Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 8(1): 1672–1681.
DOI: 10.3934/math.2023085
Received: 18 August 2022
Revised: 17 September 2022
Accepted: 28 September 2022
Published: 24 October 2022

Research article

Updating QR factorization technique for solution of saddle point problems

Salman Zeb[1], Muhammad Yousaf[1], Aziz Khan[2,*], Bahaaeldin Abdalla[2] and Thabet Abdeljawad[2,3,*]

[1] Department of Mathematics, University of Malakand, Chakdara, Dir Lower, 18800, Pakistan
[2] Department of Mathematics and Sciences, Prince Sultan University, P.O. Box 66833, Riyadh, 11586, Saudi Arabia
[3] Department of Medical Research, China Medical University, Taichung, 40402, Taiwan

* **Correspondence:** Email: akhan@psu.edu.sa, tabdeljawad@psu.edu.sa.

Abstract: We consider saddle point problem and proposed an updating QR factorization technique for its solution. In this approach, instead of working with large system which may have a number of complexities such as memory consumption and storage requirements, we computed QR factorization of matrix A and then updated its upper triangular factor R by appending the matrices B and $\begin{pmatrix} B^T & -C \end{pmatrix}$ to obtain the solution. The QR factorization updated process consisting of updating of the upper triangular factor R and avoid the involvement of orthogonal factor Q due to its expensive storage requirements. This technique is also suited as an updating strategy when QR factorization of matrix A is available and it is required that matrices of similar nature be added to its right end or at bottom position for solving the modified problems. Numerical tests are carried out to demonstrate the applications and accuracy of the proposed approach.

Keywords: saddle point problem; QR factorization; Householder reflection; updating
Mathematics Subject Classification: 65-XX, 65Fxx, 65F05, 65F25

1. Introduction

Saddle point problems occur in many scientific and engineering applications. These applications inlcudes mixed finite element approximation of elliptic partial differential equations (PDEs) [1–3], parameter identification problems [4, 5], constrained and weighted least squares problems [6, 7], model order reduction of dynamical systems [8, 9], computational fluid dynamics (CFD) [10–12], constrained optimization [13–15], image registration and image reconstruction [16–18], and optimal control problems [19–21]. Mostly iterative solvers are used for solution of such problem due to its usual large, sparse or ill-conditioned nature. However, there exists some applications areas such as optimization problems and computing the solution of subproblem in different methods which requires direct methods for solving saddle point problem. We refer the readers to [22] for detailed survey.

The Finite element method (FEM) is usually used to solve the coupled systems of differential equations. The FEM algorithm contains solving a set of linear equations possessing the structure of the saddle point problem [23, 24]. Recently, Okulicka and Smoktunowicz [25] proposed and analyzed block Gram-Schmidt methods using thin Householder QR factorization for solution of 2×2 block linear system with emphasis on saddle point problems. Updating techniques in matrix factorization is studied by many researchers, for example, see [6, 7, 26–28]. Hammarling and Lucas [29] presented updating of the QR factorization algorithms with applications to linear least squares (LLS) problems. Yousaf [30] studied QR factorization as a solution tools for LLS problems using repeated partition and updating process. Andrew and Dingle [31] performed parallel implementation of the QR factorization based updating algorithms on GPUs for solution of LLS problems. Zeb and Yousaf [32] studied equality constraints LLS problems using QR updating techniques. Saddle point problems solver with improved Variable-Reduction Method (iVRM) has been studied in [33]. The analysis of symmetric saddle point systems with augmented Lagrangian method using Generalized Singular Value Decomposition (GSVD) has been carried out by Dluzewska [34]. The null-space approach was suggested by Scott and Tuma to solve large-scale saddle point problems involving small and non-zero (2,2) block structures [35].

In this article, we proposed an updating QR factorization technique for numerical solution of saddle point problem given as

$$Mz = f \Leftrightarrow \begin{pmatrix} A & B \\ B^T & -C \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}, \tag{1.1}$$

which is a linear system where $A \in \mathcal{R}^{p\times p}$, $B \in \mathcal{R}^{p\times q}$ $(q \leq p)$ has full column rank matrix, B^T represents transpose of the matrix B, and $C \in \mathcal{R}^{q\times q}$. There exists a unique solution $z = (x, y)^T$ of problem (1.1) if 2×2 block matrix M is nonsingular. In our proposed technique, instead of working with large system having a number of complexities such as memory consumption and storage requirements, we compute QR factorization of matrix A and then updated its upper triangular factor R by appending B and $\begin{pmatrix} B^T & -C \end{pmatrix}$ to obtain the solution. The QR factorization updated process consists of updating of the upper triangular factor R and avoiding the involvement of orthogonal factor Q due to its expensive storage requirements [6]. The proposed technique is not only applicable for solving saddle point problem but also can be used as an updating strategy when QR factorization of matrix A is in hand and one needs to add matrices of similar nature to its right end or at bottom position for solving the modified problems.

The paper is organized according to the following. The background concepts are presented in Section 2. The core concept of the suggested technique is presented in Section 3, along with a MATLAB implementation of the algorithm for problem (1.1). In Section 4 we provide numerical experiments to illustrate its applications and accuracy. Conclusion is given in Section 5.

2. Background study

Some important concepts are given in this section. These concepts will be used further in our main Section 3.

The QR factorization of a matrix $S \in \mathcal{R}^{p\times q}$ is defined as

$$S = QR,\ Q \in \mathcal{R}^{p\times p},\ R \in \mathcal{R}^{p\times q}, \tag{2.1}$$

where Q is an orthogonal matrix and R is an upper trapezoidal matrix. It can be computed using Gram Schmidt orthogonalization process, Givens rotations, and Householder reflections.

The QR factorization using Householder reflections can be obtained by successively pre-multiplying matrix S with series of Householder matrices $H_q \cdots H_2H_1$ which introduces zeros in all the subdiagonal elements of a column simultaneously. The $H \in \mathcal{R}^{q\times q}$ matrix for a non-zero Householder vector $u \in \mathcal{R}^q$ is in the form

$$H = I_q - \tau uu^T,\ \tau = \frac{2}{u^Tu}. \tag{2.2}$$

Householder matrix is symmetric and orthogonal. Setting

$$u = t \pm \|t\|_2\, e_1, \tag{2.3}$$

we have

$$Ht = t - \tau uu^T t = \mp\alpha e_1, \tag{2.4}$$

where t is a non-zero vector, α is a scalar, $\|\cdot\|_2$ is the Euclidean norm, and e_1 is a unit vector.

Choosing the negative sign in (2.3), we get positive value of α. However, severe cancellation error can occur if α is close to a positive multiple of e_1 in (2.3). Let $t \in \mathcal{R}^q$ be a vector and t_1 be its first element, then the following Parlett's formula [36]

$$u_1 = t_1 - \|t\|_2 = \frac{t_1^2 - \|t\|_2^2}{t_1 + \|t\|_2} = \frac{-(t_2^2 + \cdots + t_n^2)}{t_1 + \|t\|_2},$$

can be used to avoid the cancellation error in the case when $t_1 > 0$. For further details regarding QR factorization, we refer to [6, 7].

With the aid of the following algorithm, the Householder vector u required for the Householder matrix H is computed.

Algorithm 1 Computing parameter τ and Householder vector u [6]

Input: $t \in \mathcal{R}^q$
Output: $u,\ \tau$

```
σ = ||t||_2^2
u = t, u(1) = 1
if (σ = 0) then
    τ = 0
else
    μ = sqrt(t_1^2 + σ)
end if
if t_1 ≤ 0 then
    u(1) = t_1 − μ
else
    u(1) = −σ/(t_1 + μ)
end if
τ = 2u(1)^2/(σ + u(1)^2)
u = u/u(1)
```

3. Solution procedure

We consider problem (1.1) as

$$Mz = f,$$

where

$$M = \begin{pmatrix} A & B \\ B^T & -C \end{pmatrix} \in \mathcal{R}^{(p+q)\times(p+q)},\ z = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathcal{R}^{p+q},\ \text{and } f = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \in \mathcal{R}^{p+q}.$$

Computing QR factorization of matrix A, we have

$$\hat{R} = \hat{Q}^T A,\ \hat{d} = \hat{Q}^T f_1, \tag{3.1}$$

where $\hat{R} \in \mathcal{R}^{p\times p}$ is the upper triangular matrix, $\hat{d} \in \mathcal{R}^p$ is the corresponding right hand side (RHS) vector, and $\hat{Q} \in \mathcal{R}^{p\times p}$ is the orthogonal matrix. Moreover, multiplying the transpose of matrix $\hat{Q}$ with matrix $M_c = B \in \mathcal{R}^{p\times q}$, we get

$$N_c = \hat{Q}^T M_c \in \mathcal{R}^{p\times q}. \tag{3.2}$$

Equation (3.1) is obtained using MATLAB build-in command *qr* which can also be computed by constructing Householder matrices $H_1 \dots H_p$ using Algorithm 1 and applying Householder QR algorithm [6]. Then, we have

$$\hat{R} = H_p \dots H_1 A,\ \hat{d} = H_p \dots H_1 f_1,$$

where $\hat{Q} = H_1 \dots H_p$ and $N_c = H_p \dots H_1 M_c$. It gives positive diagonal values of $\hat{R}$ and also economical with respect to storage requirements and times of calculation [6].

Appending matrix N_c given in Eq (3.2) to the right end of the upper triangular matrix $\hat{R}$ in (3.1), we get

$$\acute{R} = \begin{bmatrix} \hat{R}(1:p, 1:p) & N_c(1:p, 1:q) \end{bmatrix} \in \mathcal{R}^{p\times(p+q)}. \tag{3.3}$$

Here, if the factor $\acute{R}$ has the upper triangular structure, then $\acute{R} = \bar{R}$. Otherwise, by using Algorithm 1 to form the Householder matrices $H_{p+1} \dots H_{p+q}$ and applying it to $\acute{R}$ as

$$\bar{R} = H_{p+q} \dots H_{p+1}\acute{R} \text{ and } \bar{d} = H_{p+q} \dots H_{p+1}\hat{d}, \tag{3.4}$$

we obtain the upper triangular matrix $\bar{R}$.

Now, the matrix $M_r = \begin{pmatrix} B^T & -C \end{pmatrix}$ and its corresponding RHS $f_2 \in \mathcal{R}^q$ are added to the $\bar{R}$ factor and $\bar{d}$ respectively in (3.4)

$$\bar{R}_r = \begin{pmatrix} \bar{R}(1:p, 1:p+q) \\ M_r(q:p+q, q:p+q) \end{pmatrix} \text{ and } \bar{d}_r = \begin{pmatrix} \bar{d}(1:p) \\ f_2(1:q) \end{pmatrix}.$$

Using Algorithm 1 to build the householder matrices $H_1 \dots H_{p+q}$ and apply it to $\bar{R}_r$ and its RHS $\bar{d}_r$, this implies

$$\tilde{R} = H_{p+q} \dots H_1 \begin{pmatrix} \bar{R} \\ M_r \end{pmatrix},\ \tilde{d} = H_{p+q} \dots H_1 \begin{pmatrix} \bar{d} \\ f_2 \end{pmatrix}.$$

Hence, we determine the solution of problem (1.1) as $\tilde{z} = backsub(\tilde{R}, \tilde{d})$, where *backsub* is the MATLAB built-in command for backward substitution.

The algorithmic representation of the above procedure for solving problem (1.1) is given in Algorithm 2.

Algorithm 2 Algorithm for solution of problem (1.1)

Input: $A \in \mathcal{R}^{p\times p},\ B \in \mathcal{R}^{p\times q},\ C \in \mathcal{R}^{q\times q},\ f_1 \in \mathcal{R}^{p},\ f_2 \in \mathcal{R}^{q}$
Output: $\tilde{z} \in \mathcal{R}^{p+q}$
  $[\hat{Q}, \hat{R}] = \mathbf{qr}(A)$, $\hat{d} = \hat{Q}^T f_1$, and $N_c = \hat{Q}^T M_c$
  $\hat{R}(1:p, q+1:p+q) = N_c(1:p, 1:q)$
  if $p \leq p+q$ **then**
    $\bar{R} = \mathbf{triu}(\hat{R})$, $\bar{d} = \hat{d}$
  else
    for $m = p-1$ to $min(p, p+q)$ **do**
      $[u, \tau, \hat{R}(m,m)] = \mathbf{householder}(\hat{R}(m,m), \hat{R}(m+1:p, m))$
      $W = \hat{R}(m, m+1:p+q) + u^T \hat{R}(m+1:p, m+1:p+q)$
      $\hat{R}(m, m+1:p+q) = \hat{R}(m, m+1:p+q) - \tau W$
      if $m < p+q$ **then**
        $\hat{R}(m+1:p, m+1:p+q) = \hat{R}(m+1:p, m+1:p+q) - \tau u W$
      end if
      $\bar{d}(m:p) = \hat{d}(m:p) - \tau \begin{pmatrix} 1 \\ u \end{pmatrix} \begin{pmatrix} 1 & u^T \end{pmatrix} \hat{d}(m:p)$
    end for
    $\bar{R} = \mathbf{triu}(\hat{R})$
  end if
  for $m = 1$ to $min(p, p+q)$ **do**
    $[u, \tau, \bar{R}(m,m)] = \mathbf{householder}(\bar{R}(m,m), M_r(1:q, m))$
    $W_1 = \bar{R}(m, m+1:p+q) + u^T M_r(1:q, m+1:p+q)$
    $\bar{R}(m, m+1:p+q) = \bar{R}(m, m+1:p+q) - \tau W_1$
    if $m < p+q$ **then**
      $M_r(1:q, m+1:p+q) = M_r(1:q, m+1:p+q) - \tau u W_1$
    end if
    $\bar{d}_m = \bar{d}(m)$
    $\bar{d}(m) = (1-\tau)\bar{d}(m) - \tau u^T f_2(1:q)$
    $f_3(1:q) = f_2(1:q) - \tau u \bar{d}_m - \tau u(u^T f_2(1:q))$
  end for
  if $p < p+q$ **then**
    $[\grave{Q}_r, \grave{R}_r] = \mathbf{qr}(M_r(:, p+1:p+q))$
    $\bar{R}(p+1:p+q, p+1:p+q) = \grave{R}_r$
    $f_3 = \grave{Q}_r^T f_2$
  end if
  $\tilde{R} = \mathbf{triu}(\bar{R})$
  $\tilde{d} = f_3$
  $\tilde{z} = \mathbf{backsub}\left(\tilde{R}(1:p+q, 1:p+q), \tilde{d}(1:p+q)\right)$

4. Numerical experiments

To demonstrate applications and accuracy of our suggested algorithm, we give several numerical tests done in MATLAB in this section. Considering that $z = (x, y)^T$ be the actual solution of the problem (1.1) where $x = ones(p, 1)$ and $y = ones(q, 1)$. Let $\tilde{z}$ be our proposed Algorithm 2 solution. In our test examples, we consider randomly generated test problems of different sizes and compared the results with the block classical block Gram-Schmidt re-orthogonalization method (BCGS2) [25]. Dense matrices are taken in our test problems. We carried out numerical experiments as follow.

Example 1. *We consider*

$$A = \frac{A_1 + A_1'}{2},\ B = randn('state', 0),\ and\ C = \frac{C_1 + C_1'}{2},$$

where $randn('state', 0)$ is the MATLAB command used to reset to its initial state the random number generator; $A_1 = P_1 D_1 P_1'$, $C_1 = P_2 D_2 P_2'$, $P_1 = orth(rand(p))$ and $P_2 = orth(rand(q))$ are randomly orthogonal matrices, $D_1 = logspace(0, -k, p)$ and $D_2 = logspace(0, -k, q)$ are diagonal matrices which generates p and q points between decades 1 and 10^{-k} respectively. We describe the test matrices in Table 1 by giving its size and condition number κ. The condition number κ for a matrix S is defined as $\kappa(S) = \|S\|_2 \left\|S^{-1}\right\|_2$. Moreover, the results comparison and numerical illustration of backward error tests of the algorithm are given respectively in Tables 2 and 3.

Table 1. Test problems description.

Problem	$size(A)$	$\kappa(A)$	$size(B)$	$\kappa(B)$	$size(C)$	$\kappa(C)$
(1)	16×16	1.0000e+05	16×9	6.1242	9×9	1.0000e+05
(2)	120×120	1.0000e+05	120×80	8.4667	80×80	1.0000e+05
(3)	300×300	1.0000e+06	300×200	9.5799	200×200	1.0000e+06
(4)	400×400	1.0000e+07	400×300	13.2020	300×300	1.0000e+07
(5)	900×900	1.0000e+08	900×700	15.2316	700×700	1.0000e+08

Table 2. Numerical results.

Problem	$size(M)$	$\kappa(M)$	$\frac{\|z-\tilde{z}\|_2}{\|z\|_2}$	$\frac{\|z-z_{BCGS2}\|_2}{\|z\|_2}$
(1)	25×25	7.7824e+04	6.9881e-13	3.3805e-11
(2)	200×200	2.0053e+06	4.3281e-11	2.4911e-09
(3)	500×500	3.1268e+07	1.0582e-09	6.3938e-08
(4)	700×700	3.5628e+08	2.8419e-09	4.3195e-06
(5)	1600×1600	2.5088e+09	7.5303e-08	3.1454e-05

Table 3. Backward error tests results.

Problem	$\frac{\|M-\tilde{Q}\tilde{R}\|_F}{\|M\|_F}$	$\|I-\tilde{Q}^T\tilde{Q}\|_F$
(1)	6.7191e-16	1.1528e-15
(2)	1.4867e-15	2.7965e-15
(3)	2.2052e-15	4.1488e-15
(4)	2.7665e-15	4.9891e-15
(5)	3.9295e-15	6.4902e-15

The relative errors for the presented algorithm and its comparison with BCGS2 method in Table 2 showed that the algorithm is applicable and have good accuracy. Moreover, the numerical results for backward stability analysis of the suggested updating algorithm is given in Table 3.

Example 2. *In this experiment, we consider $A = H$ where H is a Hilbert matrix generated with MATLAB command hilb(p). It is symmetric, positive definite, and ill-conditioned matrix. Moreover, we consider test matrices B and C similar to that as given in Example 1 but with different dimensions. Tables 4–6 describe the test matrices, numerical results and backward error results, respectively.*

Table 4. Test problems description.

Problem	$size(A)$	$\kappa(A)$	$size(B)$	$\kappa(B)$	$size(C)$	$\kappa(C)$
(6)	6×6	1.4951e+07	6×3	2.6989	3×3	1.0000e+05
(7)	8×8	1.5258e+10	8×4	2.1051	4×4	1.0000e+06
(8)	12×12	1.6776e+16	12×5	3.6108	5×5	1.0000e+07
(9)	13×13	1.7590e+18	13×6	3.5163	6×6	1.0000e+10
(10)	20×20	2.0383e+18	20×10	4.4866	10×10	1.0000e+10

Table 5. Numerical results.

Problem	$size(M)$	$\kappa(M)$	$\frac{\|z-\tilde{z}\|_2}{\|z\|_2}$	$\frac{\|z-z_{BCGS2}\|_2}{\|z\|_2}$
(6)	9×9	8.2674e+02	9.4859e-15	2.2003e-14
(7)	12×12	9.7355e+03	2.2663e-13	9.3794e-13
(8)	17×17	6.8352e+08	6.8142e-09	1.8218e-08
(9)	19×19	2.3400e+07	2.5133e-10	1.8398e-09
(10)	30×30	8.0673e+11	1.9466e-05	1.0154e-03

Table 6. Backward error tests results.

Problem	$\frac{\|M-\tilde{Q}\tilde{R}\|_F}{\|M\|_F}$	$\|I-\tilde{Q}^T\tilde{Q}\|_F$
(6)	5.0194e-16	6.6704e-16
(7)	8.4673e-16	1.3631e-15
(8)	7.6613e-16	1.7197e-15
(9)	9.1814e-16	1.4360e-15
(10)	7.2266e-16	1.5554e-15

From Table 5, it can bee seen that the presented algorithm is applicable and showing good accuracy. Table 6 numerically illustrates the backward error results of the proposed Algorithm 2.

5. Conclusions

In this article, we have considered the saddle point problem and studied updated of the Householder *QR* factorization technique to compute its solution. The results of the considered test problems with dense matrices demonstrate that the proposed algorithm is applicable and showing good accuracy to solve saddle point problems. In future, the problem can be studied further for sparse data problems which are frequently arise in many applications. For such problems updating of the Givens *QR* factorization will be effective to avoid unnecessary fill-in in sparse data matrices.

Acknowledgments

The authors Aziz Khan, Bahaaeldin Abdalla and Thabet Abdeljawad would like to thank Prince Sultan university for paying the APC and support through TAS research lab.

Conflict of interest

There does not exist any kind of competing interest.

References

1. F. Brezzi, On the existence, uniqueness and approximation of saddle-point problems arising from lagrangian multipliers, *ESAIM Math. Model. Num.*, **8** (1974), 129–151. https://doi.org/10.1051/m2an/197408R201291
2. F. Brezzi, M. Fortin, *Mixed and hybrid finite element methods*, New York: Springer, 1991.
3. A. Quarteroni, A. Valli, *Numerical approximation of partial differential equations*, Cham: Springer, 1994.
4. M. Burger, W. Mühlhuber, Iterative regularization of parameter identification problems by sequential quadratic programming methods, *Inverse Probl.*, **18** (2002), 943–969. https://doi.org/10.1088/0266-5611/18/4/301
5. E. Haber, U. M. Ascher, Preconditioned all-at-once methods for large sparse parameter estimation problems, *Inverse Probl.*, **17** (2001), 1847–1864. https://doi.org/10.1088/0266-5611/17/6/319
6. G. H. Golub, C. F. Van Loan, *Matrix computations*, Baltimore: Johns Hopkins University Press, 1996.
7. Å. Björck, *Numerical methods for least squares problems*, Philadelphia: SIAM, 1996.
8. R. W. Freund, Model reduction methods based on krylov subspaces, *Acta Numer.*, **12** (2003), 267–319. https://doi.org/10.1017/S0962492902000120
9. T. Stykel, Balanced truncation model reduction for semidiscretized stokes equation, *Linear Algebra Appl.*, **415** (2006), 262–289. https://doi.org/10.1016/j.laa.2004.01.015

10. R. Glowinski, *Lectures on numerical methods for non-linear variational problems*, Berlin, Heidelberg: Springer, 2008.
11. S. Turek, *Efficient solvers for incompressible flow problems: An algorithmic and computational approache*, Berlin, Heidelberg: Springer, 1999.
12. P. Wesseling, *Principles of computational fluid dynamics*, Berlin, Heidelberg: Springer, 2009.
13. P. E. Gill, W. Murray, D. B. Ponceleón, M. A. Saunders, Preconditioners for indefinite systems arising in optimization, *SIAM J. Matrix Anal. Appl.*, **13** (1992), 292–311. https://doi.org/10.1137/0613022
14. P. E. Gill, W. Murray, M. H. Wright, *Practical optimization*, New York: Academic Press, 1981.
15. S. J. Wright, *Primal-dual interior-point methods*, Philadelphia: SIAM, 1997.
16. E. Haber, J. Modersitzki, Numerical methods for volume preserving image registration, *Inverse Probl.*, **20** (2004), 1621–1638. https://doi.org/10.1088/0266-5611/20/5/018
17. E. Hall, *Computer image processing and recognition*, New York: Academic Press, 1979.
18. J. Modersitzki, *Numerical methods for image registration*, New York: Oxford University Press, 2003. https://doi.org/10.1093/acprof:oso/9780198528418.001.0001
19. A. Battermann, M. Heinkenschloss, Preconditioners for Karush-Kuhn-Tucker matrices arising in the optimal control of distributed systems, In: *Control and estimation of distributed parameter systems*, Basel: Birkhäuser,1998. https://doi.org/10.1007/978-3-0348-8849-3-2
20. G. Biros, O. Ghattas, Parallel preconditioners for KKT systems arising in optimal control of viscous incompressible flows, In: *Parallel computational fluid dynamics*, Amsterdam: North Holland, 2000. https://doi.org/10.1016/B978-044482851-4/50017-7
21. A. Battermann, E. W. Sachs, Block preconditioners for KKT systems in PDE-governed optimal control problems, In: *Fast solution of discretized optimization problems*, Basel: Birkhäuser, 2001. https://doi.org/10.1007/978-3-0348-8233-0-1
22. M. Benzi, G. H. Golub, J. Liesen, Numerical solution of saddle point problems, *Acta Numer.*, **14** (2005), 1–137. https://doi.org/10.1017/S0962492904000212
23. J. M. Dłużewski, Nonlinear problems during consolidation process, In: *Advanced numerical applications and plasticity in geomechanics*, Vienna: Springer, 2001. https://doi.org/10.1007/978-3-7091-2578-6-4
24. F. Okulicka, Solving coupled consolidation equations, In: *Numerical methods and applications*, Berlin, Heidelberg: Springer, 2007. https://doi.org/10.1007/978-3-540-70942-8-11
25. F. Okulicka, A. Smoktunowicz, Numerical solution of 2×2 block linear systems by block Gram-Schmidt methods, *Int. J. Comput. Math.*, **94** (2016), 1562–1573. https://doi.org/10.1080/00207160.2016.1226287
26. J. W. Daniel, W. B. Gragg, L. Kaufman, G. W. Stewart, Reorthogonalization and stable algorithms for updating the Gram-Schmidt *QR* factorization, *Math. Comput.*, **30** (1976), 772–795. https://doi.org/10.1090/S0025-5718-1976-0431641-8
27. P. E. Gill, G. H. Golub, W. Murray, M. A. Saunders, Methods for modifying matrix factorizations, *Math. Comput.*, **28** (1974), 505–535.

28. L. Reichel, W. B. Gragg, Algorithm 686: FORTRAN subroutines for updating the *QR* decomposition, *ACM T. Math. Software*, **16** (1990), 369–377. https://doi.org/10.1145/98267.98291

29. S. Hammarling, C. Lucas, Updating the *QR* factorization and the least squares problem, *Univ. Manchester*, 2008.

30. M. Yousaf, Repeated updating as a solution tool for linear least squares problems, *Univ. Essex*, 2010.

31. R. Andrew, N. Dingle, Implementing *QR* factorization updating algorithms on GPUs, *Parallel Comput.*, **40** (2014), 161–172. https://doi.org/10.1016/j.parco.2014.03.003

32. S. Zeb, M. Yousaf, Updating *QR* factorization procedure for solution of linear least squares problem with equality constraints, *J. Inequal. Appl.*, **2017** (2017), 281. https://doi.org/10.1186/s13660-017-1547-0

33. A. Kamitani, T. Takayama, A. Saitoh, H. Nakamura, Linear-system solver for EFG-type saddle-point problem without using QR decomposition, *Plasma Fusion Res.*, **17** (2022), 2403014. https://doi.org/10.1585/pfr.17.2403014

34. F. O. Dluzewska, Applying the GSVD to the analysis of the augmented Lagrangian method for symmetric saddle point problem, In: *Novel research aspects in mathematical and computer science*, 2022. https://doi.org/10.9734/bpi/nramcs/v3/2072A

35. J. Scott, M. Tuma, A null-space approach for large-scale symmetric saddle point systems with a small and non zero (2, 2) block, *Numer. Algor.*, **90** (2022), 1639–1667. https://doi.org/10.1007/s11075-021-01245-z

36. B. N. Parlett, Analysis of algorithms for reflections in bisectors, *SIAM Rev.*, **13** (1971), 197–208. https://doi.org/10.1137/1013037