_{s_t} j_{s_t+1}}$ from the chain (4.2) to get a chain of length $k-1$ linking $G_{i_a j_a}$ and $G_{i_b j_b}$. Similarly, if (ii) holds for $t+1$, then (4.2) contains

$$G_{i_{s_t} j_{s_t}}, G_{i_{s_t} j_{s_t+1}}, G_{i_{s_t+2} i_{s_t}}$$

Once again, the transformation $G_{i_{s_t} j_{s_t+1}}$ can be eliminated to give a chain of length $k-1$ that still links $G_{i_a j_a}$ and $G_{i_b j_b}$.

Suppose now that (ii) holds for some t, so that (4.2) contains

$$G_{i_{s_t} j_{s_t}}, G_{i_{s_t+1} i_{s_t}}$$

If (iii) holds for $t+1$, then (4.2) contains

$$G_{i_{s_t} j_{s_t}}, G_{i_{s_t+1} i_{s_t}}, G_{i_{s_t+2} i_{s_t}},$$

and again the middle rotation can be eliminated to obtain a chain of length $k-1$

Similarly, if (iii) holds for t and also for $t+1$, then (4 2) contains

$$G_{i_{s_t}j_{s_t}},\ G_{i_{s_{t+1}}j_{s_t}},\ G_{i_{s_{t+2}}j_{s_t}},$$

and the middle rotation can be eliminated to obtain a chain of length $k-1$

If no transformations can be eliminated from a chain in such a manner, the chain is said to have *minimal* length Let us suppose that

$$G_{i_1j_1},\ G_{i_2j_2},\quad ,\ G_{i_kj_k}$$

is a chain of minimal length that links $G_{i_1j_1}$ and $G_{i_kj_k}$, and that the length is ≥ 3 The above results may be summarized as follows The sequences of rotations that are not possible in a chain of minimal length are (i) followed by (i), (i) followed by (ii), (ii) followed by (iii), (iii) followed by (iii), and (iv) That is, if (i) holds for some t, then (iii) must hold for $t+1$ Also, if (iii) holds for some t, then (i) holds for $t-1$ Other possible sequences are (iii) followed by (i), (iii) followed by (ii), (ii) followed by (i), and (ii) followed by (ii)

We now proceed to prove that the appropriate entry of **Q** is $\star$ for every chain of length 2 that is permitted by Algorithm 1 In Lemmas 4 12, 4 13 and 4 14, we consider the cases (i), (ii) and (iii), respectively

Lemma 4 11 *Let $1 \leq j \leq n$ and $j < i \leq m$ If G_{ij} is applied in Algorithm 1, then there is an (r_j, r_i)-path in $B_l(\mathbf{A})$, where $l = \min\{n, i\}$*

Proof By Lemma 4 8, $r_j \notin s_{l-1}$ Also $r_i \notin s_{l-1}$ since $l \leq i$ Applying Corollary 4 7 2, we have a (c_j, r_i)-path in $B_l(\mathbf{A})$, and combining this with the $c_j r_j$ edge, we get an (r_j, r_i)-walk, and hence an (r_j, r_i)-path in $B_l(\mathbf{A})$ □

Lemma 4 12 *Let* $1 \leq i \leq m$, $1 \leq j \leq n$ *and* $1 \leq k \leq n$ *If* G_{ik} *and* G_{ij} *are both applied in Algorithm 1, and if* G_{ik} *is applied before* G_{ij}, *then* $r_k \in p_j$

Proof Clearly $i > k$ and $i > j$ Since G_{ik} is applied before G_{ij}, $k < j$, so we have $k < j < i$, which implies that $r_i \notin s_{j-1}$ By Corollary 4 7 2 there is a (c_k, r_i)-path and also a (c_j, r_i)-path in $B_j(\mathbf{A})$ Combining these two paths with the edge $r_k c_k$ gives a (c_j, r_k)-walk in $B_j(\mathbf{A})$, and thus there is a (c_j, r_k)-path in $B_j(\mathbf{A})$ Therefore, $r_k \in p_j$ □

Lemma 4 13 *Let* $1 \leq i \leq n$, $1 \leq j \leq n$ *and* $1 \leq k \leq n$ *If* G_{jk} *and* G_{ij} *are both applied in Algorithm 1, then* $r_k \in p_i$

Proof From the proof of Lemma 4 10, we know that $k < j < i$ Lemma 4 8 implies that $r_k \notin s_{j-1}$ and $r_j \notin s_{i-1}$ Since G_{jk} is applied in Algorithm 1, there is a (c_k, r_j)-path in $B_i(\mathbf{A})$ by Corollary 4 7 2 Similarly,by Corollary 4 7 2 there is a (c_j, r_i)-path in $B_i(\mathbf{A})$ since $r_i \notin s_{i-1}$ These two paths combined with the edges $r_k c_k$, $r_j c_j$, and $r_i c_i$ make a (c_i, r_k)-walk and hence there is a (c_i, r_k)-path in $B_i(\mathbf{A})$ Thus, $r_k \in p_i$ □

Lemma 4 14 *Let* $1 \leq j \leq n$, $1 \leq i, k \leq m$, *with* $i \neq k$, *and* $l = min\{n, k\}$ *If* G_{ij} *and* G_{kj} *are both applied in Algorithm 1 and if* G_{ij} *is applied before*

G_{kj}, then there is an (r_i, r_k)-path in $B_l(\mathbf{A})$ Furthermore, if $k \leq n$, then $r_i \in p_k$

Proof Applying Lemma 4 11 to G_{kj}, we have an (r_j, r_k)-path in $B_l(\mathbf{A})$ Since G_{ij} is applied before G_{kj}, we know that $r_i \notin s_{l-1}$, so that applying Corollary 4 7 2 with $k = l - 1$ we have a (c_j, r_i)-path in $B_l(\mathbf{A})$ These paths together with the $r_j c_j$ edge give an (r_i, r_k)-walk in $B_l(\mathbf{A})$, so that there must be an (r_i, r_k)-path as well If $k < n$, the $r_k c_k$ edge and this path imply that $r_i \in p_k$ □

Lemma 4 15 *Let $1 \leq j \leq n$, $1 \leq k \leq n$, $1 \leq i \leq m$ and let $l = min\{i, n\}$ If G_{jk} and G_{ij} are both applied in Algorithm 1, then there is a (c_k, r_i)-path and an (r_k, r_i)-path in $B_l(\mathbf{A})$*

Proof From the proof of Lemma 4 10, $k < j < i$ Lemma 4 8 implies that $r_j \notin s_{l-1}$ and since G_{jk} is applied, Corollary 4 7 2 gives a (c_k, r_j)-path in $B_l(\mathbf{A})$ Clearly $r_i \notin s_{l-1}$, so since G_{ij} is applied, Corollary 4 7 2 gives a (c_j, r_i)-path in $B_l(\mathbf{A})$ Combining these with the $r_k c_k$ and $c_j r_j$ edges gives a (c_k, r_i)-walk and an (r_k, r_i)-walk in $B_l(\mathbf{A})$ Therefore, there is a (c_k, r_i)-path and an (r_k, r_i)-path in $B_l(\mathbf{A})$ □

We are now ready to prove that if there is a chain of rotations as described in Lemma 4 3 so that $(i, j) \in \bar{\mathbf{Q}}$, then $(i, j) \in \mathbf{Q}$ also

Lemma 4.16 *Let*

$$G_{i_1 j_1}, G_{i_2 j_2}, \ldots, G_{i_k j_k},$$

denote a chain of minimal length $k \geq 2$ of the rotations determined by the application of Algorithm 1 to a structure **A**.

a) *If $(i_1 = i_2$ or $i_1 = j_2)$ and $(i_{k-1} = j_k$ or $j_{k-1} = j_k)$, then there is an (r_{j_1}, r_{i_k})-path in $B_l(\mathbf{A})$ where $l = min\{n, i_k\}$, and if $i_k \leq n$, then $r_{j_1} \in p_{i_k}$*

b) *If $(i_1 = i_2$ or $i_1 = j_2$) and $i_{k-1} = i_k$, then $r_{j_1} \in p_{j_k}$*

c) *If $j_1 = j_2$ and $i_{k-1} = i_k$, then $r_{i_1} \in p_{j_k}$*

d) *If $j_1 = j_2$ and $(i_{k-1} = j_k$ or $j_{k-1} = j_k)$, then there is an (r_{i_1}, r_{i_k})-path in $B_l(\mathbf{A})$ where $l = min\{n, i_k\}$, and if $i_k \leq n$, then $r_{i_1} \in p_{i_k}$*

Proof The proof is by induction on k.

Base Case If $k = 2$ then

a) becomes $G_{i_1 j_1}$, $G_{i_2 i_1}$ and the path exists by Lemma 4.15. If $i_2 \leq n$, then $r_{j_1} \in p_{i_2}$ by Lemma 4.13. All the other cases ($G_{i_1 j_2}$, $G_{i_1 i_1}$, $G_{i_1 j_1}$, $G_{i_1 j_1}$, $G_{i_1 i_1}$, $G_{i_2 i_1}$) are impossible.

b) becomes $G_{i_1 j_1}$, $G_{i_1 j_2}$ and $r_{j_1} \in p_{j_2}$ by Lemma 4.12. The other case $G_{i_1 j_1}$, $G_{i_1 i_1}$ is impossible.

c) becomes $G_{i_1 j_1}$, $G_{i_1 j_1}$ which never occurs since we do not eliminate the same nonzero twice.

d) becomes $G_{i_1 j_1}$, $G_{i_2 j_1}$ and, by Lemma 4.14, the path condition exists and if $i_2 \leq n$, then $r_{i_1} \in p_{i_2}$. The other case $G_{j_1 j_1}$, $G_{i_2 j_1}$ is not possible.

Hence the lemma is true for the case $k = 2$.

Induction Hypothesis. Suppose that the lemma is true for all chains of minimal length $k \leq v$.

Induction Step. We consider a chain of minimal length $k = v+1$, which may be written

$$G_{i_1 j_1}, G_{i_2 j_2}, \quad , G_{i_v j_v}, G_{i_{v+1} j_{v+1}} \tag{4.3}$$

Since we know that the lemma is true for $k = 2$, we may assume that $v \geq 2$. Now

chain 1 $G_{i_1 j_1}, G_{i_2 j_2}, \quad , G_{i_v j_v}$

and

chain 2 $G_{i_2 j_2}, \quad , G_{i_v j_v}, G_{i_{v+1} j_{v+1}}$

are chains of length v, so by the induction hypothesis, the lemma is true for these two subchains (and all other subchains) of (4.3).

There are nine cases to consider, one corresponding to each of the possible combinations of the equalities listed in the statement of the lemma. We take these in order.

1. If $i_1 = i_2$ and $i_{k-1} = j_k$, then (4.3) may be written as

$$G_{i_1 j_1}, G_{i_1 j_2}, G_{i_3 j_2}, \quad , G_{i_v j_v}, G_{i_{v+1} i_v}$$

with $v \geq 3$, since such a chain is impossible if $v = 2$. Now $j_1 < j_2 < i_v \leq n$. Chain 2 satisfies case d) so that by the induction hypothesis, there is an $(r_{i_1}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$ where $l =$

$\min\{n, i_{v+1}\}$ This implies that $r_{i_1} \notin s_{l-1}$ Since $G_{i_1 j_1}$ is applied, there is a (c_{j_1}, r_{i_1})-path in $B_l(\mathbf{A})$ by Corollary 4 7 2 These two paths combined with the $r_{j_1} c_{j_1}$ edge make an $(r_{j_1}, r_{i_{v+1}})$-walk in $B_l(\mathbf{A})$, so that there is an $(r_{j_1}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$ If $i_{v+1} \leq n$, then $l = i_{v+1}$ and the $r_{i_{v+1}} c_{i_{v+1}}$ edge also exists in $B_{i_{v+1}}(\mathbf{A})$ so that there is a $(c_{i_{v+1}}, r_{j_1})$-path as well That is, $r_{j_1} \in p_{i_{v+1}}$

2 If $i_1 = j_2$ and $i_{k-1} = j_k$, then (4 3) may be written as

$$G_{i_1 j_1}, G_{i_2 i_1}, \quad , G_{i_v j_v}, G_{i_{v+1} i_v},$$

where $v \geq 2$ Either $i_2 = i_3$ or $i_2 = j_3$, so that chain 2 satisfies case a) and by the induction hypothesis, there is an $(r_{i_1}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$ where $l = \min\{n, i_{v+1}\}$ Since $r_{i_1} \notin s_{l-1}$ and since $G_{i_1 j_1}$ is applied, there is a (c_{j_1}, r_{i_1})-path in $B_l(\mathbf{A})$ by Corollary 4 7 2 Combining these paths with the edge $r_{j_1} c_{j_1}$ gives an $(r_{j_1}, r_{i_{v+1}})$-walk which implies the existence of an $(r_{j_1}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$, and if $i_{v+1} \leq n$, we have $r_{j_1} \in p_{i_{v+1}}$

3 If $i_1 = i_2$ and $j_{k-1} = j_k$, then (4 3) may be written as

$$G_{i_1 j_1}, G_{i_1 j_2}, G_{i_3 j_2}, \quad , G_{i_v j_{v-1}}, G_{i_v j_v}, G_{i_{v+1} j_v},$$

where $v \geq 2$ Chain 2 satisfies case d) so that if $l = \min\{n, i_{v+1}\}$, then $B_l(\mathbf{A})$ contains an $(r_{i_1}, r_{i_{v+1}})$-path by the induction hypothesis Since $r_{i_1} \notin s_{l-1}$ and $G_{i_1 j_1}$ is applied, there is a (c_{j_1}, r_{i_1})-path in $B_l(\mathbf{A})$ by Corollary 4 7 2 Combining these two paths with the $r_{j_1} c_{j_1}$ edge gives an $(r_{j_1}, r_{i_{v+1}})$-walk in $B_l(\mathbf{A})$ Consequently, there is an $(r_{j_1}, r_{i_{v+1}})$-path

in $B_l(\mathbf{A})$ If $i_{v+1} \leq n$, then the $r_{i_{v+1}}c_{i_{v+1}}$ edge also exists in $B_l(\mathbf{A}) = B_{i_{v+1}}(\mathbf{A})$ so that there is a $(c_{i_{v+1}}, r_{j_1})$-path as well That is, $r_{j_1} \in p_{i_{v+1}}$

4 If $i_1 = j_2$ and $j_{k-1} = j_k$, then (4 3) may be written as

$$G_{i_1 j_1},\ G_{i_2 i_1},\ \ \ \ ,\ G_{i_v j_{v-1}},\ G_{i_v j_v},\ G_{i_{v+1} j_v}$$

with $v \geq 3$, since such a chain is impossible if $v = 2$ Since either $i_2 = i_3$ or $i_2 = j_3$, chain 2 satisfies case a) so that for $l = \min\{n, i_{v+1}\}$, $B_l(\mathbf{A})$ contains an $(r_{i_1}, r_{i_{v+1}})$-path by the induction hypothesis Since $G_{i_1 j_1}$ is applied and $r_{i_1} \notin s_{l-1}$, there is a (c_{j_1}, r_{i_1})-path in $B_l(\mathbf{A})$ by Corollary 4 7 2 Combining these paths with the $r_{j_1}c_{j_1}$ edge produces an $(r_{j_1}, r_{i_{v+1}})$-walk, so that there is an $(r_{j_1}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$ If $i_{v+1} \leq n$ so that $l = i_{v+1}$, then the $r_{i_{v+1}}c_{i_{v+1}}$ edge also exists, and hence there is a $(c_{i_{v+1}}, r_{j_1})$-path in $B_{i_{v+1}}(\mathbf{A})$ That is, $r_{j_1} \in p_{i_{v+1}}$

5 If $i_1 = i_2$ and $i_{k-1} = i_k$, then (4 3) may be written as

$$G_{i_1 j_1},\ G_{i_1 j_2},\ G_{i_3 j_2},\ \ \ \ ,\ G_{i_v j_v},\ G_{i_v j_{v+1}}$$

with $v \geq 3$, since such a chain is impossible if $v = 2$ Now chain 2 satisfies case c) so that there is a $(c_{j_{v+1}}, r_{i_1})$-path in $B_{j_{v+1}}(\mathbf{A})$ Thus, $r_{i_1} \notin s_{j_{v+1}-1}$, and since $G_{i_1 j_1}$ is applied, $B_{j_{v+1}}(\mathbf{A})$ contains a (c_{j_1}, r_{i_1})-path by Corollary 4 7 2 Combining these paths with the $r_{j_1}c_{j_1}$ edge gives a $(c_{j_{v+1}}, r_{j_1})$-walk which implies the existence of a $(c_{j_{v+1}}, r_{j_1})$-path in $B_{j_{v+1}}(\mathbf{A})$ Hence, $r_{j_1} \in p_{j_{v+1}}$

6 If $i_1 = j_2$ and $i_{k-1} = i_k$, then (4 3) may be written as

$$G_{i_1 j_1},\ G_{i_2 i_1},\ \ \ \ ,\ G_{i_v j_v},\ G_{i_v j_{v+1}},$$

where $v \geq 2$. Since either $i_2 = i_3$ or $i_2 = j_3$, chain 2 satisfies case b), and by the induction hypothesis, there is a $(c_{j_{v+1}}, r_{i_1})$-path in $B_{j_{v+1}}(\mathbf{A})$. Thus $r_{i_1} \notin s_{j_{v+1}-1}$, and since $G_{i_1 j_1}$ is applied, there exists a (c_{j_1}, r_{i_1})-path in $B_{j_{v+1}}(\mathbf{A})$ by Corollary 4.7.2. These paths combined with the $r_{j_1} c_{j_1}$ edge produce a $(c_{j_{v+1}}, r_{j_1})$-walk, and thus a $(c_{j_{v+1}}, r_{j_1})$-path, in $B_{j_{v+1}}(\mathbf{A})$. Hence, $r_{j_1} \in p_{j_{v+1}}$.

7. If $j_1 = j_2$ and $i_{k-1} = i_k$, then (4.3) may be written as

$$G_{i_1 j_1}, G_{i_2 j_1}, \ldots, G_{i_v j_v}, G_{i_v j_{v+1}},$$

where $v \geq 2$. Chain 1 satisfies case d), so for $l = \min\{n, i_v\}$, $B_l(\mathbf{A})$ contains an (r_{i_1}, r_{i_v})-path. Thus $r_{i_1} \notin s_{l-1}$, and consequently $r_{i_1} \notin s_{j_{v+1}-1}$ also (since $j_{v+1} \leq l$). Either $i_2 = i_3$ or $i_2 = j_3$, so that chain 2 satisfies case b) and hence there is a $(c_{j_{v+1}}, r_{j_1})$-path in $B_{j_{v+1}}(\mathbf{A})$ by the induction hypothesis. Since $G_{i_1 j_1}$ is applied and $r_{i_1} \notin s_{j_{v+1}-1}$, by Corollary 4.7.2 there exists a (c_{j_1}, r_{i_1})-path in $B_{j_{v+1}}(\mathbf{A})$. The latter two paths combined with the $r_{j_1} c_{j_1}$ edge give a $(c_{j_{v+1}}, r_{i_1})$-walk in $B_{j_{v+1}}(\mathbf{A})$, so that there is also a $(c_{j_{v+1}}, r_{i_1})$-path, and hence $r_{i_1} \in p_{j_{v+1}}$.

8. If $j_1 = j_2$ and $i_{k-1} = j_k$, then (4.3) may be written as either

$$G_{i_1 j_1}, G_{i_2 j_1}, G_{i_3 i_2}, \ldots, G_{i_v j_v}, G_{i_{v+1} i_v}, \tag{4.4}$$

where $v \geq 2$, or

$$G_{i_1 j_1}, G_{i_2 j_1}, G_{i_2 j_3}, \ldots, G_{i_v j_v}, G_{i_{v+1} i_v}, \tag{4.5}$$

where $v \geq 4$. Consider the subchain

$$G_{i_3 i_2}, \ldots, G_{i_v j_v}, G_{i_{v+1} i_v}$$

of (4 4) If $v = 2$, then this subchain reduces to the single rotation $G_{i_3 i_2}$ In this case, there is an $(r_{i_2}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$, where $l = \min\{n, i_{v+1}\}$, by Lemma 4 11 If $v > 2$, the subchain is of length at least 2 and satisfies case a), and thus an $(r_{i_2}, r_{i_{v+1}})$-path exists in $B_l(\mathbf{A})$ by the induction hypothesis

Consider now the subchain

$$G_{i_2 j_3}, \quad \ldots \quad , G_{i_v j_v}, G_{i_{v+1} i_v}$$

of (4 5), which is of length at least 2 and satisfies case d) Thus the induction hypothesis implies that there is an $(r_{i_2}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$

So in every case, there is an $(r_{i_2}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$, which implies $r_{i_2} \notin s_{l-1}$ Since $G_{i_1 j_1}$ is applied before $G_{i_2 j_1}$, $r_{i_1} \notin s_{l-1}$ Corollary 4 7 2 applied to $G_{i_1 j_1}$ gives a (c_{j_1}, r_{i_1})-path in $B_l(\mathbf{A})$, and applying it to $G_{i_2 j_1}$ gives a (c_{j_1}, r_{i_2})-path in $B_l(\mathbf{A})$ Combining these three paths produces an $(r_{i_1}, r_{i_{v+1}})$-walk so that there must be an $(r_{i_1}, r_{i_{v+1}})$-path as well Furthermore, if $i_{v+1} \leq n$, then $B_l(\mathbf{A})$ also contains the $r_{i_{v+1}} c_{i_{v+1}}$ edge, which gives a $(c_{i_{v+1}}, r_{i_1})$-path, which means $r_{i_1} \in p_{i_{v+1}}$

9 If $j_1 = j_2$ and $j_{k-1} = j_k$, then (4 3) may be written as either

$$G_{i_1 j_1}, G_{i_2 j_1}, G_{i_2 j_3}, \quad \ldots \quad , G_{i_v j_v}, G_{i_{v+1} j_v}, \tag{4 6}$$

where $v \geq 3$, or

$$G_{i_1 j_1}, G_{i_2 j_1}, G_{i_3 i_2}, \quad \ldots \quad , G_{i_v j_v}, G_{i_{v+1} j_v}, \tag{4 7}$$

where $v \geq 4$ The subchain $G_{i_2 j_3}, \quad \ldots \quad , G_{i_v j_v}, G_{i_{v+1} j_v}$ of (4 6) is of length at least 2 and satisfies case d) By the induction hypothesis,

there is an $(r_{i_2}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$, where $l = \min\{i_{v+1}, n\}$. Similarly, the subchain $G_{i_3 i_2}$, , $G_{i_v j_v}$, $G_{i_{v+1} j_v}$ of (4.7), satisfies case a), and the induction hypothesis implies the existence of an $(r_{i_2}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$. Thus, in any case, $r_{i_2} \notin s_{l-1}$, and since $G_{i_1 j_1}$ is applied before $G_{i_2 j_1}$, $r_{i_1} \notin s_{l-1}$. So Corollary 4.7.2 applied to $G_{i_1 j_1}$ gives a (c_{j_1}, r_{i_1})-path in $B_l(\mathbf{A})$, and applying it to $G_{i_2 j_1}$ gives a (c_{j_1}, r_{i_2})-path in $B_l(\mathbf{A})$. Combining these paths with the $(r_{i_2}, r_{i_{v+1}})$-path gives an $(r_{i_1}, r_{i_{v+1}})$-walk, which implies the existence of an $(r_{i_1}, r_{i_{v+1}})$-path in $B_l(\mathbf{A})$. Furthermore, if $i_{v+1} \leq n$, then $l = i_{v+1}$ and the $r_{i_{v+1}} c_{i_{v+1}}$ edge is in $B_{i_{v+1}}(\mathbf{A})$, so there is a $(c_{i_{v+1}}, r_{i_1})$-path as well. That is, $r_{i_1} \in p_{i_{v+1}}$.

In every case, if the lemma is true for $k = v$, then it is true for $k = v + 1$, completing the proof by induction. □

We now state our main results.

Theorem 4.17 *Let* **A** *be the structure of an* $m \times n$ *matrix with the Hall property and a nonzero diagonal. Then the structure* $\bar{\mathbf{Q}}$ *that results from application of Algorithm 1 to* **A** *is identical to the structure* **Q**.

Proof The entries of

$$\bar{\mathbf{Q}} = \prod_{t=1}^{k} structure(G_{i_t j_t}) \backslash \{(i, j) \mid j > n\}$$

are characterized by Algorithm 1 and Lemma 4.3. Let $(x, y) \in \bar{\mathbf{Q}}$. Then $y \leq n$. If $x = y$, then $r_x \in p_y$ since **A** has a nonzero diagonal and $r_x \notin s_{y-1}$. If $x \neq y$ and G_{xy} is one of the applied rotations, then $r_x \in p_y$ by Corollary 4.7.3, whereas if G_{yx} (with $x \leq n$) is one of the applied rotations, then $r_x \in p_y$ by

Lemma 4.9. Finally, if there is a chain of rotations

$$G_{i_{s_1} j_{s_1}}, G_{i_{s_2} j_{s_2}}, \quad , G_{i_{s_k} j_{s_k}}$$

with $x = i_{s_1}$ or $x = j_{s_1}$ and $y = i_{s_k}$ or $y = j_{s_k}$ so that $(x, y) \in \bar{\mathbf{Q}}$, then $r_x \in p_y$ by Lemma 4.16. Given Lemma 4.10, these exhaust all possibilities in Lemma 4.3, so $(x, y) \in \mathbf{Q}$ by Theorem 2.4. Thus $\bar{\mathbf{Q}} \subseteq \mathbf{Q}$.

To show the reverse inclusion, suppose $(x, y) \in \mathbf{Q}$. For any full rank $m \times n$ matrix A such that *structure*$(A) = \mathbf{A}$, suppose its QR factorization is computed using a sequence of Givens transformations

$$G_{i_1 j_1}, G_{i_2 j_2}, \quad , G_{i_k j_k}$$

as determined by Algorithm 1. By Lemma 2.3, *structure*$(Q) \subseteq \bar{\mathbf{Q}}$. Thus $\mathbf{Q} \subseteq \bar{\mathbf{Q}}$, completing the proof. □

Theorem 4.18 *Let* $\mathbf{A}$ *be the structure of an* $m \times n$ *matrix with the Hall property and a nonzero diagonal. Then the structure* $\bar{\mathbf{R}}$ *that results from application of Algorithm 1 to* $\mathbf{A}$ *is identical to the structure* $\mathbf{R}$.

Proof. Algorithm 1 computes

$$\begin{aligned}
\bar{\mathbf{R}} &= \left(\prod_{k=t}^{1} \textit{structure}(G_{i_k j_k}) \right) \mathbf{A} \backslash \{(i, j) \mid i > j\} \\
&= \left(\prod_{k=1}^{t} \textit{structure}(G_{i_k j_k}^*) \right)^* \mathbf{A} \backslash \{(i, j) \mid i > j\} \\
&= \left(\bar{\mathbf{Q}}, \bar{\mathbf{q}}_{\mathbf{n+1}}, \quad , \bar{\mathbf{q}}_{\mathbf{m}} \right)^* \mathbf{A} \backslash \{(i, j) \mid i > j\} \\
&= \bar{\mathbf{Q}}^* \mathbf{A} \backslash \{(i, j) \mid i > j\}
\end{aligned}$$

But since $\bar{\mathbf{Q}} = \mathbf{Q}$ by Theorem 4 17, it follows that $\bar{\mathbf{R}} = \mathbf{R}$ by Theorem 2 5

□

Chapter 5

Conclusions

In the previous chapter we showed that given an $m \times n$ matrix A with the Hall property, where $m \geq n$, it is possible to find an ordering for the Givens transformations such that the first m columns of the product of the structures of the transformations is equal to the structure **Q**. In addition, the structure generated for the factor R by this ordering is correct. That is, we showed that there is a tight ordering for the Givens transformations.

A consequence of Theorems 4.17 and 4.18 is that if A has the strong Hall property or if A has the Hall property and is in Dulmage-Mendelsohn form (see Chapter 3), then all orderings that compute the QR factorization column by column using diagonal pivots are tight. In the first case, there are no nonempty Hall sets. The second case follows similarly since the diagonal blocks have the strong Hall property and the QR factorization of A is easily obtained from the QR factorizations of the diagonal blocks of A. These results correspond to those of Coleman et al. [2] for the case of row-wise

elimination using diagonal pivots

A second consequence allows an interesting upper bound on the time complexity of Algorithm 1 (or an analogous algorithm for computing a numeric QR factorization) Corollary 4 7 2 restricts the number of rotations required to the number of nonzeros below the diagonal in $\mathbf{Q}$ (In fact, the number of rotations may be fewer since not all of these positions necessarily fill during the elimination) The number of operations required for each rotation is linear in the number of nonzeros in the two rows involved, and this number is certainly not more than $2n$ This gives a crude bound on the operation count in $\mathcal{O}(n\tau(\mathbf{Q}))$ where $\tau(\mathbf{Q})$ is the number of nonzeros in $\mathbf{Q}$

We do not claim that a tight ordering for the Givens transformations is minimal in the sense that it uses the minimum number of transformations We illustrate this with an example

Example 5 1 Let

$$\mathbf{A} = \begin{pmatrix} \star & 0 & 0 \\ 0 & \star & 0 \\ \star & \star & \star \\ \star & 0 & 0 \end{pmatrix}$$

The maximal Hall sets S_j are

$$\begin{aligned} S_0 &= \emptyset \\ S_1 &= \emptyset \\ S_2 &= \emptyset \\ S_3 &= \{c_2, c_3\} \end{aligned}$$

Two possible orderings for the Givens transformations include

$$G_{31}, G_{41}, G_{32}, G_{42}, G_{43} \tag{5.1}$$

and

$$G_{41}, G_{31}, G_{32} \tag{5.2}$$

Both of these orderings are tight orderings since they conform to that specified by Algorithm 1, but (5.2) uses fewer rotations than (5.1). □

Clearly, there is still a need for heuristic ordering strategies to find better orderings within the constraints of Algorithm 1.

Example 5.1 illustrates another point. If we consider the product of the structures of the Givens transformations of the two orderings, we obtain from (5.1) the structure for the (square) factor $\hat{Q}$ given by

$$\hat{\mathbf{Q}}_1 = \begin{pmatrix} \star & \star & \star & \star \\ 0 & \star & \star & \star \\ \star & \star & \star & \star \\ \star & \star & \star & \star \end{pmatrix},$$

but from (5.2) we obtain

$$\hat{\mathbf{Q}}_2 = \begin{pmatrix} \star & \star & \star & \star \\ 0 & \star & \star & 0 \\ \star & \star & \star & 0 \\ \star & \star & \star & \star \end{pmatrix}$$

Since $\mathbf{A}$ is 4×3, the first 3 columns of $\hat{\mathbf{Q}}_1$ and $\hat{\mathbf{Q}}_2$ are identical to the structure of the (rectangular) 4×3 factor Q, but they differ in the fourth

column However, Hare et al [11] showed in the proof of their Theorem 2 1 that any unitary matrix whose first 3 columns are identical to Q must have its $(2,4)$ and $(3,4)$ entries equal to 0 The proof is based on the fact that columns 2 and 3 of any full rank $A \in \mathbf{A}$, and hence, of the factor Q of the QR factorization of A, span a subspace of dimension 2 and are zero outside of rows 2 and 3 Therefore, column 4 must have zeros in rows 2 and 3 since it is orthogonal to columns 2 and 3 $\hat{\mathbf{Q}}_2$ conforms to this structure but $\hat{\mathbf{Q}}_1$ does not It follows that if the structure of the square $m \times m$ factor $\hat{Q}$ is required, then the one determined by the sequence of transformations specified by Algorithm 1 may not be correct Finding an adaptation to Algorithm 1 that is guaranteed to give a correct structure for the square Q is an open problem

If Q is stored in factored form, where each Givens transformation is represented by a single parameter, then that part of $\mathbf{Q}$ below the diagonal is sufficient for the purpose In fact, Algorithm 1 could be modified to produce a still smaller data structure by replacing $\mathbf{Q} \leftarrow \mathbf{Q}$ *structure*$(G_{i,j})$ with $\mathbf{Q} \leftarrow \mathbf{Q} \cup \{(i,j)\}$

Other limitations on Algorithm 1 are the requirement that A have a nonzero diagonal and the use of diagonal pivoting The proofs of Chapter 4 depend critically on these assumptions It remains an open problem to determine if it is possible to relax these conditions and still obtain tight orderings for the Givens transformations

We do not know if the ordering that employs the fewest transformations is necessarily a tight ordering We have seen that in the case of matrices with the strong Hall property, all orderings that use diagonal pivoting and elimination column by column are tight However, in the case where the

matrix has the Hall property but not the strong Hall property, the situation is more complex It may be the case that variable pivot orderings are more efficient but not necessarily tight This is a possible area for future work

Bibliography

[1] J A Bondy and U S R Murty *Graph Theory with Applications* New York North-Holland, 1976

[2] Thomas F Coleman, Anders Edenbrandt and John R Gilbert 'Predicting Fill for Sparse Orthogonal Factorization', *J Assoc for Comp Mach* 33 (1986), 517-532

[3] I S Duff 'Pivot Selection and Row Ordering in Givens Reduction on Sparse Matrices', *Computing* 13 (1974), 239-248

[4] Alan George and Michael T Heath 'Solution of Sparse Linear Least Squares Problems Using Givens Rotations', *Linear Algebra and Its Applications* 34 (1980), 69-83

[5] Alan George and J W H Liu *Computer Solution of Large Sparse Positive Definite Systems* Englewood Cliffs, New Jersey Prentice-Hall Inc , 1981

[6] Alan George and J W H Liu 'The Evolution of the Minimum Degree Ordering Algorithm', *SIAM Review* 31 (1989), 1-19

[7] Alan George, Joseph Liu and Esmond Ng 'Row-Ordering Schemes for Sparse Givens Transformations I Bipartite Graph Model', *Linear Algebra and Its Applications* 61 (1984), 55-81

[8] Alan George , Joseph Liu and Esmond Ng 'A Data Structure for Sparse QR and LU Factorizations', *SIAM Journal of Scientific and Statistical Computing* 9 (1988), 100-121

[9] Alan George and E Ng 'Symbolic Factorization for Sparse Gaussian Elimination with Partial Pivoting', *SIAM Journal of Scientific and Statistical Computing* 8 (1987), 877-898

[10] G H Golub and C F Van Loan *Matrix Computations* London Oxford Academic, 1986

[11] D Hare, C R Johnson, D D Olesky and P van den Driessche 'Sparsity Analysis of the QR factorization', to appear in *SIAM J Matrix Anal Appl*

[12] Joseph W H Liu 'On General Row Merging Schemes for Sparse Givens Transformations', *SIAM Journal of Scientific and Statistical Computing* 7 (1986), 1190-1211

[13] Ole Osterby and Zahari Zlatev *Direct Methods for Sparse Matrices* Lecture Notes in Computer Science, Springer-Verlag, 1983

[14] George Ostrouchov 'Symbolic Givens Reduction and Row-Ordering in Large, Sparse, Least Squares Problems', *SIAM Journal of Scientific and Statistical Computing* 8 (1987), 248-264

[15] Sergio Pissanetsky *Sparse Matrix Technology* London Academic Press, 1984

[16] D S Watkins *Fundamentals of Matrix Computations* New York John Wiley and Sons, 1991

[17] M Yannakakis 'Computing the Minimum Fill-in is NP-complete', *SIAM J Algebraic Discrete Methods* 2 (1981), 77-79

VITA

Surname **Gillespie** Given Names **Mary Irene**
Place of Birth **Regina, Saskatchewan** Date of Birth **20 May 1951**

Educational Institutions Attended

University of Saskatchewan, Regina Campus, Regina	1969 to 1974
University of Victoria	1988 to 1992

Degrees Awarded

Bachelor of Science	1974	University of Saskatchewan, Regina, Saskatchewan

Partial Copyright License

I hereby grant the right to lend my thesis (the title of which is shown below) to users of the University of Victoria Library, and to make single copies only for such users or in response to a request from the Library of any other university, or similar institution, on its behalf or for one of its users. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by me or a member of the University designated by me. It is understood that copying of this thesis for financial gain shall not be allowed without my written permission.

Title of Thesis:

Ordering Givens Transformations for Sparse QR Factorization

Author: [redacted]
Mary Irene Gillespie
January 8, 1993