

## Numerical strategies for recursive least squares solutions to the matrix equation $AX = B$

Stella Hadjiantoni & George Loizou

To cite this article: Stella Hadjiantoni & George Loizou (2022): Numerical strategies for recursive least squares solutions to the matrix equation  $AX = B$ , International Journal of Computer Mathematics, DOI: [10.1080/00207160.2022.2123220](https://doi.org/10.1080/00207160.2022.2123220)

To link to this article: <https://doi.org/10.1080/00207160.2022.2123220>



© 2022 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



Published online: 21 Oct 2022.



Submit your article to this journal [↗](#)



Article views: 117



View related articles [↗](#)



View Crossmark data [↗](#)

# Numerical strategies for recursive least squares solutions to the matrix equation $AX = B$

Stella Hadjiantoni<sup>a</sup> and George Loizou<sup>b</sup>

<sup>a</sup>Department of Mathematical Sciences, University of Essex, Colchester, Essex, UK; <sup>b</sup>Department of Computer Science and Information Systems, Birkbeck, University of London, London, UK

## ABSTRACT

The recursive solution to the Procrustes problem –with or without constraints– is thoroughly investigated. Given known matrices  $A$  and  $B$ , the proposed solution minimizes the square of the Frobenius norm of the difference  $AX - B$  when rows or columns are added to  $A$  and  $B$ . The proposed method is based on efficient strategies which reduce the computational cost by utilizing previous computations when new data are acquired. This is particularly useful in the iterative solution of an unbalanced orthogonal Procrustes problem. The results show that the computational efficiency of the proposed recursive algorithms is more significant when the dimensions of the matrices are large. This demonstrates the usefulness of the proposed algorithms in the presence of high-dimensional data sets. The practicality of the new method is demonstrated through an application in machine learning, namely feature extraction for image processing.

## ARTICLE HISTORY

Received 31 January 2022  
Revised 22 June 2022  
Accepted 2 September 2022

## KEYWORDS

Orthogonal matrix; large-scale matrix; sequential updating; high-dimensional data


## MSC2020 AMS SUBJECT CLASSIFICATIONS

62L12; 15B10; 15A23; 68U10; 15A24

## 1. Introduction

In practice, one may be interested in finding the matrix  $X$  such that  $AX = B$  where matrices  $A$  and  $B$  come from experiments. However,  $A$  and  $B$  often do not satisfy the solvability conditions and hence, the least squares solution of the difference  $AX - B$  is required [15]. Specifically, the problem of approximating one given matrix  $A$  with another given matrix  $B$  by a transformation matrix  $X$  so that the square of the difference  $AX - B$  is minimized is known as the Procrustes problem [13,15,33]. Often, depending on the application, it is assumed that  $X$  belongs to a specific class of matrices, and thus setting in this way a set of constraints to the optimization problem. The most frequent classes of matrices for  $X$  is orthogonality and symmetry, and variants thereof, see, for example [5,11,15,19,21,29,31,33]. In many cases, orthogonal factorizations like the singular value decomposition (SVD), the eigenvalue decomposition (EVD) and the CS decomposition have been used to solve the Procrustes problem and variants thereof [8,15,19,22,31,33, pp. 327–328].

The application of the Procrustes problem in factor analysis has a long history [10,13,15,28]. It also appears in numerical analysis problems for the solution of partial differential equations, in multidimensional scaling, in growth curve modelling, in scientific computing, in computer vision, in image processing, in system and control theory, in the analysis of space structures and in aerospace engineering for spacecraft attitude determination [12,23,24,26,27,32,38,41,43]. Moreover, in the case where  $A$  is the identical matrix the problem becomes a matrix nearness problem with many applications in statistical and financial modelling and in theoretical computer science, see, for example [20,36,39].

**CONTACT** Stella Hadjiantoni  stella.hadjiantoni@essex.ac.uk  Department of Mathematical Sciences, University of Essex, Wivenhoe Park, Colchester, Essex CO4 3SQ, UK

© 2022 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives License (<http://creativecommons.org/licenses/by-nc-nd/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited, and is not altered, transformed, or built upon in any way.

The recursive solution to a least squares problem is needed when the experiment is conducted repeatedly and as a result the given matrices are updated with new arriving data regularly. Also, in high dimensional settings, the matrices  $\mathbf{A}$  and  $\mathbf{B}$  are very large and it may not be possible to treat all data at once or the computational cost of processing them may be significantly expensive. In this case, a sequential procedure which splits  $\mathbf{A}$  and  $\mathbf{B}$  into sub-matrices of smaller dimensions and then proceeds by gradually incorporating the sub-matrices into the least squares solution of  $\mathbf{AX} = \mathbf{B}$  is essential. Recursive least squares is often needed in many problems of different areas like engineering, statistics, econometrics and finance [7,9,17,18,42]. A recursive algorithm reduces the computational cost and also the storage requirements for large matrices.

Herein, the recursive least squares solution to the matrix equation  $\mathbf{AX} = \mathbf{B}$  when  $\mathbf{A}$  and  $\mathbf{B}$  are known matrices is investigated in depth. Namely, the recursive solution to the Procrustes problem is examined. The use of the QR decomposition is examined when there are no constraints on  $\mathbf{X}$ . Also, the problems of minimizing the difference  $\mathbf{AX} - \mathbf{B}$  when  $\mathbf{X}$  is orthogonal and when  $\mathbf{X}$  is symmetric are also considered and their recursive least squares solution using the eigenvalue and singular value decompositions is explored in depth. When constraints are imposed on  $\mathbf{X}$ , the method of Lagrange multipliers is used to solve the optimization problem. The proposed solution, in each case, is the matrix which minimizes the square of the Frobenius norm of  $\mathbf{AX} - \mathbf{B}$ . It is an exact solution in the sense that  $\mathbf{X}$  is explicitly determined and does not comprise any arbitrary elements. The recursive numerical solution proposed does not require the matrices be full rank.

Throughout this paper,  $\|\cdot\|_F$  denotes the Frobenius norm. Also, for known matrices  $\mathbf{S}$  and  $\mathbf{P}$ , when computing partial derivatives [25, p. 201] the following properties are used:

$$\frac{\partial(\mathbf{SXP})}{\partial\mathbf{X}} = \mathbf{S}^T\mathbf{P}^T, \quad \frac{\partial(\mathbf{SX}^T\mathbf{P})}{\partial\mathbf{X}} = \mathbf{PS}. \quad (1)$$

The paper is organized as follows. Section 2 introduces the general Procrustes problem where no assumption is made for the solution matrix  $\mathbf{X}$ . The problem is solved using the QR decomposition and then the recursive solution is presented. Section 3 considers the orthogonal Procrustes problem where the solution matrix  $\mathbf{X}$  is orthogonal and Section 4 derives the solution to the symmetric Procrustes problem when  $\mathbf{X}$  is assumed to be symmetric. Section 5 presents computational results and finally, in Section 6 we conclude and discuss future work.

## 2. Numerical solution to the general procrustes problem

Consider the problem of finding a matrix  $\mathbf{X} \in \mathbb{R}^{n \times n}$  so that the known matrix  $\mathbf{B} \in \mathbb{R}^{m \times n}$  is approximated by matrix  $\mathbf{AX}$  where  $\mathbf{A} \in \mathbb{R}^{m \times n}$  is also known. That is, a solution to the matrix equation

$$\mathbf{AX} = \mathbf{B} \quad (2)$$

is required. The least squares approximation problem to be solved is given by

$$\operatorname{argmin}_{\mathbf{X}} \|\mathbf{AX} - \mathbf{B}\|_F^2, \quad (3)$$

where

$$\begin{aligned} f(\mathbf{X}) &= \|\mathbf{AX} - \mathbf{B}\|_F^2 = \operatorname{trace} \left( (\mathbf{AX} - \mathbf{B})^T (\mathbf{AX} - \mathbf{B}) \right) \\ &= \operatorname{trace} \left( \mathbf{X}^T \mathbf{A}^T \mathbf{A} \mathbf{X} - \mathbf{X}^T \mathbf{A}^T \mathbf{B} - \mathbf{B}^T \mathbf{A} \mathbf{X} + \mathbf{B}^T \mathbf{B} \right). \end{aligned}$$

On using (1) partial differentiation yields

$$\frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} = 2\mathbf{A}^T \mathbf{A} \mathbf{X} - 2\mathbf{A}^T \mathbf{B} = \mathbf{0} \quad (4)$$

whence

$$A^T A X = A^T B. \quad (5)$$

When  $A^T A$  is non-singular the solution to (5) is given by  $X = (A^T A)^{-1} A^T B$  [14]. However,  $A^T A$  may be ill-conditioned and in this case inverting the matrix may give inaccurate results. In the case where  $A^T A$  is singular the latter will fail to give a solution for  $X$ . A numerically stable method to obtain  $X$  is to use the QR decomposition (QRD) of  $A$ , namely

$$Q_A^T (A \ B) = \begin{pmatrix} R_A & R_{B1} \\ \mathbf{0} & R_{B2} \end{pmatrix}, \quad \text{where } Q_A = (Q_{A1} \ Q_{A2}). \quad (6)$$

Then  $A = Q_{A1} R_A$  and hence  $A^T A = R_A^T R_A$  and  $A^T B = R_A^T R_{B1}$ . Therefore,  $X = R_A^{-1} R_{B1}$ . When  $A$  is rank deficient, the procedure to obtain  $X$  is similar to the above, namely (6). In this case, a complete QRD can be computed to triangularize  $A$  [1].

### 2.1. Recursive solution

We next consider the recursive solution of the Procrustes problem when the matrices  $A$  and  $B$  are updated. Without loss of generality, suppose that  $A$  and  $B$  are augmented with the addition of a *single row*; namely,

$$\tilde{A} = \begin{pmatrix} A \\ \mathbf{a} \end{pmatrix} \quad \tilde{B} = \begin{pmatrix} B \\ \mathbf{b} \end{pmatrix}, \quad (7)$$

where  $A, B$  are as in (2) and  $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{1 \times n}$  represent new data points. Then, the updated Procrustes problem, based on  $\tilde{A}$  and  $\tilde{B}$ , requires the solution of the least squares problem

$$\operatorname{argmin}_{\tilde{X}} \left\| \tilde{A} \tilde{X} - \tilde{B} \right\|_F^2 \quad (8)$$

when (3) has already been solved (see (4)–(6)). The efficient solution of (8) requires that previous computations from the solution of (3) be utilized. Namely,

$$\begin{aligned} \operatorname{argmin}_{\tilde{X}} \left\| \tilde{A} \tilde{X} - \tilde{B} \right\|_F^2 &= \operatorname{argmin}_{\tilde{X}} \left\| \begin{pmatrix} Q_{A1}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{1}_n \end{pmatrix} \left[ \begin{pmatrix} A \\ \mathbf{a} \end{pmatrix} \tilde{X} - \begin{pmatrix} B \\ \mathbf{b} \end{pmatrix} \right] \right\|_F^2 \\ &= \operatorname{argmin}_{\tilde{X}} \left\| \begin{pmatrix} R_A \\ \mathbf{a} \end{pmatrix} \tilde{X} - \begin{pmatrix} R_{B1} \\ \mathbf{b} \end{pmatrix} \right\|_F^2, \end{aligned}$$

where  $Q_{A1}$ ,  $R_A$  and  $R_{B1}$  are as in (6) and  $\mathbf{1}_n$  is the  $n$ -dimensional row vector of ones. Consider now the updating QRD

$$Q_{A_u}^T \begin{pmatrix} R_A & R_{B1} \\ \mathbf{a} & \mathbf{b} \end{pmatrix} = \begin{pmatrix} \tilde{R}_A & \tilde{R}_{B1} \\ \mathbf{0} & \tilde{R}_{B2} \end{pmatrix}. \quad (9)$$

We then have that

$$\operatorname{argmin}_{\tilde{X}} \left\| \tilde{A} \tilde{X} - \tilde{B} \right\|_F^2 = \operatorname{argmin}_{\tilde{X}} \left( \left\| \tilde{R}_A \tilde{X} - \tilde{R}_{B1} \right\|_F^2 + \left\| \tilde{R}_{B2} \right\|_F^2 \right)$$

and therefore it follows that  $\tilde{X} = \tilde{R}_A^{-1} \tilde{R}_{B1}$ .

In many cases, it is possible that the matrices  $\mathbf{A}$  and  $\mathbf{B}$  are updated with a *single column*, namely

$$\check{\mathbf{A}} = (\mathbf{A} \quad \mathbf{A}_{n+1}), \quad \check{\mathbf{B}} = (\mathbf{B} \quad \mathbf{B}_{n+1})$$

where  $\mathbf{A}_{n+1}, \mathbf{B}_{n+1} \in \mathbb{R}^{m \times 1}$  denote new variables which become available after (3) has been solved. As a result, the solution  $\check{\mathbf{X}} \in \mathbb{R}^{(n+1) \times (n+1)}$  to the updated Procrustes problem

$$\operatorname{argmin}_{\check{\mathbf{X}}} \left\| \check{\mathbf{A}}\check{\mathbf{X}} - \check{\mathbf{B}} \right\|_F^2. \quad (10)$$

needs to be computed. By utilizing efficiently previous computations from the solution of (3), (10) is written as

$$\begin{aligned} \operatorname{argmin}_{\check{\mathbf{X}}} \left\| \check{\mathbf{A}}\check{\mathbf{X}} - \check{\mathbf{B}} \right\|_F^2 &= \operatorname{argmin}_{\check{\mathbf{X}}} \left\| \mathbf{Q}_A^T \left( (\mathbf{A} \quad \mathbf{A}_{n+1}) \check{\mathbf{X}} - (\mathbf{B} \quad \mathbf{B}_{n+1}) \right) \right\|_F^2 \\ &= \operatorname{argmin}_{\check{\mathbf{X}}} \left\| \begin{pmatrix} \mathbf{R}_A & \tilde{\mathbf{A}}_{n+1} \\ 0 & \hat{\mathbf{A}}_{n+1} \end{pmatrix} \check{\mathbf{X}} - \begin{pmatrix} \mathbf{R}_{B_1} & \tilde{\mathbf{B}}_{n+1} \\ \mathbf{R}_{B_2} & \hat{\mathbf{B}}_{n+1} \end{pmatrix} \right\|_F^2, \end{aligned}$$

where  $\mathbf{Q}_A$  is from the QRD of  $\mathbf{A}$  in (6). The column-updating QRD that needs to be computed is then given by

$$\begin{pmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & \check{\mathbf{q}}^T \end{pmatrix} \begin{pmatrix} \mathbf{R}_A & \tilde{\mathbf{A}}_{n+1} \\ 0 & \hat{\mathbf{A}}_{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_A & \tilde{\mathbf{A}}_{n+1} \\ 0 & \check{a} \\ 0 & 0 \end{pmatrix},$$

where  $\check{\mathbf{q}}$  is an orthogonal transformation that eliminates all but the first element of  $\hat{\mathbf{A}}_{n+1}$  and  $\check{a}$  is a scalar. Hence, the updated Procrustes problem (10) becomes

$$\operatorname{argmin}_{\check{\mathbf{X}}} \left\| \check{\mathbf{A}}\check{\mathbf{X}} - \check{\mathbf{B}} \right\|_F^2 = \operatorname{argmin}_{\check{\mathbf{X}}} \left( \left\| \check{\mathbf{R}}_A \check{\mathbf{X}} - \check{\mathbf{R}}_B \right\|_F^2 + \left\| \check{\mathbf{R}}_{B_2} \right\|_F^2 \right), \quad (11)$$

where

$$\check{\mathbf{R}}_A = \begin{pmatrix} \mathbf{R}_A & \tilde{\mathbf{A}}_{n+1} \\ 0 & \check{a} \end{pmatrix}, \quad \check{\mathbf{R}}_B = \begin{pmatrix} \mathbf{R}_{B_1} & \tilde{\mathbf{B}}_{n+1} \\ \mathbf{R}_{B_2}^{(1)} & \hat{\mathbf{B}}_{n+1}^{(1)} \end{pmatrix} \quad \text{and} \quad \check{\mathbf{R}}_{B_2} = \begin{pmatrix} \mathbf{R}_{B_2}^{(2)} & \hat{\mathbf{B}}_{n+1}^{(2)} \end{pmatrix}.$$

The solution to problem (11) is given by  $\check{\mathbf{X}} = \check{\mathbf{R}}_A^{-1} \check{\mathbf{R}}_B$ .

### 3. The orthogonal procrustes problem

The orthogonal Procrustes problem (OPP) is that of minimizing the sum of the squared error of the difference matrix  $\mathbf{A}\mathbf{X} - \mathbf{B}$  when the unknown matrix  $\mathbf{X}$  is orthogonal. The constraint is imposed by using the method of Lagrange multipliers; the matrices  $\mathbf{A}$  and  $\mathbf{B}$  need not be full rank [33]. The constrained optimization problem is then given by

$$\operatorname{argmin}_{\mathbf{X}} \|\mathbf{A}\mathbf{X} - \mathbf{B}\|_F^2 \quad \text{subject to} \quad \mathbf{X}\mathbf{X}^T = \mathbf{X}^T\mathbf{X} = \mathbf{I}, \quad (12)$$

where  $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$  and  $\mathbf{X} \in \mathbb{R}^{n \times n}$  is orthogonal. Herein, we are most interested in cases where  $m < n$ . To find the solution to (12), we consider the Lagrangian function

$$\mathcal{L}(\mathbf{X}) = \operatorname{trace} \left( (\mathbf{A}\mathbf{X} - \mathbf{B})^T (\mathbf{A}\mathbf{X} - \mathbf{B}) \right) + \operatorname{trace} \left( \boldsymbol{\Lambda} (\mathbf{X}\mathbf{X}^T - \mathbf{I}) \right),$$

which is equivalently written as

$$\mathcal{L}(X) = \sum_{j=1}^n \sum_{i=1}^m \left( \sum_{q=1}^n a_{iq}x_{qj} - b_{ij} \right)^2 + \sum_{q,r=1}^n \lambda_{qr} \left( \sum_{k=1}^n x_{qs}x_{rs} - \delta_{qr} \right), \quad (13)$$

where  $\Lambda = [\lambda_{qr}]_{q,r=1}^n$  is the symmetric matrix of Lagrange multipliers [15]. Partial differentiation of (13) yields

$$\frac{\partial \mathcal{L}(X)}{\partial x_{pj}} = 2 \sum_{i=1}^m \left( \sum_{q=1}^n a_{iq}x_{qj} - b_{ij} \right) a_{ip} + 2 \sum_{r=1}^n \lambda_{qr}x_{rj}. \quad (14)$$

On setting (14) equal to zero and using matrix notation, (14) can be written as

$$A_q^T A_q X_j + \Lambda_q X_j = A_q^T B_j, \quad q, j = 1, \dots, n,$$

or equivalently, as

$$(A^T A + \Lambda)X = A^T B, \quad (15)$$

where  $A_q$  is the  $q$ th column of  $A$ ,  $X_j$ ,  $B_j$  are the  $j$ th columns of  $X$  and  $B$ , respectively, and  $\Lambda_q$  is the  $q$ th row of  $\Lambda$ . From (15),  $(A^T A + \Lambda)XX^T(A^T A + \Lambda)^T = A^T B(A^T B)^T$ , therefore  $\Lambda = (A^T B B^T A)^{1/2} - A^T A$  and, as observed earlier,  $\Lambda$  is symmetric. Now on post-multiplying (15) by  $X^T$  gives  $A^T A + \Lambda = A^T B X^T$  which implies that  $\Lambda = A^T B X^T - A^T A$ , whence  $A^T B X^T = X B^T A$ . Therefore,

$$A^T B = X B^T A. \quad (16)$$

Furthermore, let  $H = A^T B$  and consider the following two matrices

$$F = H H^T = A^T B B^T A$$

and

$$G = H^T H = B^T A A^T B.$$

Matrices  $F$ ,  $G \in \mathbb{R}^{n \times n}$  are symmetric and thus they are diagonalizable and their eigenvalue decomposition (EVD) exists, that is,

$$F = U D U^T \quad (17a)$$

and

$$G = V D V^T, \quad (17b)$$

where  $U$ ,  $V \in \mathbb{R}^{n \times n}$  are orthogonal. Additionally, since they are of the form  $H H^T$  and  $H^T H$  they have the same eigenvalues. Now on using (16) it follows that

$$\begin{aligned} F &= A^T B B^T A \\ &= (X B^T A X) (X B^T A X)^T \\ &= X B^T A A^T B X^T \\ &= X G X^T, \end{aligned}$$

and from (17a) and (17b) we now have that

$$F = U D U^T = X V D V^T X^T,$$

which implies that  $U = XV$ , where  $U$  is as in (17a). Therefore, the solution to the least squares problem (12) is given by

$$X = UV^T.$$

Finally, a sufficient condition which guaranties the uniqueness of the solution  $X$  and that the argument in (12) is minimized requires that all the diagonal elements of the matrix  $D^{1/2}$  in (17a) and (17b) are non-negative [15,33]. In essence, this condition determines the orientation of the orthogonal matrices  $U$  and  $V$  and is specified by the Eckart–Young decomposition (see [10]) of  $A^T B$ , namely

$$A^T B = UD^{1/2}V^T.$$

Notice that in the case of symmetric orthogonality the procedure is the same but at the last step the symmetry of  $X$  needs to be taken into account. Namely, the sum of  $UV^T$  is not explicitly computed since only the upper or lower part of  $X$  needs to be determined. This results in a dimensional reduction of the solution matrix from  $n^2$  to  $n(n+1)/2$  [40].

### 3.1. Recursive solution to the orthogonal procrustes problem

Suppose that an updated orthogonal Procrustes problem needs to be solved when new data become available. Without loss of generality it is assumed that the original OPP (12) needs to be re-solved when appending a new *single row* of data in matrices  $A$  and  $B$ . That is, let the row updated matrices  $\tilde{A}$  and  $\tilde{B}$  be as in (7), where  $A, B$  are as in (12) with  $a, b \in \mathbb{R}^{1 \times n}$ . The updated orthogonal Procrustes problem based on  $\tilde{A}$  and  $\tilde{B}$  requires that the solution of the following least squares problem be derived:

$$\underset{\tilde{X}}{\operatorname{argmin}} \left\| \tilde{A}\tilde{X} - \tilde{B} \right\|_F^2 \quad \text{subject to } \tilde{X}\tilde{X}^T = \tilde{X}^T\tilde{X} = I, \quad (18)$$

where  $\tilde{X} \in \mathbb{R}^{n \times n}$ . The solution to the least squares problem (18) is obtained by using the method of Lagrange multipliers, namely

$$\mathcal{L}(\tilde{X}) = \operatorname{trace} \left( (\tilde{A}\tilde{X} - \tilde{B})^T (\tilde{A}\tilde{X} - \tilde{B}) \right) + \operatorname{trace} \left( \tilde{\Lambda} (\tilde{X}\tilde{X}^T - I) \right) \quad (19)$$

which yields the first order condition

$$\frac{\partial \mathcal{L}(\tilde{X})}{\partial \tilde{X}} = 2\tilde{A}^T \tilde{A}\tilde{X} - 2\tilde{A}^T \tilde{B} + (\tilde{\Lambda}^T + \tilde{\Lambda})\tilde{X}$$

on using (1). In a way similar to (12), (19) has the solution

$$\tilde{X} = \tilde{U}\tilde{V}^T, \quad (20)$$

where  $\tilde{U}$  and  $\tilde{V}$  are the orthogonal matrices of the EVD of  $\tilde{A}^T \tilde{B}\tilde{B}^T \tilde{A}$  and  $\tilde{B}^T \tilde{A}\tilde{A}^T \tilde{B}$ , respectively.

The recursive solution of the OPP presumes that previous computations in (17a) from the solution of the original Procrustes problem (12) are efficiently utilized. Consider the recursive computation of the EVD of  $\tilde{A}^T \tilde{B}\tilde{B}^T \tilde{A}$ , namely

$$\begin{aligned} \tilde{A}^T \tilde{B}\tilde{B}^T \tilde{A} &= (A^T B + a^T b)(A^T B + a^T b)^T \\ &= A^T B B^T A + A^T B b^T a + a^T b B^T A + a^T b b^T a \\ &= UDU^T + A^T B b^T a + a^T b B^T A + a^T b b^T a \end{aligned} \quad (21)$$

on using (17a). Therefore, the recursive solution of an orthogonal Procrustes problem becomes a modified symmetric matrix eigenvalue problem [3,4,16]. In particular, (21) implies three rank-1 modifications of the EVD  $UDU^T$  of the matrix  $A^T B B^T A$  in (17a). That is, the EVD of  $\tilde{A}^T \tilde{B}\tilde{B}^T \tilde{A}$  is obtained

recursively in three main steps. First, (21) is written as

$$\tilde{\mathbf{A}}^T \tilde{\mathbf{B}} \tilde{\mathbf{B}}^T \tilde{\mathbf{A}} = \mathbf{U} \mathbf{D} \mathbf{U}^T + \begin{pmatrix} \tilde{\mathbf{b}} \\ \mathbf{a} \end{pmatrix}^T \begin{pmatrix} 0 & 1 \\ 1 & \beta \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{b}} \\ \mathbf{a} \end{pmatrix}, \quad (22)$$

where  $\tilde{\mathbf{b}} = \mathbf{b} \mathbf{B}^T \mathbf{A}$  and  $\beta = \mathbf{b} \mathbf{b}^T$  is a scalar since  $\mathbf{b}$  and  $\mathbf{a}$  are row vectors. Second, the following EVD is derived

$$\mathbf{\Delta} = \begin{pmatrix} 0 & 1 \\ 1 & \beta \end{pmatrix} = \mathbf{Q} \begin{pmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{pmatrix} \mathbf{Q}^T \quad (23)$$

and, since  $\mathbf{\Delta}$  is a symmetric matrix, it is diagonalizable. Third, on using (22), (23) becomes

$$\begin{aligned} \tilde{\mathbf{A}}^T \tilde{\mathbf{B}} \tilde{\mathbf{B}}^T \tilde{\mathbf{A}} &= \mathbf{U} \mathbf{D} \mathbf{U}^T + \begin{pmatrix} \tilde{\mathbf{b}} \\ \mathbf{a} \end{pmatrix}^T \mathbf{Q} \begin{pmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{pmatrix} \mathbf{Q}^T \begin{pmatrix} \tilde{\mathbf{b}} \\ \mathbf{a} \end{pmatrix} \\ &= \mathbf{U} \mathbf{D} \mathbf{U}^T + \begin{pmatrix} \tilde{\mathbf{b}}_1 \\ \tilde{\mathbf{b}}_2 \end{pmatrix}^T \begin{pmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{b}}_1 \\ \tilde{\mathbf{b}}_2 \end{pmatrix} \\ &= \mathbf{U} \mathbf{D} \mathbf{U}^T + \beta_1 \tilde{\mathbf{b}}_1^T \tilde{\mathbf{b}}_1 + \beta_2 \tilde{\mathbf{b}}_2^T \tilde{\mathbf{b}}_2 \\ &= \mathbf{U} \left( \mathbf{D} + \beta_1 \tilde{\mathbf{b}}_1^T \tilde{\mathbf{b}}_1 \right) \mathbf{U}^T + \beta_2 \tilde{\mathbf{b}}_2^T \tilde{\mathbf{b}}_2, \end{aligned} \quad (24)$$

where  $\tilde{\mathbf{b}}_1 = \tilde{\mathbf{b}}_1 \mathbf{U}$  with  $\beta_1, \beta_2$  being the eigenvalues of  $\mathbf{\Delta}$ . Consider now the sequential updating of the diagonal matrix  $\mathbf{D}$  in two steps. The first step computes the EVD of  $\mathbf{D} + \beta_1 \tilde{\mathbf{b}}_1^T \tilde{\mathbf{b}}_1 = \mathbf{U}_1 \mathbf{D}_1 \mathbf{U}_1^T$ , where  $\mathbf{U}_1 \in \mathbb{R}^{n \times n}$  is orthogonal and  $\mathbf{D}_1 \in \mathbb{R}^{n \times n}$  is diagonal. The second step computes the EVD of  $\mathbf{D}_1 + \beta_2 \tilde{\mathbf{b}}_2^T \tilde{\mathbf{b}}_2 = \mathbf{U}_2 \tilde{\mathbf{D}} \mathbf{U}_2^T$ , where  $\mathbf{U}_2 \in \mathbb{R}^{n \times n}$  is orthogonal and  $\tilde{\mathbf{D}} \in \mathbb{R}^{n \times n}$  is the diagonal matrix with elements the eigenvalues of  $\tilde{\mathbf{A}}^T \tilde{\mathbf{B}} \tilde{\mathbf{B}}^T \tilde{\mathbf{A}}$ . That is,  $\tilde{\mathbf{A}}^T \tilde{\mathbf{B}} \tilde{\mathbf{B}}^T \tilde{\mathbf{A}} = \tilde{\mathbf{U}} \tilde{\mathbf{D}} \tilde{\mathbf{U}}^T$ , where  $\tilde{\mathbf{U}} = \mathbf{U} \mathbf{U}_1 \mathbf{U}_2$  and  $\mathbf{U}$  is as in (17b). Repeating the procedure at steps (21)–(24) for  $\tilde{\mathbf{B}}^T \tilde{\mathbf{A}} \tilde{\mathbf{A}}^T \tilde{\mathbf{B}}$  will derive its EVD recursively and will therefore give  $\tilde{\mathbf{B}}^T \tilde{\mathbf{A}} \tilde{\mathbf{A}}^T \tilde{\mathbf{B}} = \tilde{\mathbf{V}} \tilde{\mathbf{D}} \tilde{\mathbf{V}}^T$ , where  $\tilde{\mathbf{V}} = \mathbf{V} \mathbf{V}_1 \mathbf{V}_2$ ,  $\mathbf{V}$  is defined in (17a) and  $\mathbf{V}_1, \mathbf{V}_2$  are computed in a way similar to that for (24). Therefore, the updated solution (20) to the Procrustes problem (18) has been derived.

When extra columns are added to  $\mathbf{A}$  and  $\mathbf{B}$ , that is when

$$\check{\mathbf{A}} = \begin{pmatrix} n & k \\ \mathbf{A} & \mathbf{A}_{n+1} \end{pmatrix} \text{ and } \check{\mathbf{B}} = \begin{pmatrix} n & k \\ \mathbf{B} & \mathbf{B}_{n+1} \end{pmatrix}, \quad (25)$$

an updated orthogonal Procrustes problem of *larger* dimensions needs to be solved. Namely,

$$\operatorname{argmin}_{\check{\mathbf{X}}} \left\| \check{\mathbf{A}} \check{\mathbf{X}} - \check{\mathbf{B}} \right\|_F^2 \quad \text{subject to } \check{\mathbf{X}} \check{\mathbf{X}}^T = \check{\mathbf{X}}^T \check{\mathbf{X}} = \mathbf{I}, \quad (26)$$

where  $\check{\mathbf{X}} \in \mathbb{R}^{(n+k) \times (n+k)}$  has been augmented by  $k$  columns and  $k$  rows. In a similar way as for (18), the solution of (26) is given by  $\check{\mathbf{X}} = \check{\mathbf{U}} \check{\mathbf{V}}^T$ . It is obtained recursively, by updating the original orthogonal decompositions as in (24). Notice that the addition of extra columns implies a rank- $k$  updating of the EVD decomposition.



#### 4. The symmetric procrustes problem

Consider the problem of minimizing the sum of squared error of the difference matrix  $\mathbf{A}\mathbf{X}_S - \mathbf{B}$  when the unknown matrix  $\mathbf{X}_S$  is symmetric. The constrained optimization problem is given by

$$\operatorname{argmin}_{\mathbf{X}_S} \|\mathbf{A}\mathbf{X}_S - \mathbf{B}\|_F^2 \quad \text{subject to } \mathbf{X}_S^T = \mathbf{X}_S,$$

where  $\mathbf{X}_S \in \mathbb{R}^{n \times n}$  is a symmetric matrix. As in the case of the OPP (12), the matrices  $\mathbf{A}$  and  $\mathbf{B}$  are not necessarily full rank. Using the method of Lagrange multipliers, the problem becomes that of finding the matrix  $\mathbf{X}_S$  which minimizes

$$\mathcal{L}(\mathbf{X}_S) = \operatorname{trace} \left( (\mathbf{A}\mathbf{X}_S - \mathbf{B})^T (\mathbf{A}\mathbf{X}_S - \mathbf{B}) \right) + \operatorname{trace} \left( \boldsymbol{\Lambda} (\mathbf{X}_S - \mathbf{X}_S^T) \right).$$

On using (1) partial differentiation yields

$$\frac{\partial \mathcal{L}(\mathbf{X}_S)}{\partial \mathbf{X}_S} = 2\mathbf{A}^T \mathbf{A} \mathbf{X}_S - 2\mathbf{A}^T \mathbf{B} + \boldsymbol{\Lambda}^T - \boldsymbol{\Lambda},$$

which is set to zero. Now let the matrix  $\mathbf{M} = \boldsymbol{\Lambda} - \boldsymbol{\Lambda}^T$ , which is skew-symmetric, that is  $\mathbf{M} = -\mathbf{M}^T$ . It follows that

$$2\mathbf{A}^T \mathbf{A} \mathbf{X}_S - 2\mathbf{A}^T \mathbf{B} = - \left( 2\mathbf{X}_S \mathbf{A}^T \mathbf{A} - 2\mathbf{B}^T \mathbf{A} \right),$$

which yields the Lyapunov equation

$$\mathbf{A}^T \mathbf{A} \mathbf{X}_S + \mathbf{X}_S \mathbf{A}^T \mathbf{A} = \mathbf{A}^T \mathbf{B} + \mathbf{B}^T \mathbf{A}. \quad (27)$$

Since  $\mathbf{A}^T \mathbf{A}$  is symmetric, it is therefore a diagonalizable matrix. That is, there is an orthogonal matrix  $\mathbf{P}$  and a diagonal matrix  $\mathbf{D}_A$  such that

$$\mathbf{A}^T \mathbf{A} = \mathbf{P} \mathbf{D}_A \mathbf{P}^T, \quad (28)$$

where  $\mathbf{P} \in \mathbb{R}^{n \times n}$  has columns the eigenvectors of  $\mathbf{A}^T \mathbf{A}$  and  $\mathbf{D}_A = \operatorname{diag}(\mu_1, \dots, \mu_n)$  has diagonal elements the eigenvalues of  $\mathbf{A}^T \mathbf{A}$ . Using (27), (28) becomes

$$\mathbf{D}_A \mathbf{X}_S^{(P)} + \mathbf{X}_S^{(P)} \mathbf{D}_A = \mathbf{S}, \quad (29)$$

where  $\mathbf{X}_S^{(P)} = \mathbf{P}^T \mathbf{X}_S \mathbf{P}$  and  $\mathbf{S} = \mathbf{P}^T (\mathbf{A}^T \mathbf{B} + \mathbf{B}^T \mathbf{A}) \mathbf{P}$ . By utilizing the diagonal structure of  $\mathbf{D}_A$ , it follows that

$$x_{i,j} = \frac{s_{ij}}{\mu_i + \mu_j},$$

where  $\mathbf{X}_S^{(P)} = [x_{i,j}]_{i,j=1}^n$ . The solution is then given by

$$\mathbf{X}_S = \mathbf{P} \mathbf{X}_S^{(P)} \mathbf{P}^T.$$

A necessary and sufficient condition for the uniqueness of  $\mathbf{X}_S$ , since  $\mathbf{S}$  is positive definite, is that all the eigenvalues of  $\mathbf{A}^T \mathbf{A}$  have a negative real part, that is,  $\mathbf{A}^T \mathbf{A}$  is a stable matrix [2,34,35].

#### 4.1. Recursive solution to the symmetric procrustes problem

Consider now the case where the matrices  $\mathbf{A}$  and  $\mathbf{B}$  are augmented by the addition of an extra row as in (7). The updated symmetric Procrustes problem requires the solution of the optimization problem

$$\operatorname{argmin}_{\tilde{\mathbf{X}}_S} \left\| \tilde{\mathbf{A}}\tilde{\mathbf{X}}_S - \tilde{\mathbf{B}} \right\|_F^2 \quad \text{subject to } \tilde{\mathbf{X}}_S = \tilde{\mathbf{X}}_S^T,$$

where  $\tilde{\mathbf{A}}$  and  $\tilde{\mathbf{B}}$  are defined as in (7). Using the Lagrange multipliers and the analysis for the symmetric Procrustes problem as in the previous section, the solution is obtained from the Lyapunov equation

$$\tilde{\mathbf{A}}^T \tilde{\mathbf{A}} \tilde{\mathbf{X}}_S + \tilde{\mathbf{X}}_S \tilde{\mathbf{A}}^T \tilde{\mathbf{A}} = \tilde{\mathbf{A}}^T \tilde{\mathbf{B}} + \tilde{\mathbf{B}}^T \tilde{\mathbf{A}} \quad (30)$$

by computing the EVD of  $\tilde{\mathbf{A}}^T \tilde{\mathbf{A}}$ . To recursively solve (30) observe that on using (28) yields

$$\begin{aligned} \tilde{\mathbf{A}}^T \tilde{\mathbf{A}} &= \mathbf{A}^T \mathbf{A} + \mathbf{a}^T \mathbf{a} \\ &= \mathbf{P} \mathbf{D}_A \mathbf{P}^T + \mathbf{a}^T \mathbf{a} \\ &= \mathbf{P} (\mathbf{D}_A + \tilde{\mathbf{a}}^T \tilde{\mathbf{a}}) \mathbf{P}^T. \end{aligned}$$

Therefore, the sequential updating of  $\mathbf{D}_A$  requires one rank-1 update, namely,  $\mathbf{D}_A + \tilde{\mathbf{a}}^T \tilde{\mathbf{a}} = \mathbf{P}_1 \tilde{\mathbf{D}}_A \mathbf{P}_1^T$ , whence

$$\tilde{\mathbf{A}}^T \tilde{\mathbf{A}} = \tilde{\mathbf{P}} \tilde{\mathbf{D}}_A \tilde{\mathbf{P}}^T$$

with  $\tilde{\mathbf{P}} = \mathbf{P} \mathbf{P}_1$ . The Lyapunov equation (30) then becomes

$$\tilde{\mathbf{D}}_A \tilde{\mathbf{X}}_S^P + \tilde{\mathbf{X}}_S^P \tilde{\mathbf{D}}_A = \tilde{\mathbf{S}}, \quad (31)$$

where  $\tilde{\mathbf{X}}_S^{(P)} = \tilde{\mathbf{P}}^T \tilde{\mathbf{X}}_S \tilde{\mathbf{P}}$  and  $\tilde{\mathbf{S}} = \tilde{\mathbf{P}}^T (\tilde{\mathbf{A}}^T \tilde{\mathbf{B}} + \tilde{\mathbf{B}}^T \tilde{\mathbf{A}}) \tilde{\mathbf{P}}$ . The solution to (31) is obtained in a way similar to that for (29).

#### 5. Computational remarks

To amplify the practical usability of the proposed method, the theoretical complexity analysis of the new algorithms has been studied. Consider the general Procrustes problem when no constraint is imposed to the least squares solution matrix  $\mathbf{X}$ . Let the matrices  $\tilde{\mathbf{A}}, \tilde{\mathbf{B}} \in \mathbb{R}^{(m+1) \times n}$  as in (7) and assume that the QRD of  $\mathbf{A}$  in (6) is available. The complexity of computing the QRD of  $\tilde{\mathbf{A}}$  afresh is  $2n^2(m + 1 - n/3)$  floating point operations (flops) and that of applying these orthogonal transformations to matrix  $\tilde{\mathbf{B}}$  is  $2n^2(2m - n + 3)$  flops. The complexities of computing the updating QRD in (9) in order to update  $\mathbf{R}_A$  and  $\mathbf{R}_B$  require  $4n^2$  and  $8n^2$  flops, respectively. As a result, to compute the QRD of  $\tilde{\mathbf{A}}$  and to apply the orthogonal transformations to  $\tilde{\mathbf{B}}$  will always be computationally more demanding than computing it recursively utilizing previous calculations. The computational efficiency of the recursive method compared to computing the QRD from scratch is, approximately, given by  $(9m + 2n)/18$ .

The solution of the orthogonal Procrustes problem is derived from the EVD of the square matrices  $\mathbf{F} = \mathbf{A}^T \mathbf{B} (\mathbf{A}^T \mathbf{B})^T$  and  $\mathbf{G} = \mathbf{B}^T \mathbf{A} \mathbf{A}^T \mathbf{B}$ . However, in practice the SVD of  $\mathbf{A}^T \mathbf{B} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$  is computed since  $\mathbf{F} = \mathbf{U} \mathbf{\Sigma}^2 \mathbf{U}^T = \mathbf{U} \mathbf{D} \mathbf{U}^T$  and  $\mathbf{G} = \mathbf{V} \mathbf{\Sigma}^2 \mathbf{V}^T = \mathbf{V} \mathbf{D} \mathbf{V}^T$ . When  $\mathbf{A}$  and  $\mathbf{B}$  are modified, the computationally efficient solution to the orthogonal Procrustes problem requires that the SVD of  $\mathbf{A}^T \mathbf{B}$  is computed recursively. Therefore, in the recursive solution of the orthogonal and symmetric Procrustes problems, updating SVD decompositions are computed which provide equivalent results as if the EVD were computed. The SVD of an  $m \times n$  matrix requires  $n^3 + mn^2 + O(mn)$  flops in

the first stage and  $kmn^2 + kn^2 + O(mn)$  flops in the second stage where  $k$  is the number of iterations required to compute each singular value. Herein, it is assumed that  $m \ll n$  and it holds that  $\text{rank}(A^T B) = r < \min(m, n)$ .

The algorithms employ a low-rank modification strategy for the recursive updating of the SVD as in [3]. Algorithms 5.1–5.3 summarize the main computational steps for the solution of the orthogonal Procrustes problem and the recursive proposed solution when new rows or columns of data are added, respectively.

---

**Algorithm 5.1** Solving the OPP in (12).

---

**Require:** Known matrices  $A \in \mathbb{R}^{m \times n}$  and  $B \in \mathbb{R}^{m \times n}$ .

**Ensure:** The solution  $X$  and  $U, V, \Sigma$  from the SVD of  $A^T B$ .

1. Compute the SVD of  $A^T B = U \Sigma V^T$ .
  2. Compute  $X = UV^T$ .
- 

---

**Algorithm 5.2** Solving the OPP in (26) recursively with column updating, given the solution of (12).

---

**Require:** The new data added are  $A_{n+1}, B_{n+1} \in \mathbb{R}^{m \times k}$ ,  $k \geq 1$  as in (25) and  $U, V, \Sigma$  from the solution of the OPP (12).

**Ensure:** The solution  $\tilde{X}$ . The  $\tilde{U}, \tilde{V}, \tilde{\Sigma}$ .

1. Let  $\tilde{U}_1 = (V_1 \& \mathbf{0}_{k \times n})$ .
  2. Compute  $M = U^T A_{n+1}$ .
  3. Compute the SVD  $K = U_K \Sigma_K V_K^T$ , where  $K = (\Sigma \& M)$ .
  4. Compute  $U_1 = UU_K$  and  $V_1 = VV_K$ . Let  $\tilde{U}_1 = \begin{pmatrix} U_1 \\ \mathbf{0}_{k \times n} \end{pmatrix}$ .
  5. Compute the QRD  $Q^T (V \& B_{n+1}) = \begin{pmatrix} I_n \& N \\ \mathbf{0} \& R_B \end{pmatrix}$  and the SVD  $\begin{pmatrix} \Sigma_K \& N \\ \mathbf{0} \& R_B \end{pmatrix} = U_L \Sigma_L V_L^T$ .
  6. Compute  $\tilde{X} = \tilde{U} \tilde{V}^T$ , where  $\tilde{U} = \tilde{U}_1 U_L$  and  $\tilde{V} = V_1 V_L$ .
- 

To evaluate further the new algorithms, experiments based on synthetic and real data have been conducted. The computational times of the new algorithms have been compared with the algorithm that solves the same problem afresh in order to obtain the efficiency ratio.

Table 1 presents the execution times (in CPU seconds) of Algorithm 5.1 compared with Algorithm 5.3 when  $m = 50, 100, 250, 500$  rows and  $n = 1000, 5000$  and a single column of data is added to  $A$  and  $B$ , that is  $k = 1$ . The results show that keeping  $n$  fixed and increasing  $m$  reduces the computational efficiency. However, comparing Panel A and B shows that the efficiency is more significant when  $n$  increases. Table 2 presents the execution times (in CPU seconds) of the Algorithms 5.1 and 5.2 when  $m = 100, n = 1000$  and when  $m = 500, n = 10,000$  with a variable number  $k$  of new columns are added to  $A$  and  $B$ . All the times presented are the average times after solving the same OPP (afresh or recursively) 100 times. The computational results in Table 2 show that as the number of dimensions increase the computational efficiency also increases. Furthermore, the results suggest that the proposed algorithm for the addition on new column is more efficient when a small number of extra columns is included. Comparing the results in Panel A and B of Table 2 we see that the efficiency of the proposed recursive method increases when the dimensions of the matrices in the original OPP also increase.

Many a time, the solution of an OPP is applied in the development of algorithms for face recognition, see for example [6,30,37,43]. The algorithm proposed in [43] for feature extraction requires to solve -until convergence- a series of updating OPPs as in (12) where  $A$  is the data matrix and  $B$

**Table 1.** Execution times in seconds of the recursive solution of the OPP when one single column is added.

$m$	Alg. 5.1	Alg. 5.2	$\frac{\text{Alg. 5.1}}{\text{Alg. 5.2}}$
	afresh	recursive	
(a) Panel A: Orthogonal Procrustes updating with $n = 1000$ and $k = 1$ .			
50	239	6	37
100	273	11	26
250	285	33	9
500	387	178	2
(b) Panel B: Orthogonal Procrustes updating with $n = 5000$ , $k = 1$ .			
50	25723	7	3701
100	25510	14	1800
250	25985	71	366
500	26973	307	88

Notes: Results are presented where the matrices in the original Procrustes have  $m = 50, 100, 250, 500$  rows and  $n = 1000$  or  $5000$  columns and  $k = 1$  column is added.

**Table 2.** Execution times in seconds of the recursive solution of the OPP when new columns are added.

$k$	Alg. 5.1	Alg. 5.2	$\frac{\text{Alg. 5.1}}{\text{Alg. 5.2}}$
	afresh	recursive	
(a) Panel A: Orthogonal Procrustes updating with $m = 100$ and $n = 1000$ .			
5	2.0	0.1	24
25	2.1	0.1	22
100	2.7	0.2	15
250	3.7	0.5	8
500	6.5	1.6	4
(b) Panel B: Orthogonal Procrustes updating with $m = 500$ and $n = 10,000$ .			
5	2216.9	11.3	196
25	2117.3	11.3	187
100	2069.8	14.1	147
250	3.7	0.5	8
500	6.5	1.6	4

Notes: Results are presented where the matrices in the original OPP have  $m = 100$  or  $500$  and  $n = 1000$  and  $k = 5, 25, 100, 250, 500$  new columns are added.

is the class indicator matrix after they have both been centred. The processing of an image starts by considering their pixels as the elements of a matrix and then converting each matrix (or each image) to a column vector. Each row in  $A$  has length equal to the feature number of the images. When a new image becomes available and is added to the database, a row will be added to  $A$ . The face recognition algorithm will then have to process the extra image, that is the additional row of the data matrix  $A$ . This is equivalent to row updating as in (18). Herein, Algorithm 5.3 is employed to show the computational efficiency of utilizing previous computations when a series of updating OPPs is solved after augmenting the data matrix.

In Table 3 the algorithm that estimates the problems afresh and the recursive algorithm are compared. It is assumed that there are  $m = 2000, 5000, 10,000, 15,000, 20,000$  images available which have been cropped and re-sized to  $16 \times 16$  and  $32 \times 32$  pixels, that is  $n = 256, 1024$ . The run times (in CPU seconds) when an extra image or alternatively an extra row of data is included 100 times are presented. The results show that when new rows of data are added to the problem the efficiency increases as the number of rows increase. The reason for this is the fact that the recursive algorithm does not depend on  $m$ , the number of rows of matrices  $A$  and  $B$ . Instead, it depends on the number of

---

**Algorithm 5.3** Solving the OPP in (18) recursively with row updating, given the solution of (12).

---

**Require:** The new data added are  $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{1 \times n}$  as in (7) and  $\mathbf{U}, \mathbf{V}, \mathbf{\Sigma}$  from the solution of the OPP (12).

**Ensure:** The solution  $\tilde{\mathbf{X}}$  and  $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{\Sigma}}$ .

1. **Repeat row updating**

**Require:** The new data added are  $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{1 \times n}$  as in (7) and  $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{\Sigma}}$  from the solution of (18).

**Ensure:** The solution  $\tilde{\mathbf{X}}$  and  $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{\Sigma}}$ .

2. Compute  $\mathbf{a}_u = \mathbf{U}^T \mathbf{a}$  and  $\mathbf{b}_v = \mathbf{V}^T \mathbf{b}$ .
  3. Compute the SVD  $\mathbf{K} = \mathbf{U}_K \mathbf{\Sigma}_K \mathbf{V}_K^T$ , where  $\mathbf{K} = \mathbf{\Sigma} + \mathbf{a}_u \mathbf{b}_v^T$ .
  4. Compute  $\tilde{\mathbf{X}} = \tilde{\mathbf{U}} \tilde{\mathbf{V}}^T$ , where  $\tilde{\mathbf{U}} = \mathbf{U} \mathbf{U}_K$  and  $\tilde{\mathbf{V}} = \mathbf{V} \mathbf{V}_K$ .
- 

**Table 3.** Total execution times in seconds of the recursive solution of the OPP when a new single row of data is added 100 times.

$m$	Alg. 5.1	Alg. 5.3	$\frac{\text{Alg. 5.1}}{\text{Alg. 5.3}}$
	afresh	recursive	
(a) Panel A: Orthogonal Procrustes updating with $n = 256$ and $l = 1$ .			
2000	10	6.1	2
5000	20.9	6.1	3
1000	39.3	6.1	6
15,000	57.1	6.1	9
20,000	74.9	6.1	12
(b) Panel B: Orthogonal Procrustes updating with $n = 1024$ and $l = 1$ .			
2000	369.8	371.9	1
5000	546.3	371.9	1
10,000	823.4	371.9	2
15,000	1090.7	371.9	3
20,000	1369.8	371.9	4

Notes: Results are presented when the matrices in the original OPP have dimensions  $m = 2000, 5000, 10,000, 15,000, 20,000$  and  $n = 256, 1024$ , and during the updating one extra row is added ( $l = 1$ ).

columns  $n$  of  $\mathbf{A}$  and  $\mathbf{B}$  and on the number of rows  $l$  added to the model. This is also why the timings of Algorithm 5.3 are the same when  $n$  and  $l$  are fixed.

Overall, the results show that the computational efficiency of the proposed recursive algorithm increases when the dimensions of the matrices (data) increase. The computational efficiency is significantly more important when a small number of rows or columns is amended -compared to the original matrices- that is, the modification is low-rank. This demonstrates the practicability of the proposed method when solving sequentially OPPs with small modifications in the underlying dataset. The proposed methods are particularly usable when the matrices involved are large-scale and the data are high-dimensional. All the reported computational results were performed on a 64-bit 1.80 GHz Core(TM) i7-8550U processor and 16.0 GB RAM using R (version 3.6.1).

## 6. Conclusions and future work

The recursive least squares solutions to the matrix equation  $\mathbf{AX} = \mathbf{B}$  when new rows or columns of data are added to data matrices  $\mathbf{A}$  and  $\mathbf{B}$  is thoroughly investigated. A computationally efficient algorithm which is based on the singular value decomposition is proposed. Computational results are presented for synthetic data and also for a machine learning application based on feature extraction for face recognition. The recursive solution to the symmetric Procrustes problem when including

extra data is also investigated. The experimental results suggest that the proposed algorithms are computationally more efficient when the matrices are high-dimensional and when they are augmented with a small number of rows or columns.

The extension of the proposed method to include other special classes of matrices like reflexive and anti-reflexive, Stiefel matrices or Toeplitz matrices merits further investigation. Future work will also consider the solution to the Procrustes problem with regularization constraints after modifying the matrices with the addition or deletion of data.

## Disclosure statement

No potential conflict of interest was reported by the author(s).

## References

- [1] E. Anderson, Z. Bai, and J. Dongarra, *Generalized QR factorization and its applications*, Linear Algebra Appl. 162–164(0) (1992), pp. 243–271.
- [2] R.H. Bartels and G.W. Stewart, *Solution of the matrix equation  $AX + XB = C$  [F4]*, Commun. ACM 15(9) (1972), pp. 820–826.
- [3] M. Brand, *Fast low-rank modifications of the thin singular value decomposition*, Linear Algebra Appl. 415(1) (2006), pp. 20–30.
- [4] J.R. Bunch and C.P. Nielsen, *Updating the singular value decomposition*, Numer. Math. 31(2) (1978), pp. 111–129.
- [5] J.R. Cardoso and K. Ziętak, *On a sub-Stiefel procrustes problem arising in computer vision*, Numer. Linear Algebra Appl. 22(3) (2015), pp. 523–547.
- [6] D. Chen, X. Cao, F. Wen, and J. Sun. *Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification*, in *2013 Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Portland, OR, USA, 2013, pp. 3025–3032.
- [7] R. Christensen, L.M. Pearson, and W. Johnson, *Case-deletion diagnostics for mixed models*, Technometrics 34(1) (1992), pp. 38–45.
- [8] K. Chu, *Singular value and generalized singular value decompositions and the solution of linear matrix equations*, Linear Algebra Appl. 88 (1987), pp. 83–98.
- [9] R.D. Cook, *Influential observations in linear regression*, J. Am. Stat. Assoc. 74(365) (1979), pp. 169–174.
- [10] C. Eckart and G. Young, *The approximation of one matrix by another of lower rank*, Psychometrika 1(3) (1936), pp. 211–218.
- [11] L. Eldén and H. Park, *A procrustes problem on the Stiefel manifold*, Numer. Math. 82(4) (1999), pp. 599–619.
- [12] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, *Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval*, IEEE Trans. Pattern Anal. Mac. Intell. 35(12) (2013 Dec), pp. 2916–2929.
- [13] J.C. Gower, *Generalized procrustes analysis*, Psychometrika 40(1) (1975), pp. 33–51.
- [14] J.C. Gower, *Procrustes methods*, Interdiscip. Rev. Comput. Stat. 2(4) (2010), pp. 503–508.
- [15] B.F. Green, *The orthogonal approximation of an oblique structure in factor analysis*, Psychometrika 17(4) (1952), pp. 429–440.
- [16] M. Gu and S.C. Eisenstat, *Downdating the singular value decomposition*, SIAM J. Matrix Anal. Appl. 16(3) (1995), pp. 793–810.
- [17] S. Hadjiantoni and E.J. Kontoghiorghes, *Estimating large-scale general linear and seemingly unrelated regressions models after deleting observations*, Stat. Comput. 27(2) (2017), pp. 349–361.
- [18] S. Hadjiantoni and E.J. Kontoghiorghes, *A recursive three-stage least squares method for large-scale systems of simultaneous equations*, Linear Algebra Appl. 536 (2018), pp. 210–227.
- [19] N.J. Higham, *The symmetric procrustes problem*, BIT Numer. Math. 28(1) (1988), pp. 133–143.
- [20] N.J. Higham and N. Strabić, *Bounds for the distance to the nearest correlation matrix*, SIAM J. Matrix Anal. Appl. 37(3) (2016), pp. 1088–1102.
- [21] D. Hua, *On the symmetric solutions of linear matrix equations*, Linear Algebra Appl. 131 (1990), pp. 1–7.
- [22] A.-P. Liao and Y. Lei, *Least-squares solution with the minimum-norm for the matrix equation  $(AXB, GXH) = (C, D)$* , Comput. Math. Appl. 50(3) (2005), pp. 539–549.
- [23] X. Liu, *Hermitian and non-negative definite reflexive and anti-reflexive solutions to  $AX = B$* , Int. J. Comput. Math. 95(8) (2018), pp. 1666–1671.
- [24] Y.H. Liu, *Ranks of least squares solutions of the matrix equation  $AXB = C$* , Comput. Math. Appl. 55(6) (2008), pp. 1270–1278.
- [25] J.R. Magnus and H. Neudecker, *Matrix Differential Calculus with Applications in Statistics and Econometrics*, 3rd ed., Wiley, Chichester, UK, 2007.

- [26] F.L. Markley, *Attitude determination using vector observations and the singular value decomposition*, J. Astronaut. Sci. 36(3) (1988), pp. 245–258.
- [27] C. Meng, X. Hu, and L. Zhang, *The skew-symmetric orthogonal solutions of the matrix equation  $AX = B$* , Linear Algebra Appl. 402 (2005), pp. 303–318.
- [28] C.I. Mosier, *Determining a simple structure when loadings for certain tests are known*, Psychometrika 4(2) (1939), pp. 149–162.
- [29] X.Y. Peng, X.Y. Hu, and L. Zhang, *The reflexive and anti-reflexive solutions of the matrix equation  $AHXB = C$* , J. Comput. Appl. Math. 200(2) (2007), pp. 749–760.
- [30] Y. Peng, W. Kong, and B. Yang, *Orthogonal extreme learning machine for image classification*, Neurocomputing 266 (2017), pp. 458–464.
- [31] Y. Qiu and A. Wang, *Solving balanced procrustes problem with some constraints by eigenvalue decomposition*, J. Comput. Appl. Math. 233(11) (2010), pp. 2916–2924.
- [32] M.B. Rubin and D. Solav, *Unphysical properties of the rotation tensor estimated by least squares optimization with specific application to biomechanics*, Int. J. Eng. Sci. 103 (2016), pp. 11–18.
- [33] P.H. Schönemann, *A generalized solution of the orthogonal procrustes problem*, Psychometrika 31(1) (1966), pp. 1–10.
- [34] V. Simoncini, *Computational methods for linear matrix equations*, SIAM Rev. 58(3) (2016), pp. 377–441.
- [35] J. Snyders and M. Zakai, *On nonnegative solutions of the equation  $AD + DA' = -C$* , SIAM J. Appl. Math. 18(3) (1970), pp. 704–714.
- [36] Y. Sun and L. Vandenberghe, *Decomposition methods for sparse matrix nearness problems*, SIAM J. Matrix Anal. Appl. 36(4) (2015), pp. 1691–1717.
- [37] Y. Tai, J. Yang, Y. Zhang, L. Luo, J. Qian, and Y. Chen, *Face recognition with pose variations and misalignment via orthogonal procrustes regression*, IEEE Trans. Image Process. 25(6) (2016), pp. 2673–2683.
- [38] Y. Tian and Y. Takane, *On consistency, natural restrictions and estimability under classical and extended growth curve models*, J. Stat. Plan. Inference 139(7) (2009), pp. 2445–2458.
- [39] J.A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, *Practical sketching algorithms for low-rank matrix approximation*, SIAM J. Matrix Anal. Appl. 38(4) (2017), pp. 1454–1485.
- [40] W.J. Vetter, *Vector structures and solutions of linear matrix equations*, Linear Algebra Appl. 10(2) (1975), pp. 181–188.
- [41] G. Wahba, *A least squares estimate of satellite attitude*, SIAM Rev. 7(3) (1965), pp. 409–409.
- [42] P.C. Young, *Recursive Estimation and Time-Series Analysis*, 2nd ed., Springer-Verlag, Berlin Heidelberg, 2011.
- [43] H. Zhao, Z. Wang, and F. Nie, *Orthogonal least squares regression for feature extraction*, Neurocomputing 216 (2016), pp. 200–207.