Algorithmica (2004) 39: 83–93
DOI: 10.1007/s00453-003-1080-z

Algorithmica

Algorithms for Computing the QR Decomposition of a Set of Matrices with Common Columns[1]

Petko Yanev,[2] Paolo Foschi,[2] and Erricos John Kontoghiorghes[2,3]

Abstract. The QR decomposition of a set of matrices which have common columns is investigated. The triangular factors of the QR decompositions are represented as nodes of a weighted directed graph. An edge between two nodes exists if and only if the columns of one of the matrices is a subset of the columns of the other. The weight of an edge denotes the computational complexity of deriving the triangular factor of the destination node from that of the source node. The problem is equivalent to constructing the graph and finding the minimum cost for visiting all the nodes. An algorithm which computes the QR decompositions by deriving the minimum spanning tree of the graph is proposed. Theoretical measures of complexity are derived and numerical results from the implementation of this and alternative heuristic algorithms are given.

Key Words. QR decomposition, Givens rotations, Minimum spanning tree.

1. Introduction. Computationally intensive methods for deriving the least-squares estimators of seemingly unrelated regression and simultaneous equation models have been proposed [12]. These estimation methods require the QR decompositions of a set of matrices which have common columns. These columns correspond to exogenous factors that occur in more than one econometric relationship of the model. Consider the QR decomposition (QRD) of the full column rank matrix $A_i \in \mathbb{R}^{m\times k_i}$:

$$Q_i^T A_i = \begin{pmatrix} R_i \\ 0 \end{pmatrix}^{k_i}_{m-k_i} \qquad (i = 1, \ldots, G), \tag{1}$$

where $Q_i \in \mathbb{R}^{m\times m}$ is orthogonal and $R_i \in \mathbb{R}^{k_i\times k_i}$ is upper triangular. The exogenous matrices with common columns can be expressed as

$$A_i = AS_i \qquad (i = 1, \ldots, G), \tag{2}$$

where $A \in \mathbb{R}^{m\times n}$ and $S_i \in \mathbb{R}^{n\times k_i}$ is a selection matrix [5], [12], [15]. It is often the case that $n \ll \sum_{i=1}^{G} k_i$, i.e. the number of distinct factors is much less than the total number of factors occurring in the whole model.

The main method used to compute (1) is by forming the QRDs of $A_1, \ldots, A_G$ one at a time, without taking into account that the matrices may share common columns. Let

[1] This work is in part supported by Swiss National Foundation Grants 101312-100757 and 200020-100116/1. Part of the work was done while the third author was visiting INRIA-IRISA, Rennes, France, under the support of the host institution and Swiss National Foundation Grant 83R-065887.
[2] Institut d'Informatique, Université de Neuchâtel, CH-2007 Neuchatel, Switzerland. {Petko.Yanev, Paolo.Foschi, Erricos.Kontoghiorghes}@unine.ch.
[3] Computer Science and Information Systems, Birkbeck College, University of London, Bloomsbury, London WC1E 7HX, England, and Department of Public and Business Administration, University of Cyprus, 1678 Nicosia, Cyprus.

Received August 10, 2002; revised October 4, 2003. Communicated by Y. M. Kao.
Online publication January 28, 2004.

the QRD of A be given by

$$Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix} \begin{matrix} n \\ m-n \end{matrix}, \tag{3}$$

where $Q \in \mathbb{R}^{m\times m}$ is orthogonal and $R \in \mathbb{R}^{n\times n}$ is upper triangular. Thus, the upper triangular factor R_i in (1) can be derived by computing the QRD

$$\tilde{Q}_i^T RS_i = \begin{pmatrix} R_i \\ 0 \end{pmatrix} \begin{matrix} k_i \\ n-k_i \end{matrix} \qquad (i = 1, \ldots, G), \tag{4}$$

where $\tilde{Q}_i \in \mathbb{R}^{n\times n}$ is orthogonal [9]–[11]. The orthogonal matrix Q_i in (1) is defined by

$$Q_i = Q \begin{pmatrix} \tilde{Q}_i & 0 \\ 0 & I_{m-n} \end{pmatrix} \tag{5}$$

Notice that the QRDs in (4) are equivalent to the re-triangularization of a set of upper-triangular matrices after deleting columns.

Sequential and parallel strategies which compute the QRD of RS_i have been proposed [11], [12], [15]. These strategies use Givens rotations and exploit the non-full structure of RS_i. However, the occurrence of common columns among $RS_1, \ldots, RS_G$ has not been exploited. The purpose of this work is to propose and investigate sequential factorization strategies that take advantage of this possibility when $n \ll \sum_{i=1}^{G} k_i$. The algorithms are based on Givens rotations [10].

A Givens rotation in plane (i, j) that reduces to zero the element $b_{j,k}$ when it is applied from the left of $B = [b_{i,j}] \in \mathbb{R}^{m\times n}$ will be denoted by $G_{i,j}^{(k)}$, where $1 \le i, j \le m$ and $1 \le k \le n$. The rotation $G_{i,j}^{(k)} B$ affects only the ith and jth rows of B. The changes in these rows can be written as

$$\begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} b_{i,:} \\ b_{j,:} \end{pmatrix} = \begin{pmatrix} \tilde{b}_{i,1} & \cdots & \tilde{b}_{i,k} & \cdots & \tilde{b}_{i,n} \\ \tilde{b}_{j,1} & \cdots & \tilde{b}_{j,k} & \cdots & \tilde{b}_{j,n} \end{pmatrix}, \tag{6}$$

where $b_{j,k} \neq 0$, $c^2 + s^2 = 1$, $c = b_{i,k}/\tau$, $s = b_{j,k}/\tau$, $\tau^2 = b_{i,k}^2 + b_{j,k}^2$, $\tilde{b}_{i,k} = \tau$ and $\tilde{b}_{j,k} = 0$. If $b_{j,k} = 0$, then $G_{i,j}^{(k)} \equiv I_m$. Standard column notation is used to denote sub-vectors and sub-matrices [10]. The construction of a Givens rotation requires six flops in time denoted by t. The same time is required to apply the rotation to a two-element vector. Thus, nt flops are needed to compute (6). Notice that the rotation is not applied to the pair of elements $b_{i,k}$ and $b_{j,k}$ used in the construction of the rotation. These are set to τ and zero, respectively.

In the next section Givens' sequences for computing the QRD of RS_i $(i = 1, \ldots, G)$ are presented. Section 3 proposes an efficient algorithm for computing the QRDs of $RS_1, \ldots, RS_G$, which are represented as nodes of a directed graph. Numerical results are presented in Section 4 and the performance of the algorithm is evaluated. In Section 5 conclusions are offered.

2. Computing the QR decomposition of RS_i. There are many equivalent strategies for computing the QRD using Givens rotations [10]. Consider the case where the elements of a matrix below the main diagonal are annihilated column-by-column and from

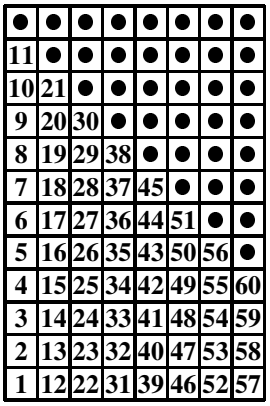


Fig. 1. Computing the QRD of $A \in \mathbb{R}^{12\times 8}$ using Givens rotations.

bottom to the top with zero elements being preserved throughout the annihilation process. Furthermore, let the Givens rotations be between adjacent planes. The number of Givens rotations required to compute (3) is given by $\sum_{i=1}^{n}(m-i) = n(2m-n-1)/2$ and Q^T is defined by

$$Q^T = \prod_{i=1}^{n}\prod_{j=1}^{m-i} G^{(i)}_{m-j,m-j+1}. \tag{7}$$

Figure 1 shows the annihilation pattern corresponding to this Givens' sequence, where $m = 12$ and $n = 8$. An entry i $(i = 1, \ldots, 35)$ indicates that the element is reduced to zero by the ith rotation. The complexity of computing the QRD (3) using this strategy is given by

$$\begin{aligned} C(m,n) &= t\sum_{i=1}^{n}(m-i)(n-i+1) \\ &= tn(3m(n+1) - n^2 - 3n - 2)/6. \end{aligned} \tag{8}$$

Thus, the complexity of computing the QRDs of $A_1, \ldots, A_G$ simultaneously is given by

$$T_1(m,k,G) = \sum_{i=1}^{G} C(m,\ k_i), \tag{9}$$

where $k = (k_1, \ldots, k_G)$.

Let S_i in (2) be expressed as $S_i \equiv (e_{\lambda_{i,1}} \cdots e_{\lambda_{i,k_i}})$ with $\lambda_i = (\lambda_{i,1}, \ldots, \lambda_{i,k_i})$, where $e_{\lambda_{i,j}}$ is the $\lambda_{i,j}$th column of the unit matrix I_n, $i = 1, \ldots, G$ and $j = 1, \ldots, k_i$ [11], [12], [15]. Then the number of Givens rotations needed to compute the QRD (4) is given by $\sum_{j=1}^{k_i}(\lambda_{i,j} - j)$ and the orthogonal matrix $\tilde{Q}_i^T$ is defined as

$$\tilde{Q}_i^T = \prod_{n=1}^{k_i}\prod_{j=1}^{\lambda_{i,n}-n} G^{(n)}_{\lambda_{i,n}-j,\lambda_{i,n}-j+1}. \tag{10}$$

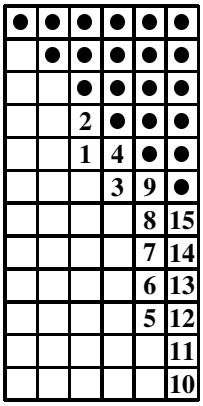


Fig. 2. Computing the QRD of RS_i, where $R \in \mathbb{R}^{12\times 12}$, $k_i = 6$ and $\lambda_i = (1, 2, 5, 6, 10, 12)$.

Figure 2 shows the Givens' sequence when re-triangularizing RS_i, where $n = 12, k_i = 6$ and $\lambda_i = (1,\ 2,\ 5,\ 6,\ 10,\ 12)$.

The complexity of computing the QRD (4) is given by

$$C_i(\lambda_i,\ k_i) = t\sum_{j=1}^{k_i}(\lambda_{i,j} - j)(k_i - j + 1). \tag{11}$$

Thus, the total complexity of computing (3) followed by re-triangularization of $RS_1, \ldots, RS_G$ one at a time is given by

$$T_2(\lambda_i,\ k_i,\ G) = C(m,\ n) + \sum_{i=1}^{G} C_i(\lambda_i,\ k_i). \tag{12}$$

3. The Minimum Spanning Tree Algorithm. The triangular factors $R, R_1, \ldots, R_G$ can be represented as nodes $N_0, N_1, \ldots, N_G$ of a weighted directed graph. An edge between two nodes N_i and N_j (denoted by $E_{i,j}$) exists and is directed from N_i towards N_j if and only if R_i contains all the columns of R_j ($i, j = 0, 1, \ldots, G$ and $i \neq j$). The weight of $E_{i,j}$ is denoted by $C_{i,j}$, the complexity of computing R_j given R_i. The goal is to construct the graph and to find the shortest path for visiting all the nodes. This is equivalent to finding the Minimum Spanning Tree (MST) of the graph which provides the minimum computational cost for deriving $R_1, \ldots, R_G$ [16], [17]. A spanning tree is a subgraph of a graph which contains all the nodes of the graph and is a tree.

To determine the MST the properties of the graph need to be explored. Let $\Gamma(V, E, n)$ be a graph with the sets of nodes and edges denoted by V and E, respectively, and n denotes the number of columns of the matrix R. The graph $\Gamma(V, E, n)$ can be divided into n levels $L_1, \ldots, L_n$. The matrices with k columns belong to the level L_k ($k = 1, \ldots, n$). Notice that R belongs to level L_n and level L_{n-1} can have at most n nodes (matrices). In general, there are at most $C_k^n = n!/k!\ (n-k)!$ nodes in the level L_k ($k = 1, \ldots, n$).

Therefore, the maximum number of nodes in $\Gamma(V, E, n)$ is

$$|V|_{\max} = \sum_{i=0}^{n-1} C_i^n = 2^n - 1. \tag{13}$$

Now, from the kth level there exists a maximum of $C_k^n(2^{(n-k)} - 2)$ edges. Thus, the maximum number of edges in the graph is

$$\begin{aligned} |E|_{\max} &= \sum_{i=0}^{n-2} C_i^n (2^{(n-i)} - 2) \\ &= 3^n - 2^{n+1} + 1. \end{aligned} \tag{14}$$

Let $E_{i,j}$ exist and let $p_{i,j}$ denote the position of the jth column of R_j in R_i. Notice that $p_{i,j} \geq j$ for every j. Then the cost of the edge $C_{i,j}$ is given by

$$C_{i,j} = t \sum_{j=1}^{k_j} (p_{i,j} - j)(k_j - j + 1). \tag{15}$$

Now, let $R_s \in L_p$, $R_h \in L_q$ and $R_i \in L_r$, where $E_{s,i}$ and $E_{h,i}$ exist, and $p \neq q \neq r$. If there is a path from R_s to R_h, then $C_{h,i} \leq C_{s,i}$. Therefore, $E_{s,i}$ can be deleted from the graph. A path from R_s to R_h exists if and only if the node R_h can be reached from the node R_s. When this rule is applied the number of the edges to be computed is reduced. Figure 3 illustrates the graph $\Gamma(V, E, 6)$ with all (Figure 3(a)) and with the reduced (Figure 3(b)) number of edges. The matrices R and R_i $(i = 1, \ldots, G)$ are denoted by square and round frames, respectively. The indexes of columns for each matrix are shown by a sequence of digits in the frames.

In order to determine the MST, the cost $C_{i,j}$ of each edge is computed and, for each node, the incoming edge with minimum cost is selected. If more than one incoming edge with equal weights exist, then one of them is selected randomly. The correctness of this algorithm follows from the acyclic property of $\Gamma(V, E, n)$ [1]. The time required to derive $C_{i,j}$ depends on the time to compute $p_{i,1}, \ldots, p_{i,k_j}$ and calculate the summation (15). At most k_i comparisons are necessary to determine $p_{i,1}, \ldots, p_{i,k_j}$. A single comparison and the summation of (15) requires one and $5k_j$ flops, respectively. The total time needed to compute $C_{i,j}$ is $k_i + 5k_j \leq 6k_i \equiv k_i t$. Let

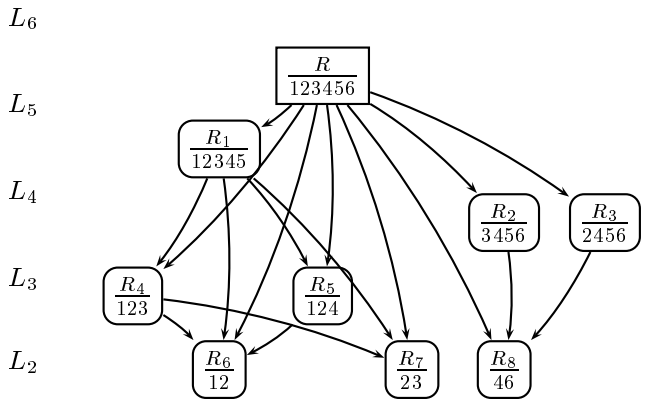
$$T_{\text{EDGE}} = \max_{i=1,\ldots,G} (k_i t) \tag{16}$$

and the upper bound of the time needed for deriving the MST of a graph with $|E|$ nodes be given by

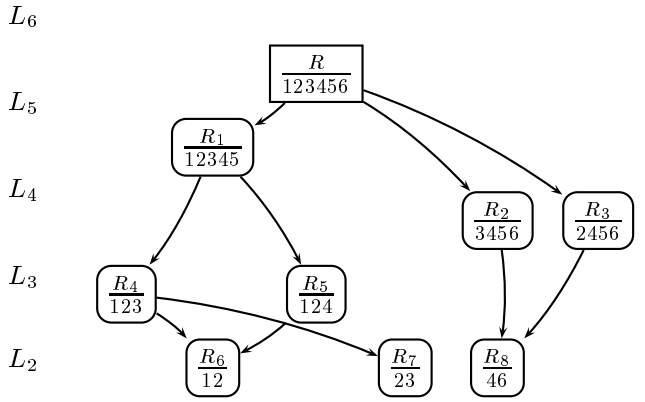
$$T_{\text{MST}} \leq |E| T_{\text{EDGE}} + |E|. \tag{17}$$

Here $|E|T_{\text{EDGE}}$ is the maximum time needed to compute the costs of all edges and $|E|$ is the maximum number of comparisons that could be done. Then the complexity of computing the matrices $R_1, \ldots, R_G$ using the MST approach is

$$T_3(k_i, m, n, p_i, G) = C(m, n) + \sum_{i=1}^{G} \sum_{j=1}^{k_i} (p_{i,j} - j)(k_i - j + 1) + T_{\text{MST}}, \tag{18}$$



(a)



(b)

Fig. 3. The Graph $\Gamma(V, E, n)$ with all and the reduced number of edges, where $|V| = 9$ and $n = 6$. (a) The Graph $\Gamma(V, E, 6)$, where $|E| = 17$. (b) The Graph $\Gamma(V, E, 6)$, where $|E| = 10$.

where $C(m, n)$ is given by (8) and corresponds to the complexity of computing the QRD (3) and $p_i = (p_{i,1} \cdots p_{i,k_i})$.

The MST approach reduces the complexity in the specific case where the columns of some of the matrices are subsets of the columns of other matrices. In order to exploit the possibility of common columns occurring in $R_1, \ldots, R_G$ new nodes (hereafter called artificial nodes) are added in the graph $\Gamma(V, E, n)$. An artificial node is the conjunction of the columns of two or more matrices. The QRD of these matrices might be more quickly derivable given the QRD of the artificial node. Figure 4 illustrates the graph $\Gamma(V, E, 6)$, where the two artificial nodes $\tilde{R}_9$ and $\tilde{R}_{10}$ are denoted by square frames. Thus, the problem becomes one of finding the optimal tree which covers $R_1, \ldots, R_G$ in the graph that includes all artificial nodes. Algorithm 1 computes this optimal tree.

Now, let the full graph generated by Algorithm 1 be denoted by $\Gamma_{\text{F}}(V_{\text{F}}, E_{\text{F}}, n)$, where $|V_{\text{F}}| = 2^G + 1$ and the maximum number of edges is given by $|E_{\text{F}}|_{\text{MAX}} = 2^G(2^G + 1)/2$.

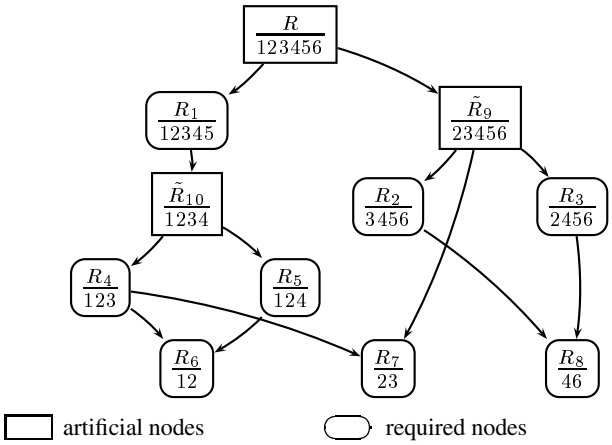


Fig. 4. The Graph $\Gamma(V, E, n)$ with the artificial nodes $\tilde{R}_9$ and $\tilde{R}_{10}$, where $|V| = 10$, $|E| = 9$ and $n = 6$.

The number of all subgraphs which include the matrices $R_1, \ldots, R_G$ is $2^{(2^G - G)}$. Thus, an upper bound for the total complexity of this algorithm is

$$C(G) = 2^{(2^G - G)} T_{\text{MST}} + 2^G (2^G + 1) T_{\text{EDGE}}/2 + T_s, \tag{19}$$

where T_{EDGE} and T_{MST} are given by (16) and (17), respectively. The time to compute the complexities of each MST and to derive the minimum one is denoted by T_s.

Algorithm 1 implements the optimal strategy for computing $R_1, \ldots, R_G$, given R. However, this optimal strategy has a double exponential complexity. Thus, it is not computationally feasible. To reduce the computational cost of Algorithm 1 a heuristic approach can be considered. The heuristic algorithm (Algorithm 2) computes the MST of the initial matrices $R_1, \ldots, R_G$ and then searches for artificial nodes which can reduce the weight of the tree. An artificial node is added to the MST if and only if it reduces the complexity between an existing node and its children. Then a new MST is reconstructed and the procedure is repeated. A maximum of 2^G artificial nodes can be constructed from the G initial matrices. Each of these artificial nodes is evaluated to determine whether it should be included in the tree or not. Thus, the complexity of finding the MST using this heuristic approach is exponential $O(2^G)$ and computationally expensive.

Algorithm 2 can be modified to reduce its high complexity. The artificial nodes are constructed from the columns of those two child nodes which have the maximum number of common columns. In this way not all 2^G artificial nodes are considered. The

Algorithm 1. The optimal MST algorithm

1. Construct the full graph consisting of $R_1, \ldots, R_G$ and all possible artificial nodes.
2. Find all the edges and their corresponding weights.
3. For all subgraphs which include $R_1, \ldots, R_G$ find the MST of each of them.
4. Compute the complexity of each MST and choose the minimum one.

Algorithm 2. The heuristic MST algorithm

1. Find the MST of $R_1, \ldots, R_G$.
2. **for** each node with more than one outgoing edge **do**
3. Construct all artificial nodes from the columns of the child nodes.
4. Compute the weights of the incoming and outgoing edges of the new artificial node.
5. Add the new artificial node to the tree if it reduces the cost.
6. Re-construct the MST until no more artificial nodes can be added.
7. **end for**

total number of computed matrices is $\bar{G}$, where $\max(\bar{G}) = 2G$. Thus, the complexity of determining the MST is polynomial $O(kG^2)$. The total complexity of the modified heuristic method is

$$T_4(k_i, m, n, p_i, \bar{G}) = C(m,n) + \sum_{i=1}^{\bar{G}} \sum_{j=1}^{k_i} (p_{i,j} - j)(k_i - j + 1) + O(kG^4). \tag{20}$$

4. Numerical Results. The modified heuristic approach is most efficient in the two cases, where there are many artificial nodes or none, but the columns of some matrices are subsets of the columns of others. The performance of the algorithms is considered in these two cases. First, when R_i is a sub-matrix of R_j for all $i = k, k+1, \ldots, G$ $j = 0, 1, \ldots, G$ $(i \neq j)$, where $1 < k < G/2$ and the MST containing $R_1, \ldots, R_G$ cannot be optimized. In this case no artificial nodes can be determined and the solution is optimal. Second, when the columns of none of the initial matrices $R_1, \ldots, R_G$ are subsets of the columns of other initial matrices, but where they have most of their columns common. Here many artificial nodes can be determined, but the solution may not be optimal. Table 1 shows the execution times of the modified heuristic method in these two cases. The performance of computing the QRDs (4) one at a time is also reported. Comparisons between the two methods are made also using their theoretical measures of complexity.

The constructed MST of each of the matrices in Table 1(a) is a binary tree. In this case no artificial nodes can be determined. Thus the MST strategy for factorizing the matrices $R_1, \ldots, R_G$ is optimal. Furthermore, the execution time of the *modified* heuristic algorithm is the same as that of Algorithm 2. In Table 1(b) the matrices $R_1, \ldots, R_G$ have a large number of common columns, but none of them is a sub-matrix of another matrix. In this example $G/2$ artificial nodes are constructed. Thus, the MST consists of $3G/2$ nodes. An artificial node is constructed from two matrices if they have at least half of their columns in common. Notice that, in both cases, the heuristic method executes in less than two-thirds of the time required by re-triangularization of $R_1, \ldots, R_G$ one at a time. The discrepancy between the theoretical and actual performance of the heuristic algorithm is attributable to the implementation overheads.

Table 1. Theoretical complexity and execution time of the modified heuristic method and that of re-triangularizing the G matrices one at a time, where the total number of distinct columns of all matrices is n.

		Execution times			Theoretical complexity
G	n	Retriang. method	Heuristic method	Retriang. / Heuristic	Retriang. / Heuristic
(a) No artificial nodes exist in the graph. The MST is a binary tree which consists of exactly G matrices.					
14	1120	4.39	3.09	1.42	1.70
14	2560	54.29	36.90	1.47	1.70
14	2880	85.68	56.05	1.52	1.70
14	3200	120.87	82.60	1.46	1.70
30	1440	10.69	6.63	1.61	1.85
30	2240	35.70	22.25	1.60	1.85
30	2560	56.68	37.29	1.52	1.85
30	3040	120.46	74.31	1.62	1.85
62	1920	25.10	16.38	1.53	1.93
62	2560	65.66	42.89	1.53	1.93
(b) The MST consists of $3G/2$ nodes from which $G/2$ are artificial.					
16	500	0.67	0.44	1.52	1.60
16	1000	3.97	2.56	1.55	1.60
16	1500	11.79	7.53	1.57	1.60
16	2000	27.44	17.88	1.55	1.60
28	1500	8.80	5.66	1.55	1.67
28	2000	18.88	11.99	1.57	1.67
28	2500	35.39	21.91	1.61	1.67
28	3000	62.71	37.71	1.66	1.67
40	2400	27.37	18.05	1.52	1.72
40	3200	62.92	38.31	1.64	1.72

5. Conclusion. Strategies for computing the QRD of the set of matrices $A_1, \ldots, A_G$ which have common columns have been considered. The first strategy computes the QRD of each matrix independently and does not exploit the relationship that may exist among the matrices. The second strategy expresses the matrix A_i as AS_i, where A consists of all the distinct columns of $A_1, \ldots, A_G$ and S_i is a column-selection matrix. Initially it computes the triangular factor R of the QRD of A. Then it derives the QRD of A_i by re-triangularizing RS_i $(i = 1, \ldots, G)$. This re-triangularization is equivalent to the multiple-column downdating of the QRD [9], [12]. The second strategy is found to have better complexity than the first.

The remaining novel strategies use a weighted directed graph to express the relationship (common columns) among the matrices. The nodes represent the triangular factors $R_1, \ldots, R_G$ derived from the QRDs of $A_1, \ldots, A_G$, respectively. An edge between two nodes exist if the columns of one of their corresponding matrices is a subset of the columns of the other. The weight of an edge is the computational cost of deriving the triangular factor of the subset matrix given the QRD of the larger matrix. The MST of this graph provides efficient strategies for computing the QRDs of $A_1, \ldots, A_G$ when

the columns of some of them are subsets of the columns of others. If no such matrices exist, then the MST is equivalent to the second strategy which derives $R_1, \ldots, R_G$ one at a time. This is offset by adding new (artificial) nodes which correspond to matrices constructed from the conjunction of columns of two or more matrices.

The algorithm for deriving the MST of the graph that includes all artificial nodes has double exponential complexity and is thus computationally intractable. A heuristic approach that reduces the complexity of the algorithm to polynomial time has been proposed. The performance of the heuristic method has been investigated in two cases, where it is most efficient. The numerical results indicate the superiority of this method compared with that of the second strategy which re-triangularizes $RS_1, \ldots, RS_G$ one at a time.

The re-triangularization of the matrices RS_i $(i = 1, \ldots, G)$ has been performed using Givens rotations. Householder transformations and block versions of Givens rotations can also be used [3], [13], [14], [18]. Furthermore, in some econometric models the data matrices $A_1, \ldots, A_G$ may have special structures and properties [2], [4]–[5],[6], [12]. In such cases the efficient re-triangularization of RS_i $(i = 1, \ldots, G)$ will require special algorithms. This will result in the edges of the directed graphs having different costs. However, the derivation of the MST and heuristic strategies for factorizing the matrices will remain the same. Currently, the adaptation of the proposed strategies to compute subset regression models is under investigation [7], [8].

Acknowledgement. The authors are grateful to Maurice Clint for his constructive comments and suggestions.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows*: *Theory*, *Algorithms*, *and Applications*. Prentice-Hall, Englewood Cliffs, NJ, 1993.

[2] P. Foschi, D. Belsley, and E. J. Kontoghiorghes. A comparative study of algorithms for solving seemingly unrelated regressions models. *Computational Statistic & Data Analysis*, 44(1–2):3–35, 2003.

[3] P. Foschi, L. Garin, and E. J. Kontoghiorghes. Numerical and computational methods for solving SUR models. In E. J. Kontoghiorghes, B. Rustem, and S.Siokos, editors, *Computational Methods in Decision-Making*, *Economics and Finance*, volume 74 of Applied Optimization, pages 405–427. Kluwer, Boston, MA, 2002.

[4] P. Foschi and E. J. Kontoghiorghes. Estimation of seemingly unrelated regression models with unequal size of observations: computational aspects. *Computational Statistics and Data Analysis*, 41(1):211–229, 2002.

[5] P. Foschi and E. J. Kontoghiorghes. Estimation of VAR models: computational aspects. *Computational Economics*, 21(1-2):3–22, 2003.

[6] P. Foschi and E. J. Kontoghiorghes. Estimating seemingly unrelated regression models with vector autoregressive disturbances. *Journal of Economic Dynamics and Control*, 28(1):27–44, 2003.

[7] C. Gatu and E. J. Kontoghiorghes. A branch and bound algorithm for computing the best subset regression models. Technical Report RT-2002/08-1, Institut d'informatique, Université de Neuchâtel, 2002.

[8] C. Gatu and E. J. Kontoghiorghes. Parallel algorithms for computing all possible subset regression models using the QR decomposition. *Parallel Computing*, (29):505–521, 2003.

[9] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 1974.

[10] G. H. Golub and C. F. Van Loan. *Matrix Computations*, 3rd edition. Johns Hopkins University Press, Baltimore, 1996.

[11] E. J. Kontoghiorghes. Parallel strategies for computing the orthogonal factorizations used in the estimation of econometric models. *Algorithmica*, 25:58–74, 1999.

[12] E. J. Kontoghiorghes. *Parallel Algorithms for Linear Models*: *Numerical Methods and Estimation Problems*, volume 15 of Advances in Computational Economics. Kluwer, Boston, MA, 2000.

[13] E. J. Kontoghiorghes. Parallel strategies for rank-k updating of the QR decomposition. *SIAM Journal on Matrix Analysis and Applications*, 22(3):714–725, 2000.

[14] E. J. Kontoghiorghes. Computational methods for modifying seemingly unrelated regressions models. *Journal of Computational and Applied Mathematics*, 162(1):247–261, 2004.

[15] E. J. Kontoghiorghes and E. Dinenis. Computing 3SLS solutions of simultaneous equation models with a possible singular variance–covariance matrix. *Computational Economics*, 10:231–250, 1997.

[16] J. B. Kruskal. On the shortest spanning tree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7:48–50, 1956.

[17] R. C. Prim. Shortest connection networks and some generalizations. *Bell Systems Technical Journal*, 36:1389–1401, 1957.

[18] P. Yanev and E. J. Kontoghiorghes. Efficient algorithms for block downdating of least squares solution. *Applied Numerical Mathematics*, 2004. In press.