Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Fast Moving Window Algorithm for QR and Cholesky Decompositions

Wuxing Liang * **Uwe Kruger**† ** **Xun Wang** * **Lei Xie** ***
Tim Littler *

* *School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, BT9 5AH, U.K. (e-mail: wliang02@qub.ac.uk; x.wang@ee.qub.ac.uk; t.littler@qub.ac.uk)*
** *Department of Electrical Engineering, The Petroleum Institute, P.O. Box 2533 , Abu Dhabi, U. A. E. (e-mail: ukruger@pi.ac.ae)*
*** *Institute of Advanced Process Control, Zhejiang University, Hangzhou, 310027, P.R. China (e-mail: leix@iipc.zju.edu.cn)*

Abstract: This paper proposes a fast moving window algorithm for QR and Cholesky decompositions by simultaneously applying data updating and downdating. The developed procedure is based on inner products and entails a similar downdating to that of the Chambers' approach. For adding and deleting one row of data from the original matrix, a detailed analysis shows that the proposed algorithm outperforms existing ones in terms or computational efficiency, if the number of columns exceeds 7. For a large number of columns, the proposed algorithm is numerically superior compared to the traditional sequential technique.

Keywords: Cholesky factorization; QR decomposition; adaptive algorithms; update and downdate; recursive algorithms.

1. INTRODUCTION

Matrix decomposition methods have had a significant impact upon linear algebra over the past decades. Stewart (2000) summarised a total of six most widely used decompositions, which are LU, Cholesky, QR, spectral decomposition, Schur decomposition, and singular value decomposition. This paper focuses on QR decomposition, which decomposes an $m \times n$ matrix into a product of an orthogonal matrix and an upper triangular matrix. The analysis in this paper also includes Cholesky decomposition which has a close relationship with QR decomposition in that it also produces a triangular matrix. Both decompositions have been found to play an important role in modern control theory (Landau, et al., 1998), signal processing (Diniz, 2002), and system identification (Overschee and Moor, 1996) for example.

For the QR decomposition of a matrix of n column and $m \leq n$ row vectors, the complexity of a complete QR decomposition is of $O(mn^2)$, whereas the cost for a Cholesky decomposition of a squared matrix of dimension n is $O(n^3)$ (Golub and Loan, 1996). This high computational effort renders an on-line adaptation for large matrices a difficult task (Liu, 1995). It is therefore desirable to develop fast adaptation algorithms that are of $O(n^2)$. Existing algorithms include (i) exponentially decreasing data weighting (usually referred to the recursive method) (Borow and Murray, 1993) and (ii) a sliding window of data (often defined as moving window) (Baykal and Constantinides, 1998). The main difference between these techniques lies in the length of data window and how old data are treated. The former one augments the data window each time a new data point becomes available. Old data are degraded in order to place more emphasis on newer information. In contrast, moving window approach discards the oldest sample each time a new one is added to the data window. The advantage of a moving window approach is that a constant speed of adaptation can be achieved, since the length of the data window remains constant. For recursive techniques, however, the data window constantly grows and it is often difficult to determine the most suitable exponential forgetting factor.

A recursive QR decomposition has been proposed in reference (Borow and Murray, 1993). Adding the impact of an additional data point to the original matrix produces an upper Hessenberg matrix, from which the upper triangular structure could be restored using a Givens transformation. Such a procedure is defined as "updating". Three techniques have been developed for removing the effect of an older sample from the original matrix, defined as "downdating". The hyperbolic transformation method (Golub, 1969) regards the elimination of one row as augmenting the triangular matrix by an additional row which is to be multiplied by $i = \sqrt{-1}$. The Chambers method (Chambers, 1971) that relies on rearranging the calculation orders of hyperbolic method improves the stability of the existing hyperbolic transformation method. The LINPACK method (P. E. Gill and Saunders, 1974) treats the downdating problem as a construction of an orthogonal transformation matrix which is obtained by solving a triangular system, and solves the problem in the same way as updating problem. Pan (1990) interlinked two

[1] †Correspondence author: Tel.: +971-2-607-5150, Fax.: +971-2-607-5200

different phases of the LINPACK method to produce a faster algorithm.

More recently, the LINPACK method was extended to produce a moving window algorithm for the QR decomposition (Strobach and Goryn, 1993). Based on the fact that the R-matrix of a QR decomposition can be updated in a similar fashion to the L-matrix of a Cholesky decomposition (P. E. Gill and Saunders, 1974), this paper develops of a moving window method that is applicable to both QR and Cholesky decompositions. Rather than updating and downdating sequentially in separate steps as presented in paper (Strobach and Goryn, 1993), this paper combines these two steps together and achieves a simultaneous adaptation for both decompositions, which, to the best of the authors' knowledge, has not been addressed in the existing research literature.

The QR and Cholesky decompositions will be briefly introduced in Section 2, along with existing methods on updating and downdating routines. Based on these methods, an improved fast moving window method for QR and Cholesky decompositions is proposed in Section 3. It is then compared with an alternative method in Section 4 in terms of complexity. Section 5 provides two simulation studies to test these methods, which experimentally checks the accuracy of the proposed method. The paper finally presents concluding remarks in Section 6.

2. PRELIMINARIES

This section gives a brief overview of QR and Cholesky decompositions and introduces the updating and downdating problems.

2.1 QR and Cholesky Decompositions

QR decomposition handles a given matrix $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{m\times n}$ $(m \geq n)$ of full column rank. There exists an orthogonal matrix $\boldsymbol{\mathcal{Q}} \in \mathbb{R}^{m\times m}$ and an upper triangular matrix $\boldsymbol{\mathcal{R}} \in \mathbb{R}^{n\times n}$ with positive diagonal elements, such that $\boldsymbol{\mathcal{Y}}$ can be decomposed as follows.

$$\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{Q}} \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{0} \end{bmatrix} \tag{1}$$

where $\mathbf{0} \in \mathbb{R}^{(m-n)\times n}$. It is equivalent to triangularize the matrix by reducing the elements below diagonal to zero step by step with orthogonal transformations such as Givens rotation or Householder reflection. Suppose diagonal elements of $\boldsymbol{\mathcal{R}}$ are positive, the decomposition is unique.

Cholesky decomposition works on a given symmetric and positive definite matrix $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n\times n}$, so that

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{L}}\boldsymbol{\mathcal{L}}^T = \boldsymbol{\mathcal{R}}^T\boldsymbol{\mathcal{R}} \tag{2}$$

where $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{n\times n}$ is a lower triangular matrix with strictly positive diagonal entries, and the upper triangular matrix $\boldsymbol{\mathcal{R}}$ is the transpose of $\boldsymbol{\mathcal{L}}$, denoted as $\boldsymbol{\mathcal{L}}^T$. Both $\boldsymbol{\mathcal{L}}$ and $\boldsymbol{\mathcal{R}}$ are unique under the assumption of positive diagonal elements.

It should be noted that there are similar upper triangular factors in the Cholesky and QR decompositions. More precisely, defining $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{Y}}^T\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{R}}^T\boldsymbol{\mathcal{R}}$, where the positive definiteness of $\boldsymbol{\mathcal{X}}$ is guaranteed by the full column rank of $\boldsymbol{\mathcal{Y}}$, and assuming that the diagonal elements are positive, the upper triangular matrix from QR decomposition on $\boldsymbol{\mathcal{Y}}$ is identical to that of the Cholesky decomposition of $\boldsymbol{\mathcal{X}}$. For simplicity, both upper triangular matrices are denoted by $\boldsymbol{\mathcal{R}}$ throughout this article. Moreover, the combined adaptation algorithm is applicable to both, the QR and Cholesky decompositions.

To adapt a matrix decomposition based on the moving window approach, each factor matrix needs to be updated with new data and downdated by discarding the oldest data. If the Q factor of QR decomposition is not required to be stored and updated, the adaptation task for the Cholesky and QR decompositions are based on the R factor matrix (Lawson and Hanson, 1974).

2.2 QR and Cholesky Updating Problem

The updating problem refers to the construction of the new factor based on the old one and a new sample. With Cholesky decomposition, the upper triangular matrix $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{n\times n}$ is to be determined based on the former upper triangular matrix $\boldsymbol{\mathcal{R}} \in \mathbb{R}^{n\times n}$ and a vector $\mathbf{x} \in \mathbb{R}^n$, so that

$$\boldsymbol{\mathcal{T}}^T\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{R}}^T\boldsymbol{\mathcal{R}} + \mathbf{x}\mathbf{x}^T \tag{3}$$

Let $\boldsymbol{\mathcal{Z}} = \boldsymbol{\mathcal{X}} + \mathbf{x}\mathbf{x}^T$ following the denotations in Equation (2), which is a rank-1 modification to $\boldsymbol{\mathcal{X}}$. The symmetry and positive definiteness of $\boldsymbol{\mathcal{Z}}$ still hold. This verifies the existence and uniqueness of factor $\boldsymbol{\mathcal{T}}$ from Cholesky decomposition of $\boldsymbol{\mathcal{Z}}$.

To show the updating procedure more explicitly, the influence of new data is studied based on Equation (1). Without loss of generality, $\mathbf{x}^T$ is assumed to be appended at the bottom of $\boldsymbol{\mathcal{Y}}$, resulting in matrix $\boldsymbol{\mathcal{Y}}^*$ that can be partitioned as below.

$$\begin{aligned}\boldsymbol{\mathcal{Y}}^* = \begin{bmatrix} \boldsymbol{\mathcal{Y}} \\ \mathbf{x}^T \end{bmatrix} &= \begin{bmatrix} \boldsymbol{\mathcal{Q}} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{0} \\ \mathbf{x}^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mathcal{Q}} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} (\boldsymbol{\mathcal{P}}^T\boldsymbol{\mathcal{P}}) \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{0} \\ \mathbf{x}^T \end{bmatrix} \\ &= \left(\begin{bmatrix} \boldsymbol{\mathcal{Q}} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \boldsymbol{\mathcal{P}}^T \right) \left(\boldsymbol{\mathcal{P}} \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{0} \\ \mathbf{x}^T \end{bmatrix} \right) = \boldsymbol{\mathcal{Q}}^{**} \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{0} \\ \mathbf{x}^T \end{bmatrix}\end{aligned} \tag{4}$$

where $\boldsymbol{\mathcal{P}} \in \mathbb{R}^{(m+1)\times(m+1)}$ is permutation matrix to switch $\mathbf{x}^T$ to the bottom of $\boldsymbol{\mathcal{R}}$. Obviously, $\boldsymbol{\mathcal{P}}^T\boldsymbol{\mathcal{P}} = \boldsymbol{\mathcal{I}}_{m+1}$, where $\boldsymbol{\mathcal{I}}_{m+1}$ is identity matrix of dimension $m+1$. Then $\boldsymbol{\mathcal{Q}}^{**}$ is still orthogonal. Continue to conduct the QR decomposition as follows,

$$\boldsymbol{\mathcal{Y}}^* = \boldsymbol{\mathcal{Q}}^{**} \left(\begin{bmatrix} \boldsymbol{\mathcal{Q}}_a & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mathcal{I}} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{\mathcal{Q}}_a & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mathcal{I}} \end{bmatrix} \right) \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{x}^T \\ \mathbf{0} \end{bmatrix} = \boldsymbol{\mathcal{Q}}^* \begin{bmatrix} \boldsymbol{\mathcal{T}} \\ \mathbf{0}^T \end{bmatrix} \tag{5}$$

where $\boldsymbol{\mathcal{T}}$ is finally achieved, an orthogonal matrix $\boldsymbol{\mathcal{Q}}_a$ is introduced in Equation (5) which fulfills the following requirements: (i) $\boldsymbol{\mathcal{Q}}_a^T\boldsymbol{\mathcal{Q}}_a = \boldsymbol{\mathcal{I}}_{n+1}$; (ii) $\boldsymbol{\mathcal{Q}}^* = \boldsymbol{\mathcal{Q}}^{**} \begin{bmatrix} \boldsymbol{\mathcal{Q}}_a & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\mathcal{I}} \end{bmatrix}^T$ maintains the orthogonality; and (iii)

$$\boldsymbol{\mathcal{Q}}_a \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{x}^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mathcal{T}} \\ \mathbf{0}^T \end{bmatrix} \tag{6}$$

Since the elements of $\boldsymbol{\mathcal{T}}$ and $\mathbf{x}$ are known, $\boldsymbol{\mathcal{Q}}_a$ can be constructed by Givens rotation which aims to eliminate the added data $\mathbf{x}$ step by step. The derivatives here are

different with traditional method (Strobach and Goryn, 1993) by placing $\mathbf{x}^T$ on the bottom of the original matrix not on the top of the original matrix $\boldsymbol{\mathcal{R}}$. For ease of expression, $\mathbf{e}_j^T\boldsymbol{\mathcal{M}}$ is used to denote the j^{th} row of any matrix $\boldsymbol{\mathcal{M}}$ where $\mathbf{e}_j$ is a vector with 1 in position j and zeros elsewhere. Superscripts represent the times of transformation. Specially, $\mathbf{x}^{(0)} = \mathbf{x}$. Then

$$\boldsymbol{\mathcal{G}}_a^{(i)} \begin{bmatrix} \mathbf{e}_i^T\boldsymbol{\mathcal{R}} \\ (\mathbf{x}^{(i-1)})^T \end{bmatrix} = \begin{bmatrix} \mathbf{e}_i^T\boldsymbol{\mathcal{T}} \\ (\mathbf{x}^{(i)})^T \end{bmatrix} \tag{7}$$

where

$$\boldsymbol{\mathcal{G}}_a^{(i)} = \begin{bmatrix} c_a^{(i)} & -s_a^{(i)} \\ s_a^{(i)} & c_a^{(i)} \end{bmatrix} \tag{8}$$

$$c_a^{(i)} = r_{ii}/l_a^{(i)} \tag{9}$$

$$-s_a^{(i)} = x_i^{(i-1)}/l_a^{(i)} \tag{10}$$

$$l_a^{(i)} = [r_{ii}^2 + x_i^{(i-1)}]^{1/2} \tag{11}$$

Then $\boldsymbol{\mathcal{Q}}_a^{(i)} \in \mathbb{R}^{(n+1)\times(n+1)}$ can be constructed as

$$\boldsymbol{\mathcal{Q}}_a^{(i)} = \begin{bmatrix} \boldsymbol{\mathcal{I}}_{i-1} & 0 & \cdots & 0 \\ 0 & c_a^{(i)} & \cdots & -s_a^{(i)} \\ \vdots & \vdots & \boldsymbol{\mathcal{I}}_{n-i} & \vdots \\ 0 & s_a^{(i)} & \cdots & c_a^{(i)} \end{bmatrix} \tag{12}$$

The Givens transformation matrix can be constructed row by row and finally $\boldsymbol{\mathcal{Q}}_a = \boldsymbol{\mathcal{Q}}_a^{(n)} \cdots \boldsymbol{\mathcal{Q}}_a^{(2)}\boldsymbol{\mathcal{Q}}_a^{(1)}$. The calculated $\boldsymbol{\mathcal{T}}$ is exactly the Cholesky factor in Equation (3).

2.3 QR and Cholesky Downdating Problem

The Cholesky downdating problem is the inverse of the updating problem: to find $\boldsymbol{\mathcal{R}}$ from $\boldsymbol{\mathcal{T}}$ and $\mathbf{x}$ by rearranging $\mathbf{x}\mathbf{x}^T$ to the left side of Equation (3). For ease of description, here $\boldsymbol{\mathcal{U}}$ and $\mathbf{y}$ are introduced to represent the unknown matrix and the sample to be deleted respectively. If $\mathbf{x} = \mathbf{y}$, then $\boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{R}}$. The downdating problem is formulated as follows.

$$\boldsymbol{\mathcal{U}}^T\boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{T}}^T\boldsymbol{\mathcal{T}} - \mathbf{y}\mathbf{y}^T \tag{13}$$

Among the three downdating methods mentioned in Section 1, the Chambers method (Chambers, 1971) is applied here because it is suitable to parallel processing with better numerical stability comparing with LINPACK method and hyperbolic transformation method (A. W. Bojanczyk and Hoog, 1987).

Inspired by the concept used for Cholesky updating problem, $\boldsymbol{\mathcal{Q}}_d$ is introduced to form the following relation similar to Equation (6).

$$\boldsymbol{\mathcal{Q}}_d \begin{bmatrix} \boldsymbol{\mathcal{U}} \\ \mathbf{y}^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mathcal{T}} \\ \mathbf{0}^T \end{bmatrix} \tag{14}$$

The target is to discover relationships to formulate the elements in $\boldsymbol{\mathcal{U}}$. Similar with $\boldsymbol{\mathcal{G}}_a^1$, $\boldsymbol{\mathcal{G}}_d^{(1)}$ transform $\begin{bmatrix} \mathbf{e}_1^T\boldsymbol{\mathcal{U}} \\ (\mathbf{y}^{(0)})^T \end{bmatrix}$ to $\begin{bmatrix} \mathbf{e}_1^T\boldsymbol{\mathcal{T}} \\ (\mathbf{y}^{(1)})^T \end{bmatrix}$. Noticing that $y_1^{(1)} = 0$, the following relationship holds.

$$(l_d^{(1)})^2 = u_{11}^2 + (y_1^{(0)})^2 = t_{11}^2 \tag{15}$$

from which u_{11} can be obtained as follows.

$$u_{11} = [t_{11}^2 - (y_1^{(0)})^2]^{1/2} = [(t_{11} + y_1^{(0)})(t_{11} - y_1^{(0)})]^{1/2} \tag{16}$$

By applying

$$c_d^{(1)}u_{1j} - s_d^{(1)}y_j^{(0)} = t_{1j} \tag{17}$$

the remaining elements of the first row ($2 \le j \le n$) can be calculated as follows.

$$u_{1j} = [t_{1j} - (-s_d^{(1)})y_j^{(0)}]/c_d^{(1)} \tag{18}$$

And the relationship between $\mathbf{y}^{(0)}$ and $\mathbf{y}^{(1)}$ is obtained from the second row of augmented matrix.

$$y_j^{(1)} = c_d^{(1)}y_j^{(0)} - (-s_d^{(1)})u_{1j} \tag{19}$$

In general, at the i^{th} step,

$$u_{ii} = [(t_{ii} + y_i^{(i-1)})(t_{ii} - y_i^{(i-1)})]^{1/2} \tag{20}$$

$$u_{ij} = [t_{ij} - (-s_d^{(i)})y_j^{(i-1)}]/c_d^{(i)} \quad i+1 \le j \le n \tag{21}$$

$$y_j^{(i)} = c_d^{(i)}y_j^{(i-1)} - (-s_d^{(i)})u_{ij} \tag{22}$$

$$\tag{23}$$

where

$$c_d^{(i)} = u_{ii}/l_d^{(i)} \tag{24}$$

$$-s_d^{(i)} = y_i^{(i-1)}/l_d^{(i)} \tag{25}$$

$$l_d^{(i)} = t_{ii} \tag{26}$$

Once all the n steps are taken, $\boldsymbol{\mathcal{U}}$ can be finally completed.

3. PROPOSED MOVING WINDOW ALGORITHM

This section introduces the combined moving window approach. Commencing with a problem formulation, Subsection 3.2 gives details of the combined algorithm.

3.1 Problem Formulation

Since techniques are available for both updating and downdating steps, as outlined in Section 2, it is straightforward to carry out the two steps sequentially. This paper develops a faster moving window algorithm by combining these two steps together. Thus, the procedure of a two-step rank-1 modification is changed to a single step rank-2 modification. This means from triangular matrix $\boldsymbol{\mathcal{R}} \in \mathbb{R}^{n\times n}$ and vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, the triangular matrix $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{n\times n}$ is obtained directly.

$$\boldsymbol{\mathcal{U}}^T\boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{R}}^T\boldsymbol{\mathcal{R}} + \mathbf{x}\mathbf{x}^T - \mathbf{y}\mathbf{y}^T \tag{27}$$

3.2 Moving Window QR and Cholesky Decompositions

Lemma 1 below shows that a solution for Equation (27) can be obtained, that is further refined to be the combined up- and downdating algorithm.

Lemma 1. Given that the old matrix $\boldsymbol{\mathcal{R}}$, the new sample $\mathbf{x}$ and the old sample $\mathbf{y}$ are available, the new upper triangular matrix $\boldsymbol{\mathcal{U}}$ can be determined as follows:

$$u_{ii} = \left[\langle \mathbf{a}_i, \mathbf{a}_i \rangle - y_i^2 - \sum_{k=1}^{i-1} u_{ki}^2\right]^{1/2} \tag{28}$$

$$u_{ij} = \frac{1}{u_{ii}}\left(\langle \mathbf{a}_i, \mathbf{a}_j \rangle - y_i y_j - \sum_{k=1}^{i-1} u_{ki}u_{kj}\right) \tag{29}$$

Here, $\langle \cdot, \cdot \rangle$ represents the inner product of two vectors. A proof of Lemma 1 is given in Appendix A.

Based on Lemma 1, the up- and downdating steps can be combined to produce a numerically more efficient algorithm for the Cholesky and QR decompositions. The combined algorithm is summarised by the following theorem, which is proven in Appendix B.

Theorem 2. Elements of $\boldsymbol{\mathcal{U}}$ can be expressed as follows.

$$\begin{aligned} u_{ii} &= [r_{ii}^2 + (x_i^{(i-1)})^2 - (y_i^{(i-1)})^2]^{1/2} \\ &= [(l_a^{(i)} + y_i^{(i-1)})(l_a^{(i)} - y_i^{(i-1)})]^{1/2} \end{aligned} \tag{30}$$

$$\begin{aligned} u_{ij} &= \frac{1}{u_{ii}}[r_{ii}r_{ij} + x_i^{(i-1)}x_j^{(i-1)} - y_i^{(i-1)}y_j^{(i-1)}] \\ &= \frac{r_{ii}}{u_{ii}}r_{ij} + \frac{x_i^{(i-1)}}{u_{ii}}x_j^{(i-1)} - \frac{y_i^{(i-1)}}{u_{ii}}y_j^{(i-1)} \end{aligned} \tag{31}$$

$$x_j^{(i)} = c_a^{(i)}x_j^{(i-1)} - (-s_a^{(i)})r_{ij} \tag{32}$$

$$y_j^{(i)} = c_d^{(i)}y_j^{(i-1)} - (-s_d^{(i)})u_{ij} \tag{33}$$

where $1 \le i \le n$, $i+1 \le j \le n$ and

$$l_a^{(i)} = [r_{ii}^2 + (x_i^{(i-1)})^2]^{1/2} \tag{34}$$

$$c_a^{(i)} = r_{ii}/l_a^{(i)} \tag{35}$$

$$-s_a^{(i)} = x_j^{(i-1)}/l_a^{(i)} \tag{36}$$

$$l_d^{(i)} = [u_{ii}^2 + (y_i^{(i-1)})^2]^{1/2} = l_a^{(i)} \tag{37}$$

$$c_d^{(i)} = u_{ii}/l_d^{(i)} \tag{38}$$

$$-s_d^{(i)} = y_i^{(i-1)}/l_d^{(i)} \tag{39}$$

4. COMPLEXITY ANALYSIS

The computational aspect of the combination algorithm is investigated in this section. The numbers of flops involved during the procedure of the proposed algorithm are shown in Table 1. It follows from Table 1 that the total number of flops is equal to $5.5n^2 + 9.5n$. And the total flops of sequential algorithm is $6n^2 + 6n$ (referred to Section 2, the flops of i^{th} step of sequential algorithm is $12(n - i) + 12$). This, however, implies that the combined up- and downdating steps compare favourably, that is $6n^2 + 6n > 5.5n^2 + 9.5n$, for $n > 7$. In practice, it is numerically more efficient to employ the combined steps for large scale applications, whilst the sequential updating is superior for smaller matrices ≤ 7 variables.

In the other extreme case for very large matrices, that is $n \to \infty$ the theoretical savings can amount to over 8%, since $1 - \lim_{n\to\infty} \frac{5.5n^2+9.5n}{6n^2+6n} \approx 0.083$.

5. NUMERICAL ACCURACY OF THE PROPOSED MOVING WINDOW ALGORITHM

This section summarises an application study of the moving window technique detailed in Section 3. The aim of this investigation is to evaluate its numerical accuracy. This is of importance since it can often be observed that an increase in computational efficiency compromises accuracy. Next, details of the simulation example are given prior to the analysis for one shift of the moving window followed by multiple shifts.

Procedure	Determine	Equation to be used	flops
1	$l_a^{(i)}$	(34)	4
	$c_a^{(i)}$	(35)	1
	$-s_a^{(i)}$	(36)	1
2	u_{ii}	(30)	4
3	$l_d^{(i)}$	(37)	0
	$c_d^{(i)}$	(38)	1
	$-s_d^{(i)}$	(39)	1
4	r_{ii}/u_{ii}	-	1
	$x_i^{(i-1)}/u_{ii}$	-	1
	$y_i^{(i-1)}/u_{ii}$	-	1
5	u_{ij}	(31)	$5(n-i)$
6	$\mathbf{x}^{(i)}$	(32)	$3(n-i)$
	$\mathbf{y}^{(i)}$	(33)	$3(n-i)$
Total	-	-	$11(n-i)+15$

Table 1. Procedure of algorithm proposed and flops required

5.1 Tests Design

The elements of analysed matrices were generated randomly and Monte Carlo experiments were carried out to evaluate general trends. For simplicity the number of columns was set to $n = 100$ and a date set of $m = 2200$ random entries $\mathcal{Y}_{mn} \in \mathcal{N}(0, 1)$ was generated. An initial QR decomposition was carried out on the first 200 samples. The conventional sequential and the proposed combined moving window approaches were then applied for samples 201 to 2200. To circumvent that random deviations arise, a total of 1000 Monte Carlo experiments were carried out, that is a total of 1000 data sets of $n = 100$ and $m = 2200$ were generated.

The numerical accuracy of both moving window techniques has been compared as follows. A QR decomposition using Givens rotations was obtained for the data within a window that was moved forward by one step. This R matrix computed was then compared to that obtained by both moving window techniques and the relative error, e, was evaluated as follows:

$$e = \frac{\|\boldsymbol{\mathcal{U}} - \boldsymbol{\mathcal{R}}\|_F}{\|\boldsymbol{\mathcal{R}}\|_F} \tag{40}$$

where F refers to the Frobenius norm of a matrix. It should be noted that since the accuracy analysis is based on 1000 Monte Carlo experiments, the mean relative error, $\overline{e} = \frac{1}{1000}\sum_{\kappa=1}^{1000} e_\kappa$ is considered below.

5.2 Matrix Adaptation using One Step

In this study, the window of dimension $m = 200$ is only moved forward by one step. Applying both moving window approaches generated one updated R matrix each and the application of the traditional QR decomposition produced the benchmark R matrix. Utilising Equation (40) gave rise to an error value of 6.527×10^{-16} and 6.514×10^{-16} for the sequential and the combined moving window algorithms, respectively. Altering the number of column vectors to 50, 150, and 500 produced a similar outcome. This suggests that the combined approach is numerically more accurate than the conventional sequential one. Although these observations are based on simulation examples only, this outcome is expected since the combined algorithm requires fewer floating point operations to be carried out, which, in

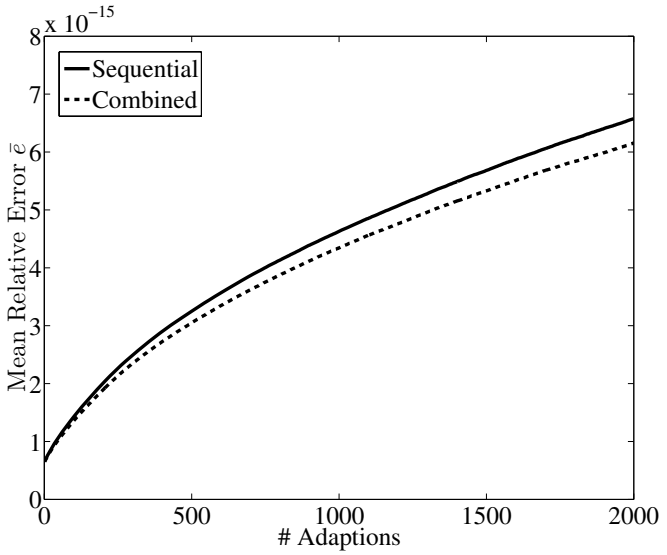

Fig. 1. Error Comparison Curve

turn, can reduce numerical errors introduced by floating point operations. For future work, a detailed error analysis for example backward error analysis (Wilkinson, 1965) needs to be conducted to verify these findings.

5.3 Multi-Step Test

Comparing with the one step test in Subsection 5.2, the multi-step test here took into account of error accumulation. As shown in Figure 1, the final error of new algorithm is smaller than that of the conventional one. It should be noted that the new algorithm shows not only computationally more efficient but also more accurate.

6. CONCLUSION

This paper has introduced a fast moving window algorithm for Cholesky and QR decompositions. Compared to conventional up- and downdating methods, discussed in the research literature, the proposed algorithm combines these steps. The analysis in this paper has shown that the proposed combined technique is computationally superior, as it requires fewer floating point operation. As the moving window slides one step forward, it can save in the region of $O(0.5n^2)$ floating point operations, which, in turn, results in an 8 % decrease in computational effort if the number of columns exceeds 7.

The numerical accuracy of the new approach has been compared with the conventional sequential application of up- and downdating in a simulation example. This application has found that the new approach provides computational improvement without compromising the numerical accuracy. Moreover, the analysis has indicated that the combined algorithm produces a smaller numerical error. The numerical accuracy issue, however, needs to be verified by a detailed error analysis in the future.

Appendix A. PROOF OF LEMMA 1

Proof: Let $\boldsymbol{\mathcal{A}} = \begin{bmatrix} \boldsymbol{\mathcal{R}} \\ \mathbf{x}^T \end{bmatrix} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_n]$, $\boldsymbol{\mathcal{B}} = \begin{bmatrix} \boldsymbol{\mathcal{U}} \\ \mathbf{y}^T \end{bmatrix} = [\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_n]$, and $\boldsymbol{\mathcal{C}} = \begin{bmatrix} \boldsymbol{\mathcal{T}} \\ \mathbf{0}^T \end{bmatrix} = [\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_n]$, where $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{n \times n}$ is intermediate matrix and $\mathbf{a}_i$, $\mathbf{b}_i$, and $\mathbf{c}_i$ are column vectors of $\boldsymbol{\mathcal{A}}$, $\boldsymbol{\mathcal{B}}$, and $\boldsymbol{\mathcal{C}}$ respectively. According to analysis in Section 2, during updating and downdating, the orthogonal matrices $\boldsymbol{\mathcal{Q}}_a^{(i)}$ and $\boldsymbol{\mathcal{Q}}_d^{(i)}$ can always be constructed. The final orthogonal matrices are $\boldsymbol{\mathcal{Q}}_a = \boldsymbol{\mathcal{Q}}^{(n)}\boldsymbol{\mathcal{Q}}_a^{(n-1)} \cdots \boldsymbol{\mathcal{Q}}^{(1)}$ and $\boldsymbol{\mathcal{Q}}_d = \boldsymbol{\mathcal{Q}}_d^{(n)}\boldsymbol{\mathcal{Q}}_d^{(n-1)} \cdots \boldsymbol{\mathcal{Q}}_d^{(1)}$ respectively, which is equivalent to carrying out n orthogonal transform operations to reduce the augmented matrices to triangular structure step by step. Since $\boldsymbol{\mathcal{Q}}_a\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{C}}$ and $\boldsymbol{\mathcal{Q}}_d\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{C}}$, it can be derived that

$$\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{Q}}_d^T\boldsymbol{\mathcal{C}} = \boldsymbol{\mathcal{Q}}_d^T(\boldsymbol{\mathcal{Q}}_a\boldsymbol{\mathcal{A}}) = (\boldsymbol{\mathcal{Q}}_d^T\boldsymbol{\mathcal{Q}}_a)\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{Q}}\boldsymbol{\mathcal{A}} \qquad \text{(A.1)}$$

Because of the orthogonality of $\boldsymbol{\mathcal{Q}}$, the inner product of the columns in $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ has the following relationship.

$$\langle \mathbf{a}_i, \mathbf{a}_j \rangle = \langle \mathbf{b}_i, \mathbf{b}_j \rangle \qquad 1 \le i \le j \le n \qquad \text{(A.2)}$$

There are exactly C_n^2 equations and C_n^2 unknowns. By expanding the equations and noticing the special structure of $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$, the elements of $\boldsymbol{\mathcal{U}}$ can be solved row by and row and expressed as follows.

$$u_{ii} = \left[\langle \mathbf{a}_i, \mathbf{a}_i \rangle - y_i^2 - \sum_{k=1}^{i-1} u_{ki}^2 \right]^{1/2} \qquad \text{(A.3)}$$

$$u_{ij} = \frac{1}{u_{ii}} \left(\langle \mathbf{a}_i, \mathbf{a}_j \rangle - y_i y_j - \sum_{k=1}^{i-1} u_{ki} u_{kj} \right) \qquad \text{(A.4)}$$

where $1 \le i \le n$ for Equation (A.3) and $1 \le i \le n; i+1 \le j \le n$ for Equation (A.4). □.

Appendix B. PROOF OF THEOREM 2

Proof: This proof follows the rule of induction which starts from the first element $\boldsymbol{\mathcal{U}}$. Applying Equation (A.2) to the first columns of $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ gives $\langle \mathbf{a}_1, \mathbf{a}_1 \rangle = \langle \mathbf{b}_1, \mathbf{b}_1 \rangle$. A more detailed equation expression reveals the following relationship concerning the only two elements in vectors $\mathbf{a}_1$ and $\mathbf{b}_1$.

$$r_{11}^2 + (x_1^{(0)})^2 = u_{11}^2 + (y_1^{(0)})^2 \qquad \text{(B.1)}$$

Hence, u_{11} can be calculated as follows, which matches Equation (30) in the case of $i = 1$.

$$u_{11} = [r_{11}^2 + (x_1^{(0)})^2 - (y_1^{(0)})^2]^{1/2} = [(l_a^{(1)} + y_1^{(0)})(l_a^{(1)} - y_1^{(0)})]^{1/2} \qquad \text{(B.2)}$$

where $(l_a^{(1)})^2 = r_{11}^2 + (x_1^{(0)})^2$. By applying $\langle \mathbf{a}_1, \mathbf{a}_j \rangle = \langle \mathbf{b}_1, \mathbf{b}_j \rangle$ $(2 \le j \le n)$ repeatedly, the remaining elements in the first row of $\boldsymbol{\mathcal{U}}$ can be obtained as follows, which validates Equation (31) in the case of $i = 1$.

$$\begin{aligned} u_{1j} &= \frac{1}{u_{11}}(r_{11}r_{1j} + x_1^{(0)}x_j^{(0)} - y_1^{(0)}y_j^{(0)}) \\ &= \frac{r_{11}}{u_{11}}r_{1j} + \frac{x_1^{(0)}}{u_{11}}x_j^{(0)} - \frac{y_1^{(0)}}{u_{11}}y_j^{(0)} \end{aligned} \qquad \text{(B.3)}$$

Using the compact matrix representation of Equation (A.1) as follows.

$$\boldsymbol{\mathcal{G}}_a^{(1)} \begin{bmatrix} \mathbf{e}_1^T \boldsymbol{\mathcal{R}} \\ (\mathbf{x}^{(0)})^T \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1^T \boldsymbol{\mathcal{T}} \\ (\mathbf{x}^{(1)})^T \end{bmatrix} \tag{B.4}$$

$$\boldsymbol{\mathcal{G}}_d^{(1)} \begin{bmatrix} \mathbf{e}_1^T \boldsymbol{\mathcal{U}} \\ (\mathbf{y}^{(0)})^T \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1^T \boldsymbol{\mathcal{T}} \\ (\mathbf{y}^{(1)})^T \end{bmatrix} \tag{B.5}$$

where $\boldsymbol{\mathcal{G}}_a^{(1)} = \begin{bmatrix} c_a^{(1)} & -s_a^{(1)} \\ s_a^{(1)} & c_a^{(1)} \end{bmatrix}$ and $\boldsymbol{\mathcal{G}}_d^{(1)} = \begin{bmatrix} c_d^{(1)} & -s_d^{(1)} \\ s_d^{(1)} & c_d^{(1)} \end{bmatrix}$, which can transform $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ to intermediate matrix $\boldsymbol{\mathcal{C}}$. Then, the data vectors $\mathbf{x}$ and $\mathbf{y}$ after orthogonal transformation can be given as follows.

$$x_j^{(1)} = c_a^{(1)} x_j^{(0)} - (-s_a^{(1)}) r_{1j} \tag{B.6}$$

$$y_j^{(1)} = c_d^{(1)} y_j^{(0)} - (-s_d^{(1)}) u_{1j} \tag{B.7}$$

where $2 \leq j \leq n$. The elements of these Givens matrices can be constructed as standard method.

$$l_a^{(1)} = [r_{11}^2 + (x_1^{(0)})^2]^{1/2} \tag{B.8}$$

$$c_a^{(1)} = r_{11}/l_a^{(1)} \tag{B.9}$$

$$-s_a^{(1)} = x_1^{(0)}/l_a^{(1)} \tag{B.10}$$

$$l_d^{(1)} = [u_{11}^2 + (y_1^{(0)})^2]^{1/2} \tag{B.11}$$

$$c_d^{(1)} = u_{11}/l_d^{(1)} \tag{B.12}$$

$$-s_d^{(1)} = y_1^{(0)}/l_d^{(1)} \tag{B.13}$$

These proves that Equations (34) to (39) are valid. By now, the theorem has been proved for the case of $i = 1$.

Now suppose that the theorem holds for $i = k$, which means that the first k rows of following matrices are the same.

$$\mathbf{e}_i^T (\boldsymbol{\mathcal{Q}}_a^{(i)} \dots \boldsymbol{\mathcal{Q}}_a^{(1)} \boldsymbol{\mathcal{A}}) = \mathbf{e}_i^T (\boldsymbol{\mathcal{Q}}_d^{(i)} \dots \boldsymbol{\mathcal{Q}}_d^{(1)} \boldsymbol{\mathcal{B}}) \quad 1 \leq i \leq k \tag{B.14}$$

For $i = k+1$, $u_{k+1,k+1}$ can still be derived from the inner product of $\langle \mathbf{a}_{k+1}^{(k)}, \mathbf{a}_{k+1}^{(k)} \rangle = \langle \mathbf{b}_{k+1}^{(k)}, \mathbf{b}_{k+1}^{(k)} \rangle$. It can be expanded the equation to the following expression.

$$\begin{aligned} &\sum_{l=1}^{k} t_{l,k+1}^2 + r_{k+1,k+1}^2 + (x_{k+1}^{(k)})^2 \\ &= \sum_{l=1}^{k} t_{l,k+1}^2 + u_{k+1,k+1}^2 + (y_{k+1}^{(k)})^2 \end{aligned} \tag{B.15}$$

which can be simplified to

$$\begin{aligned} u_{k+1,k+1} &= [r_{k+1,k+1}^2 + (x_{k+1}^{(k)})^2 - (y_{k+1}^{(k)})^2]^{1/2} \\ &= [(l_a^{(k+1)} + y_{k+1}^{(k)})(l_a^{(k+1)} - y_{k+1}^{(k)})]^{1/2} \end{aligned} \tag{B.16}$$

where $(l_a^{(k+1)})^2 = r_{k+1,k+1}^2 + (x_{k+1}^{(k)})^2$. By following the same procedure as in the case of $i = 1$, the remaining unknown elements in the $(k+1)^{th}$ row can be calculated as

$$\begin{aligned} u_{k+1,j} &= \frac{1}{u_{k+1,k+1}} (r_{k+1,k+1} r_{k+1,j} + x_{k+1}^{(k)} x_j^{(k)} - y_{k+1}^{(k)} y_j^{(k)}) \\ &= \frac{r_{k+1,k+1}}{u_{k+1,k+1}} r_{k+1,j} + \frac{x_{k+1}^{(k)}}{u_{k+1,k+1}} x_j^{(k)} - \frac{y_{k+1}^{(k)}}{u_{k+1,k+1}} y_j^{(k)} \end{aligned} \tag{B.17}$$

where $k+2 \leq j \leq n$. Similarly, the rest of equations in Theorem can be all validated for $i = k+1$. Therefore, the proof is complete for $1 \leq i \leq n$. □

ACKNOWLEDGEMENTS

The first author would like to thank the Secretary of State for Education and Science, U.K. for the financial support of ORS scholarship to undertake this research.

REFERENCES

A. W. Bojanczyk, P. Van Dooren, R. P. Brent, and F. R. De Hoog. A note on downdating the cholesky factorization. *SIAM J. SCI STAT COMPUT*, 8(3):210–221, May 1987.

B. Baykal and A. G. Constantinides. Sliding window adaptive fast qr and qr-lattice algorithms. *IEEE Trans. on Signal Processing*, 46(11):2877–2887, November 1998.

J. E. Borow and W. Murray. An algorithm for rls idnetification of parameters that vary quickly with time. *IEEE Trans. Automatic Control*, 38(2):351–354, February 1993.

J. M. Chambers. Regression updating. *Journal of the American Statistical Association*, 66(336):744–748, December 1971.

Paulo S. R. Diniz. *Adaptive filtering: algorithms and practical implementation*. Kluwer Academic, Boston; London, second edition, 2002.

G. H. Golub. Matrix decompositions and statistical calculations. In R. C. Milton and J. A. Nelder, editors, *Statistical Computation*, pages 365–397, The University of Wisconsin, April 1969. Academic Press.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore and London, third edition, 1996.

I. D. Landau, R. Lozano, and M. M'Saad. *Adaptive control*. Springer, Berlin; New York; London, 1998.

C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Prentice-Hall, INC, Englewood Cliffs, New Jersey, 1974.

Z. S. Liu. QR methods of O(n) complexity in adaptive parameter estimation. *IEEE Trans. Signal Processing*, 43:720–729, 1995.

P. Van Overschee and B. De Moor. *Subspace Identification for Linear Systems: Theory-Implementation-Applications*. Kluwer Academic Publishers, Boston, 1996.

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

C. -T. Pan. A modification to the linpack downdating algorithm. *BIT*, 30:702–722, 1990.

G. W. Stewart. The decompositional approach to matrix computation. *Computing in Science and Engineering*, 2 (1):50–59, January 2000.

P. Strobach and D. Goryn. A computation of the sliding window recursive qr decomposition. In *Proc. ICASSP*, pages 29–32, 1993.

J.H. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1965.