Low-rank Updates of Balanced Incomplete Factorization preconditioners[*]

J. Cerdán J. Marín J. Mas
Institut de Matemàtica Multidisciplinar,
Universitat Politècnica de València,
46022 València, Spain
jcerdan@imm.upv.es, jmarinma@imm.upv.es, jmasm@imm.upv.es

Abstract

Let $Ax = b$ be a large and sparse system of linear equations where A is a nonsingular matrix. An approximate solution is frequently obtained by applying preconditioned iterations. Consider the matrix $B = A + PQ^T$ where $P, Q \in \mathbb{R}^{n\times k}$ are full rank matrices. In this work we study the problem of updating a previously computed preconditioner for A in order to solve the updated linear system $Bx = b$ by preconditioned iterations. In particular we propose a method for updating a Balanced Incomplete Factorization preconditioner. The strategy is based on the computation of an approximate Inverse Sherman-Morrison decomposition for an equivalent augmented linear system. Approximation properties of the preconditioned matrix and an analysis of the computational cost of the algorithm are studied. Moreover, the results of the numerical experiments with different types of problems show that the proposed method contributes to accelerate the convergence.

Keywords: Iterative methods; preconditioning; low rank update; balanced incomplete factorization; sparse linear systems

1 Introduction

Let A be a nonsingular matrix and consider the matrix B given by

$$B = A + PQ^T \tag{1}$$

where P and Q are $n \times k$ rectangular matrices with $k \ll n$. We assume that B is a nonsingular matrix and that P and Q have full rank. We are interested in solving the updated linear system

$$Bx = b \tag{2}$$

[*] This work was supported by the Spanish Ministerio de Economía y Competitividad under grant MTM2014-58159-P.

by preconditioned iterations updating a previously computed preconditioner for A.

The problem of updating a preconditioner for a matrix arise in many applications of numerical linear algebra. Very often it is coupled with the problem of solving a sequence of linear systems as, for example, the solution of nonlinear equations by a Newton-like method [26]. Quasi-Newton methods involve low-rank updates of an approximate factorization of an initial Jacobian matrix [3, 4]. Other approaches are based on an estimation of the difference between a reference matrix and the updated one [16]. In this context are of special interest the so called matrix-free algorithms that do not require forming explicitly the Jacobian matrix, see [17, 1] and the references therein. These approaches follow somehow the ideas found in [2, 7] and generally involve a rank-n correction of an initial factorized preconditioner. A different scenario corresponds to the case where a given matrix can be split in two parts as $A = M - N$, with M being a well conditioned matrix for which a proper preconditioner can be computed efficiently, and the matrix N has low-rank. This kind of problems arise in the application of the boundary integral methods where the Green's function can be approximated by a numerically low-rank matrix [22], spectral-tau methods for Navier-Stokes equations [6], wave propagation phenomena and topology optimization [19, 20]. The matrix N can be approximated as a product of two matrices by applying a singular value decomposition [21], [23], or with probabilistic algorithms for constructing matrix decompositions [25]. In [18] the authors propose a preconditioner based on splitting a nearly Hermitian matrix A into its Hermitian and skew-Hermitian parts. Finally, in overdetermined least squares problems that involve the addition of a set of new equations, the normal equations can be formulated as a rank-k update of the normal equations for the initial matrix [24].

In this paper we present an algebraic algorithm for problems whose system matrix can be written in the form (1), i.e., it comes from a low-rank update. In particular, we are interested on updating an initial Balanced Incomplete Factorization (BIF) preconditioner. BIF is based on the Inverse Sherman-Morrison (ISM) decomposition of a matrix and it is able to compute both the LDU factorization and its inverse at the same time with a process that exhibits mutual influence between both factorizations throughout the computation [10, 13]. In addition, with suitable dropping strategies a balance in the factors can be achieved, helping to control the conditioning of the factors. These properties give raise to robust preconditioners for a wide variety of problems as numerical results presented in [11, 12] confirm. Our aim is to carry out the updating computations in such a way that the updated preconditioner exhibits the same characteristics.

The paper is organized as follows. In section 2 we review the main characteristics of the ISM decomposition and the BIF preconditioner. The method to update a given BIF preconditioner is presented in section 3 and the preconditioning step is described in section 4. An analysis of the computational complexity is done in section 5. The approximation properties of the updated preconditioner are analyzed in section 6. In section 7 we present the results of the numerical experiments for different problems. Finally, in section 8 we give our conclusions.

2 ISM decomposition and BIF preconditioner

In this section we review the main characteristics of BIF preconditioners that are based on the computation of an approximate ISM decomposition. ISM was introduced in [10] as a method to compute an approximate inverse preconditioner. The authors showed that by applying repeatedly the Sherman-Morrison formula a factorization of a shifted inverse of a given matrix A can be obtained. Consider the square matrices A_0 and Y given by

$$A_0 = sI, \qquad Y = (A - A_0)^T,$$

where s is a positive parameter and I is the identity matrix. By applying recursively, for $j = 1, \dots, n$, the formulas

$$\begin{aligned} z_j &= e_j - \sum_{i=1}^{j-1} \frac{v_{ji}}{sr_i} z_i, \\ v_j &= y_j - \sum_{i=1}^{j-1} \frac{a^j z_i}{sr_i} v_i, \\ r_j &= 1 + \frac{y_j^T z_j}{s} = \frac{a^j z_j}{s} = 1 + \frac{v_{jj}}{s} \end{aligned} \tag{3}$$

where e_j and e^j are the jth column and row respectively of the identity matrix, and a^j is the jth row of A. Let Z and V_s be the matrices whose columns are the vectors z_j and v_j, and D_s a diagonal matrix containing the scalars r_j as defined in (3). Then, the expression

$$s^{-1}I - A^{-1} = s^{-2} Z D_s^{-1} V_s^T \tag{4}$$

is called an ISM decomposition of A. This decomposition is not unique since it depends of the value of the parameter s, but two different ISM decompositions obtained for different parameters are closely related [11]. What it is more interesting for us in this paper is that the ISM decomposition and the LDU factorization of a matrix are strongly connected. In [12] it is proved the following theorem.

Theorem 1. *A square matrix A has LDU factorization, $A = LDU$, if and only if it has ISM decomposition $s^{-1}I - A^{-1} = s^{-2} Z D_s^{-1} V_s^T$.*

Then

$$D = s^{-1} D_s, \qquad U = Z^{-1} \qquad \text{and} \qquad V_s = U^T D - sL^{-T}.$$

That is, the ISM decomposition contains the D (scaled) and U factors of the LDU factorization of A, and the factors L^{-1}, U^{-1} of the LDU factorization of A^{-1}. When A is a symmetric positive definite matrix all factors of the LDL^T factorizations of A are available and it is possible to get a Cholesky preconditioner computing an incomplete ISM decomposition. In the general case, one can get the missed factor from the ISM decomposition of A^T, as it is proposed in [12]. Let

$$s^{-1}I - A^{-T} = s^{-2}\tilde{Z}\tilde{D}_s^{-1}\tilde{V}_s^T,$$

the ISM decomposition of A^T. Then

$$\tilde{V}_s = LD - sU^{-1} \ .$$

Thus, the matrices V_s and $\tilde{V}_s$ contain not only the LDU factorization of A but also its inverse as it is graphically shown in Figure 1. An analysis of the expressions (3) shows that there is a computational influence between the entries of the matrices L and L^{-1} (U and U^{-1} as well), that is, the computed factors do influence each other numerically. Furthermore, the computation of the upper triangular part of the matrices V_s and $\tilde{V}_s$ can be simplified, as it is stated in Lemma 3.1 of [12]

$$v_{pj} = s\frac{\tilde{v}_{jp}}{d_p} - \sum_{i=p+1}^{j-1} \frac{\tilde{v}_{ji}}{d_i} v_{pi}, \qquad \text{for } p < j. \tag{5}$$

The BIF preconditioner uses this mutual influence between factors to compute an approximate ISM decomposition of a matrix. An additional feature is that, since the LDU factorizations of A and its inverse are stored in matrices V_s and $\tilde{V}_s$, it is possible to apply special dropping strategies. In [8, 9] the authors propose a robust strategy that drops entries by value compared with the norm of a row of the inverse. That is, given a drop tolerance τ, an entry l_{ij} in the jth column of the factor L is zeroed if

$$|l_{ij}| \|e^j L^{-1}\| \leq \tau.$$

Also the dual dropping criterion can be used. Thus, if ℓ_{ij} denotes an entry of the jth row of the factor L^{-1}, it is zeroed if

$$|\ell_{ij}| \|e^j L\| \leq \tau.$$

The same dropping strategy can be applied to obtain approximations of U and U^{-1}. Since these factors are explicitly available in the ISM decomposition, the BIF algorithm implements this technique to sparsify matrices in a straightforward manner. This dual dropping strategy together with the interlaced computations mentioned above, give raise to robust and efficient preconditioners for a wide variety of problems as the numerical experiments in [11, 12] show. Our aim is to update an existing BIF preconditioner mantaining these characteristics throughout the update process and at the same time, making the computation as simple as possible.

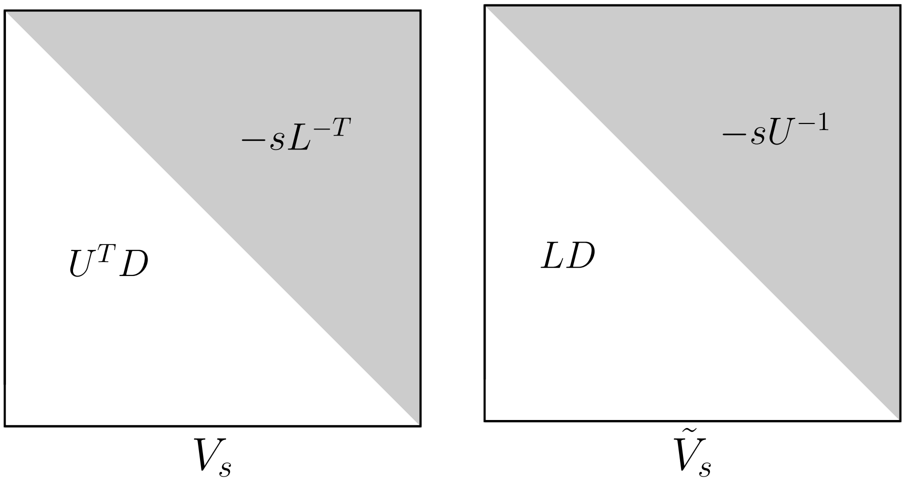


Figure 1: Factors of the LDU factorizations of A and A^{-1} included in V_s and $\tilde{V}_s$. White parts are factors of A, while grey parts are factors of A^{-1}.

3 Preconditioner update algorithm

As we stated in the Introduction we consider the iterative solution of the linear system

$$Bx = b$$

where $B = A + PQ^T$, and $P, Q \in \mathbb{R}^{n\times k}$ are full rank matrices. We study the problem of updating a previously computed BIF preconditioner for the matrix A in order to solve the updated linear system with a Krylov iterative method. To simplify notation the factors of the computed preconditioner will be denoted as L, D, and U. We note that, due to the recursive nature of the computation formulas of the preconditioner (3), updating directly the entries of a BIF preconditioner can be difficult and expensive. To simplify this task we will work instead with the matrix of the equivalent augmented linear system with $n+k$ equations $\mathbf{Bx} = \mathbf{b}$,

$$\left[\begin{array}{cc} A & P \\ -Q^T & I \end{array}\right]\left[\begin{array}{c} x \\ Q^Tx \end{array}\right] = \left[\begin{array}{c} b \\ 0 \end{array}\right]. \tag{6}$$

The computation of incomplete ISM decompositions for the matrices $\mathbf{B}$ and $\mathbf{B^T}$ is done by applying directly the BIF algorithm to these matrices. Thus, an approximation of the LDU factorization of $\mathbf{B}$ is obtained and it will be used as a preconditioner to compute iteratively an approximate solution of the updated linear system (2).

It is of general agreement that algorithms that exploit the block structure of a matrix, whenever it is possible, may benefit of an improved performance. Therefore, we find of interest trying to exploit the structure of $\mathbf{B}$ and $\mathbf{B}^T$.

Note from (6) that the block structure of these matrices not only depends on the rank of the update, that determines the size of P and Q, but also on A being structured itself that may induce a partition of these matrices. Thus, the computation of the ISM decompositions of $\mathbf{B}$ and $\mathbf{B}^T$ will be formulated by blocks [13]. The block ISM decomposition and its relation with the block LU factorization of a matrix is given by

$$\begin{aligned} \mathbf{B}^{-1} &= s^{-1}\mathbf{I} - s^{-2}\mathbf{Z}\mathbf{D}_s^{-1}\mathbf{V}_s^T \\ \mathbf{B}^{-T} &= s^{-1}\mathbf{I} - s^{-2}\tilde{\mathbf{Z}}\tilde{\mathbf{D}}_s^{-1}\tilde{\mathbf{V}}_s^T \end{aligned}$$

where

$$\mathbf{Z} = \mathbf{U}^{-1} \qquad , \qquad \mathbf{V}_s = \mathbf{U}^T\mathbf{D}^T - s\mathbf{L}^{-T} \tag{7}$$

and

$$\tilde{\mathbf{Z}} = \mathbf{L}^{-T} \qquad , \qquad \tilde{\mathbf{V}}_s = \mathbf{L}\mathbf{D} - s\mathbf{U}^{-1}\ . \tag{8}$$

These expressions are an extension of Theorem 1 for block structured matrices, and the only difference resides in the transposition of the block diagonal matrix $\mathbf{D}$ in (7), see [14].

An immediate advantage of working with the augmented linear system (6) is that computing from scratch the ISM decompositions of the matrices $\mathbf{B}$ and $\mathbf{B^T}$ is avoided. If we assume that the matrices A and A^T are not structured and their point ISM decompositions are available, then, it is natural to identify the following block structure for the $(n+k)\times(n+k)$ matrices $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_\mathbf{s}$

$$\mathbf{V_s} = \left[\begin{array}{ccc|c} & & & V_{1,n+1} \\ & V_s & & \vdots \\ & & & V_{n,n+1} \\ V_{n+1,1} & \cdots & V_{n+1,n} & V_{n+1,n+1} \end{array}\right] \tag{9}$$

and

$$\tilde{\mathbf{V}}_\mathbf{s} = \left[\begin{array}{ccc|c} & & & \tilde{V}_{1,n+1} \\ & \tilde{V}_s & & \vdots \\ & & & \tilde{V}_{n,n+1} \\ \tilde{V}_{n+1,1} & \cdots & \tilde{V}_{n+1,n} & \tilde{V}_{n+1,n+1} \end{array}\right], \tag{10}$$

where the $n\times n$ principal submatrices V_s and $\tilde{V}_s$ correspond to the ISM decompositions of A and A^T, respectively. Thus, only the last k rows and columns must be computed for both matrices. In equation (9) the new k entries in the jth column and row are denoted by $V_{n+1,j}$ and $V_{j,n+1}$, $j = 1,\dots,n$, respectively. Moreover, the block $V_{n+1,n+1}$ contains the last $k\times k$ entries that must be computed. Similar notation is used for the matrix $\tilde{V}_s$ in equation (10). The computation of the preconditioner update and the application of the dropping element criteria are formulated according to this structure.

From equation (7) it follows that $V_{n+1,j} = U_{j,n+1}^T d_j = P^T(L^{-T})_j$, where the last equality is obtained considering the relation between the LDU factorization of the matrices $\mathbf{B}$ and A. From the ISM decomposition of A^T one has that $(L^{-T})_j = \tilde{z}_j$, and therefore $V_{n+1,j} = P^T\tilde{z}_j$ for $j = 1, \ldots, n$. The elements of the last k rows of $\tilde{\mathbf{V}}_\mathbf{s}$ can be obtained analogously from equation (8). Thus,

$$\begin{array}{rcl} V_{n+1,j} & = & P^T\tilde{z}_j \ , \ j = 1, \ldots, n \\ \tilde{V}_{n+1,j} & = & -Q^T z_j \ , \ j = 1, \ldots, n \end{array} \tag{11}$$

From equation (5), it follows that the last k columns in a given row j of the matrices $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_\mathbf{s}$ are given by

$$\begin{array}{ll} V_{j,n+1} & = \dfrac{s}{d_j}\tilde{V}^T_{n+1,j} - \sum_{i=j+1}^n \dfrac{v_{j,i}}{d_i}\tilde{V}^T_{n+1,i} \ , \ j = 1, \ldots, n \\ \tilde{V}_{j,n+1} & = \dfrac{s}{d_j}V^T_{n+1,j} - \sum_{i=j+1}^n \dfrac{\tilde{v}_{j,i}}{d_i}V^T_{n+1,i} \ , \ j = 1, \ldots, n \end{array} \tag{12}$$

The expression for the last $k \times k$ diagonal block is easily derived from equation (7) and taking into account the form of the block LU factorization of $\mathbf{B}$,

$$V^T_{n+1,n+1} = \tilde{V}_{n+1,n+1} = -s^{-1}\left[\begin{array}{ccc} V^T_{1,n+1} & \cdots & V^T_{n,n+1} \end{array}\right] P + (1-s)I_k. \tag{13}$$

The updated ISM decompositions need additional storage for $4k$ sparse vectors of size n and a $k \times k$ matrix. This amount of memory is moderate provided that $k \ll n$ and the new vectors are kept sparse. The explicit computation of the matrices $\mathbf{Z}$ and $\tilde{\mathbf{Z}}$ can be avoided since they could be only required in the framework of a sequence of rank-k updates. Note that their entries, in exact arithmetic, can be extracted from $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_\mathbf{s}$. Nevertheless, as the authors report in [12], it is recommended performing the computations with these matrices in case of instabilities.

The updated BIF preconditioner is obtained after applying a dropping strategy to the new computed rows and columns. The strategy employed has been briefly reviewed in the previous section but, instead of applying the rule element by element, to be consistent with the block structure identified, the last k entries of a row or column are zeroed at once. A new block vector computed from equations (11) or (12) is dropped if the following conditions are satisfied. First, for dropping elements in the last k rows of $\mathbf{V_s}$ that correspond to $\mathbf{U^T D}$, it is used the criterion

$$\|d_j^{-1}V_{n+1,j}\|_\infty \|U^{-1}e_j\| \le \tau, \quad j = 1, \ldots, n \tag{14}$$

Similarly, for $\tilde{\mathbf{V}}_\mathbf{s}$ that contains the entries of $\mathbf{LD}$

$$\|d_j^{-1}\tilde{V}_{n+1,j}\|_\infty \|e^j L^{-1}\| \le \tau, \quad j = 1, \ldots, n \tag{15}$$

Note that in these expressions the norms of the inverse factors of the matrix A are needed and they are available from its ISM decomposition. By contrast, to drop entries in the last k columns of $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_\mathbf{s}$ the norm of the new rows in these matrices must be evaluated. The entries in the last k columns contain the inverse factors $\mathbf{sL^{-T}}$ and $\mathbf{sU^{-1}}$, respectively, and are also dropped blockwise according to the rules

$$\|s^{-1}V_{j,n+1}\|_\infty \|e^{n+1:n+k}\mathbf{L}\| \leq k\tau, \quad j = 1, \dots, n \tag{16}$$

and

$$\|s^{-1}\tilde{V}_{j,n+1}\|_\infty \|\mathbf{U}e_{n+1:n+k}\| \leq k\tau, \quad j = 1, \dots, n \tag{17}$$

The quantities $\|e^{n+1:n+k}\mathbf{L}\|$ and $\|\mathbf{U}e_{n+1:n+k}\|$ represent the norm of last k rows and columns of the LU factors of the matrix $\mathbf{B}$. Observe in equation (19) below, that they correspond to the norm of the blocks containing the last k rows of the matrices $\tilde{\mathbf{V}}_{\mathbf{s}}$ and $\mathbf{V_s}$, respectively. Finally, we recall that dropping elements in the last diagonal block computed from equation (13) may be not necessary since we are assuming $k \ll n$, and if it is computed after dropping elements in the new columns of the matrix $\mathbf{V_s}$, fill-in on this block can be moderate. Algorithm 1 summarizes the steps to update the preconditioner.

Algorithm 1 Preconditioner update computation

```
**Input:**Incomplete ISM decompositions of $A$, $A^T$, and matrices $P, Q$
**Output:** Last $k$ rows and columns of $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_{\mathbf{s}}$.
**for** $j \leftarrow 1, n$ **do**
    Compute $V_{n+1,j}$ and $\tilde{V}_{n+1,j}$ from equations (11)
    Drop entries in $V_{n+1,j}$ and $\tilde{V}_{n+1,j}$ applying (14) and (15)
**end for**
**for** $j \leftarrow 1, n$ **do**
    Compute $V_{j,n+1}$ and $\tilde{V}_{j,n+1}$ from equations (12)
    Drop entries in $V_{n+1,j}$ and $\tilde{V}_{n+1,j}$ applying (16) and (17)
**end for**
Compute $V_{n+1,n+1}$ and $\tilde{V}_{n+1,n+1}$ from equation (13)
```

4 Preconditioner application

The preconditioning step for a Krylov subspace iterative method tipically consists in obtaining the solution of linear systems of the form $My = r$, where M is the preconditioner, r is the residual and y is the preconditioned vector. With the method proposed the application of the updated BIF preconditioner for the linear system (2) implies the solution of an augmented linear system of the form

$$\mathbf{M} \begin{bmatrix} y \\ Q^T y \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}, \tag{18}$$

where $\mathbf{M}$ is the updated BIF preconditioner for the linear system (6). We recall that the BIF preconditioner for the matrix $\mathbf{B}$ is available from the factors $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_{\mathbf{s}}$ computed as it was described in the previous section. It follows that

$\mathbf{M} = \mathbf{LDU} \approx \mathbf{B}$ with

$$\mathbf{L} = \left[\begin{array}{ccc} & L & O \\ d_1^{-1}\tilde{V}_{n+1,1} & \cdots \; d_n^{-1}\tilde{V}_{n+1,n} & I \end{array} \right], \quad \mathbf{U} = \left[\begin{array}{cc} & d_1^{-1}V_{n+1,1}^T \\ U & \vdots \\ & d_n^{-1}V_{n+1,n}^T \\ O & I \end{array} \right] \tag{19}$$

and

$$\mathbf{D} = \left[\begin{array}{cc} D & O \\ O & R \end{array} \right], \; R = sI + V_{n+1,n+1}. \tag{20}$$

Thus, the preconditioned vector y is obtained in three steps as it is shown in Algorithm 2.

Algorithm 2 Preconditioner update application

```
**Input:** Matrices $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_\mathbf{s}$ and residual vector $r$.
**Output:** Preconditioned vector $y$
1. Solve the linear system $LD\tilde{r} = r$.
2. Update $\tilde{r}$ as
   $\tilde{r} \leftarrow \tilde{r} + \left[ \begin{array}{c} d_1^{-1}V_{n+1,1}^T \\ \vdots \\ d_n^{-1}V_{n+1,n}^T \end{array} \right] R^{-1} \left[ \begin{array}{ccc} \tilde{V}_{n+1,1} & \cdots & \tilde{V}_{n+1,n} \end{array} \right] \tilde{r}.$
3. Solve the linear system $Uy = \tilde{r}$.
```

5 Computational complexity

In this section we study the complexity of the updating method proposed. We observe from equations (19) and (20) that $2k$ additional vectors of size n, and a $k \times k$ matrix are required to store the $\mathbf{L}$, $\mathbf{U}$ and $\mathbf{D}$ factors. Nevertheless, the amount of this extra memory is not large since the vectors are sparse and it is assumed that k is very small relative to the size of the matrix. As a consequence, we note that the computational cost of the preconditioning step is slightly higher compared with a standard preconditioning step with an LU factorization of size n, and depends on the range of the update k. This extra cost correponds to step 2 in Algorithm 2. With computational cost we mean the number of arithmetic operations needed to perform a given computational step. Taking into account both, the computational cost of the preconditioner updating process described in section 3 and its application, it is possible to estimate the reduction on the number of matrix-vector products needed to solve the linear system in order to obtain a performance benefit compared with the option of reusing the initial (non-updated) preconditioner.

Proposition 1. *Let C_0 and C_u be the arithmetic operations needed to solve the updated linear system with the initial and the updated preconditioner, respectively. Then, $C_u < C_0$ if*

$$m_u < m_0 - O(k), \quad \textit{and} \quad k \ll n \ ,$$

where m_0 and m_u are the number of matrix-vector products performed by an iterative Krylov method to solve the linear system with the initial and the updated preconditioners, respectively.

Proof. We recall that the most expensive operation of a Krylov iterative method is a matrix-vector product with the preconditioned system matrix. The number of arithmetic operations needed for this product is denoted by c_0 and c_u for the initial and the updated preconditioners, respectively. Thefore, $C_0 = m_0 c_0$ and $C_u = m_u c_u + C_p$, where C_p is the computational cost of updating the preconditioner described by equations (11), (12) and (13). The nonzero density of the matrix is denoted by d, such that there are dn^2 nonzero elements evenly distributed, and therefore dn nonzero elements on a given row or column. Moreover, it is assumed that dn is negligibly compared with n. Equations (11) implies the computation of $2k$ matrix-vector products that requires $2k(2dn^2-n)$ arithmetic operations. The cost of equations (12) is equivalent to the cost of solving a triangular linear system for each new column of the matrices $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_\mathbf{s}$. Hence this phase requires $2k(dn^2+(d-1)n)$ arithmetic operations. Finally, the computational cost of equation (13) is $k^2(2dn-1)$ arithmetic operations. Adding these costs it follows that $C_p = 6kdn^2 + 2k^2dn + 2kdn - 4kn - k^2$ and therefore, its complexity is $O(kdn^2)$ when n is large and $k \ll n$.

On the other hand, step 2 consists in two matrix-vector products and the inversion of the matrix R. The application of the inverse of R is done by computing its LU factorization, that only is required once and therefore can be considered as a preconditioner computational cost, and the solution of the corresponding triangular systems. Thus, this step requires $2k(2dn-1)+2k^2$ operations and therefore, $c_u = c_0 + 2k(2dn-1) + 2k^2$. A matrix-vector product with the preconditioned matrix with the initial preconditioner requires a matrix-vector product and the solution of two sparse triangular systems since we are working with BIF preconditioners. Then, $c_0 = 4dn^2 + 2dn - 3n$ that has order $O(dn^2)$. Observe that the complexity of C_p is $O(kc_0)$. One has that, $C_u < C_0$ if, and only if

$$m_u < \frac{c_0 m_0 - C_p}{c_u} = \frac{c_0(m_0 - O(k))}{c_0 + 2k(2dn-1) + 2k^2} \ .$$

Clearly, $C_u < C_0$ when

$$m_u < m_0 - O(k)$$

□

This result implies that a reduction of about order k iterations will compensate the extra cost of computing and applying the update, and it will be used in section 7 to compare the performance of the updated preconditioner with the initial one.

6 Approximation properties of the updated preconditioner

In this section we study the approximation properties of the proposed updated preconditioner. It is shown that, under appropriate conditions that can be realistically fulfilled, the spectrum of the preconditioned matrix of the linear system (6) may be clustered around one. Moreover, in the symmetric and positive definite case it is possible to determine its full spectrum.

The proposed strategy relies on improving the performance of the preconditioned iterations for solving the linear system (2) by developing a good preconditioner for the equivalent augmented linear system(6). Solving (2) with a preconditioned Krylov method involves the computation of matrix-vector products with B and an approximation of its inverse operator B^{-1}. We have the following relations between the linear operators B and $\mathbf{B}$,

$$B = A + PQ^T = \begin{bmatrix} I & O \end{bmatrix} \begin{bmatrix} A & P \\ -Q^T & I \end{bmatrix} \begin{bmatrix} I \\ Q^T \end{bmatrix} = \begin{bmatrix} I & O \end{bmatrix} \mathbf{B} \begin{bmatrix} I \\ Q^T \end{bmatrix} \tag{21}$$

and

$$\begin{aligned} B^{-1} = (A + PQ^T)^{-1} &= \begin{bmatrix} I & O \end{bmatrix} \begin{bmatrix} A & P \\ -Q^T & I \end{bmatrix}^{-1} \begin{bmatrix} I \\ O \end{bmatrix} \\ &= \begin{bmatrix} I & O \end{bmatrix} \mathbf{B^{-1}} \begin{bmatrix} I \\ O \end{bmatrix}. \end{aligned} \tag{22}$$

The preconditioning strategy followed in this paper proposes the computation of a BIF preconditioner for the augmented linear operator in (21) that it is used to approximate the inverse linear operator in (22) by direct preconditioning, i.e., solving the corresponding upper and lower triangular systems. Next result relates the condition numbers of the matrices B and $\mathbf{B}$.

Proposition 2. *Let B and $\mathbf{B}$ be the matrices given by equations (1) and (6), respectively. Then,* $\operatorname{cond}(B) \leq \operatorname{cond}(\mathbf{B})\sqrt{1+\sigma_1^2}$, *where* $\sigma_1 = \min\{\sigma_1(Q), \sigma_1(P)\}$ *and $\sigma_1(Q)$, $\sigma_1(P)$ are the largest singular values of Q and P, respectively.*

Proof. One has that,

$$\begin{aligned} \operatorname{cond}(B) = \|B\|_2 \|B^{-1}\|_2 &= \left\| \begin{bmatrix} I & O \end{bmatrix} \mathbf{B} \begin{bmatrix} I \\ Q^T \end{bmatrix} \right\|_2 \left\| \begin{bmatrix} I & O \end{bmatrix} \mathbf{B^{-1}} \begin{bmatrix} I \\ O \end{bmatrix} \right\|_2 \\ &\leq \operatorname{cond}(\mathbf{B}) \left\| \begin{bmatrix} I \\ Q^T \end{bmatrix} \right\|_2 \end{aligned}$$

Since,

$$\left\| \begin{bmatrix} I \\ Q^T \end{bmatrix} \right\|_2^2 = \rho(I + QQ^T) = \lambda_{\max}(I + QQ^T) = 1 + \sigma_1^2(Q),$$

it follows that

$$\text{cond}\,(B) \leq \text{cond}\,(\mathbf{B})\sqrt{1+\sigma_1^2(Q)}.$$

Reasoning similarly with B^T one has, $\text{cond}\,(B) \leq \text{cond}\,(\mathbf{B})\sqrt{1+\sigma_1^2(P)}$. Thus,

$$\text{cond}\,(B) \leq \text{cond}\,(\mathbf{B})\sqrt{1+\sigma_1^2}, \quad \text{where} \quad \sigma_1 = \min\{\sigma_1(Q), \sigma_1(P)\}.$$

□

This proposition suggests that one can expect a faster convergence of the iterative method used to solve the linear system (2) if the condition number of the matrix **B** is improved with a proper preconditioner.

To study the quality of the updated preconditioner, first we evaluate the approximation error norm. A comparison with the non-updated preconditioner is also presented. These preconditioners are given by

$$\mathbf{M} = \mathbf{LDU} = \left[\begin{array}{cc} \hat{A} & \hat{P} \\ -\hat{Q}^T & I \end{array}\right] \quad \text{and} \quad \mathbf{M_0} = \left[\begin{array}{cc} \hat{A} & O \\ O & I \end{array}\right], \tag{23}$$

where $\hat{A} = LDU$ denotes the initial BIF preconditioner for A, the matrices $\hat{P} = L\,[V_{n+1,1}\dots V_{n+1,n}]^T = L\hat{L}^{-1}P$ and $\hat{Q}^T = -[\tilde{V}_{n+1,1}\dots\tilde{V}_{n+1,n}]U = Q^T\hat{U}^{-1}U$ are the approximations of P and Q^T, respectively. Note that $\hat{L}^{-1}P \approx L^{-1}P$ is obtained after dropping elements in the product of $\tilde{Z}^T P$, see equation (11). Similarly, $Q^T\hat{U}^{-1} \approx Q^T U^{-1}$. Since the difference between the error matrices $\mathbf{M}-\mathbf{B}$ and $\mathbf{M_0}-\mathbf{B}$ is located into the offdiagonal blocks, the Frobenius norm is a natural and straightforward choice to study their norm.

Theorem 2. *Let $\hat{A} = LDU$ be an incomplete LDU factorization of A. Let $\mathbf{M}$ and $\mathbf{M_0}$ be the matrices given in (23). Let $\epsilon = \parallel \hat{A}-A \parallel_F^2$, $\delta = \parallel P \parallel_F^2 + \parallel Q \parallel_F^2$ and $c = \max\{\parallel L\hat{L}^{-1}-I \parallel_F^2, \parallel I-\hat{U}^{-1}U \parallel_F^2\}$. Then*

$$\parallel \mathbf{M}-\mathbf{B} \parallel_F^2 \leq \epsilon + c\delta.$$

Moreover, if $c \leq 1$ then

$$\parallel \mathbf{M}-\mathbf{B} \parallel_F \leq \parallel \mathbf{M_0}-\mathbf{B} \parallel_F$$

.

Proof. From (23) we have that

$$\mathbf{M}-\mathbf{B} \;=\; \left[\begin{array}{cc} \hat{A}-A & \hat{P}-P \\ -\hat{Q}^T+Q^T & O \end{array}\right] = \left[\begin{array}{cc} \hat{A}-A & (L\hat{L}^{-1}-I)P \\ Q^T(I-\hat{U}^{-1}U) & O \end{array}\right]$$

Then

$$\begin{array}{rcl} \parallel \mathbf{M}-\mathbf{B} \parallel_F^2 & = & \parallel \hat{A}-A \parallel_F^2 + \parallel (L\hat{L}^{-1}-I)P \parallel_F^2 + \parallel Q^T(I-\hat{U}^{-1}U) \parallel_F^2 \\ & \leq & \parallel \hat{A}-A \parallel_F^2 + c(\parallel P \parallel_F^2 + \parallel Q \parallel_F^2) \\ & \leq & \epsilon + c\delta\ . \end{array}$$

If $c \leq 1$, then

$$\| \mathbf{M} - \mathbf{B} \|_F^2 \leq \| \hat{A} - A \|_F^2 + \| P \|_F^2 + \| Q \|_F^2 = \| \mathbf{M_0} - \mathbf{B} \|_F^2$$

□

As it could be expected, the above theorem shows that the approximation degree of $\mathbf{M}$ depends on $\hat{A}$, $\hat{P}$ and $\hat{Q}$ being a good approximation of A, P and Q, respectively. We note that these conditions depend on the quality of the ISM decomposition of A and also on how accurately the new k rows and columns of $\mathbf{V_s}$ and $\tilde{\mathbf{V}}_\mathbf{s}$ are computed. This accuracy is influenced also by the dropping strategy employed. Moreover, we have proved that if these approximations are good enough, the updated preconditioner $\mathbf{M}$ is closer to the matrix $\mathbf{B}$ than the initial one, $\mathbf{M_0}$.

While Theorem 2 analyses the approximation of $\mathbf{M}$ to $\mathbf{B}$ as a matrix, the next results study the preconditioned matrix itself and its spectral properties.

Theorem 3. *Let the assumptions of Theorem 2 hold. Then the preconditioned matrix* $\mathbf{M^{-1}B}$ *can be written as*

$$\mathbf{M^{-1}B} = \mathbf{I} + \mathbf{M^{-1}E},$$

where

$$\| \mathbf{M^{-1}E} \|_F^2 \leq \| \mathbf{M^{-1}} \|_F^2 \ (\epsilon + c\delta) \tag{24}$$

Proof.

$$\| \mathbf{M^{-1}E} \|_F^2 = \| \mathbf{M^{-1}}(\mathbf{B} - \mathbf{M}) \|_F^2 \leq \| \mathbf{M^{-1}} \|_F^2 \| \mathbf{M} - \mathbf{B} \|_F^2 \leq \| \mathbf{M^{-1}} \|_F^2 \ (\epsilon + c\delta)$$

□

Corollary 1. *Let the assumptions of Theorem 2 hold. Then, there exist* ϵ^* *and* c^* *such that for any* $0 < \epsilon \leq \epsilon^*$ *and* $0 < c \leq c^*$*, the eigenvalues of the preconditioned matrix* $\mathbf{M^{-1}B}$ *are clustered at 1 in the right half complex plane.*

Proof. Consider the bound (24), and let ϵ^* and c^* be a couple of real positive numbers such that for any $0 < \epsilon \leq \epsilon^*$ and $0 < c \leq c^*$

$$\epsilon + c\delta < \frac{1}{\| \mathbf{M^{-1}} \|_F^2}$$

Then $\rho(\mathbf{M^{-1}E}) \leq \| \mathbf{M^{-1}E} \|_F < 1$ and therefore the eigenvalues of the preconditioned matrix are clustered at 1 in the right half complex plane. □

With a clustered spectrum one can expect a faster convergence of an iterative method although we recall that other aspects may influence the behaviour of Krylov-based iterative methods, as for instance the loss of linear independence of the generated basis of non-optimal Krylov subspace methods [29].

Next, we consider the symmetric and positive definite (spd) case. We consider that the matrix of the linear system (2) has the form $B = A + QQ^T$ where A is a symmetric and positive definite matrix. The following result characterizes the spectrum of $\mathbf{M^{-1}B}$.

Theorem 4. *Let* $\mathbf{B}$ *and* $\mathbf{M}$ *be the matrices given by*

$$\mathbf{B} = \begin{bmatrix} A & Q \\ -Q^T & I \end{bmatrix} \quad \textit{and} \quad \mathbf{M} = \begin{bmatrix} \hat{A} & \hat{Q} \\ -\hat{Q}^T & I \end{bmatrix} .$$

Assume that A *is spd,* Q *and* $\hat{Q}$ *have full rank* k*, and the error matrix* $E = Q - \hat{Q}$ *has rank* p*,* $p \leq k$*. Then, the eigenvalues of* $\mathbf{M^{-1}B}$ *are either one or real positive and bounded by*

$$\lambda_{\min}((\hat{A} + \eta I)^{-1}(A + \eta I)) \leq \lambda \leq \lambda_{\max}((\hat{A} + \eta I)^{-1}(A + \eta I)),$$

for some positive η*, or*

$$\lambda_{\min}((\hat{A} + QQ^T)^{-1}(A + QQ^T)) \leq \lambda \leq \lambda_{\max}((\hat{A} + QQ^T)^{-1}(A + QQ^T)),$$

or complex bounded by

$$|\lambda| \leq 1 + \frac{\parallel E \parallel}{\sqrt{1 + \sigma^2_{\min}(\hat{Q})}}$$

where $\sigma_{\min}(\hat{Q})$ *represents the smallest singular value of* $\hat{Q}$*.*

Proof. The technique to prove the result is standard and similar to the one that can be found in [5]. The eigenvalues and eigenvectors of $\mathbf{M^{-1}B}$ are solutions of the following generalized eigenvalue problem

$$\begin{bmatrix} A & Q \\ -Q^T & I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} \hat{A} & Q - E \\ -(Q - E)^T & I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} .$$

One has equivalently that,

$$\begin{array}{rcl} Ax + Qy & = & \lambda \hat{A} x + \lambda Q y - \lambda E y \ , \\ Q^T x & = & \lambda Q^T x - \lambda E^T x + (1 - \lambda) y \ . \end{array} \tag{25}$$

We distinguish the following cases:

1. $x = 0$. From the second equation in (25) it follows that $0 = (1 - \lambda) y$. Then $\lambda = 1$ and therefore $Ey = 0$ from the first equation. Since $y \in \ker E$ that has dimension $k - p$, we obtain that there are $k - p$ eigenvectors $\begin{bmatrix} 0 \\ y \end{bmatrix}$ associated to the unit eigenvalue.

2. $x \neq 0$.

   (a) $Q^T x = 0$. Since Q has rank k it follows that there are $n - k$ linearly independent vectors satisfying this condition. From the second equation we have $\lambda E^T x = (1 - \lambda) y$. By substituting in the first equation and multiplying by x^T one has

$$x^T A x = \lambda x^T \hat{A} x - (1 - \lambda) y^T y \ ,$$

that can be writen as

$$x^T(A+\eta I)x = \lambda x^T(\hat{A}+\eta I)x,$$

where $\eta = \| y \|^2 / \| x \|^2$. Since the matrices $A+\eta I$ and $\hat{A}+\eta I$ are spd, it follows that the eigenvalues are bounded by

$$\lambda_{\min}((\hat{A}+\eta I)^{-1}(A+\eta I)) \leq \lambda \leq \lambda_{\max}((\hat{A}+\eta I)^{-1}(A+\eta I)) \ .$$

(b) $Q^Tx \neq 0$ and $E^Tx = 0$. In this case $k-p$ linearly independent vectors satisfy these conditions. The second equation reduces to

$$(1-\lambda)Q^Tx = (1-\lambda)y$$

and it is satisfied for eigenvalues equal to 1 or when $y = Q^Tx$. In this last case, by substituting in the first equation one has

$$Ax + QQ^Tx = \lambda\hat{A}x + \lambda QQ^Tx - \lambda EQ^Tx \ .$$

Multiplying by x^T we obtain

$$x^T(A+QQ^T)x = \lambda x^T(\hat{A}+QQ^T)x \ .$$

Reasoning as in 2.(a) these eigenvalues are bounded by

$$\lambda_{\min}((\hat{A}+QQ^T)^{-1}(A+QQ^T)) \leq \lambda \leq \lambda_{\max}((\hat{A}+QQ^T)^{-1}(A+QQ^T)) \ .$$

(c) $Q^Tx \neq 0$ and $E^Tx \neq 0$. Multiplying the first equation by x^H and the second by y^H one has

$$\begin{array}{rcl} x^HAx + x^HQy & = & \lambda x^H\hat{A}x + \lambda x^H\hat{Q}y \ , \\ y^HQ^Tx & = & \lambda y^H\hat{Q}^Tx + (1-\lambda)y^Hy \ . \end{array} \tag{26}$$

Adding both equations in (26) we obtain

$$x^HAx + 2Re(x^HQy) - y^Hy = \lambda(x^H\hat{A}x + 2Re(x^H\hat{Q}y) - y^Hy).$$

The eigenvalue λ can be complex if $x^H\hat{A}x + 2Re(x^H\hat{Q}y) - y^Hy = 0$. From the second equation in (25) one has $(Q^Tx - y) = \lambda(\hat{Q}^Tx - y)$. Note that $\hat{Q}^Tx - y \neq 0$ since we are considering $E^Tx \neq 0$. By denoting with $w = \left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right]$ the corresponding eigenvector, it follows that

$$|\lambda| = \frac{\| Q^Tx - y \|}{\| \hat{Q}^Tx - y \|} = \frac{\| \left[\begin{array}{cc} Q^T & -I \end{array} \right] w\|}{\| \left[\begin{array}{cc} \hat{Q}^T & -I \end{array} \right] w \|} \leq 1 + \frac{\| \left[\begin{array}{cc} E^T & O \end{array} \right] w \|}{\| \left[\begin{array}{cc} \hat{Q}^T & -I \end{array} \right] w \|}$$

$$\leq 1 + \frac{\| E \| \| w \|}{\| \left[\begin{array}{cc} \hat{Q}^T & -I \end{array} \right] w \|} \leq 1 + \frac{\| E \|}{\sqrt{1+\sigma^2_{\min}(\hat{Q})}} \ ,$$

where $\sigma_{\min}(\hat{Q})$ is the smallest singular value of $\hat{Q}$.
On the other hand, if $x^H\hat{A}x + 2Re(x^H\hat{Q}y) - y^Hy \neq 0$ then $\lambda \in R$. By subtracting the transpose of the second equation from the first one, we obtain the same equation and the corresponding bound as in 2.(a). Note that $2p$ is the maximum number of complex eigenvalues. □

As in the general case, Theorem 4 indicates that the quality of the updated preconditioner depends on $\hat{A}$ being a good approximation of A, and also the norm of E being as small as possible that may be more important than its rank. Both requirements help to keep both, the real and complex eigenvalues of the preconditioned matrix clustered around the unit. Next example illustrates Theorem 4.

Example 1. We consider the spd linear system $B = A + QQ^T$ where the matrix A corresponds to the 5-point stencil discretization of the 2D Laplacian operator. The columns of the matrix Q are vectors of the canonical basis scaled either by ϵ or $1/\epsilon$, for some real number ϵ. With a suitable value of ϵ we can control the norm of the matrix and also which columns have the largest norm. Therefore, we can generate in a straightforward manner matrices $\hat{Q}$ that produce error matrices with small norm but large rank, or the opposite. For instance, for the following 3−rank matrices

$$Q = \begin{bmatrix} 1/\epsilon & 0 & 0 & 0 & 0 \\ 0 & \epsilon & 0 & 0 & 0 \\ 0 & 0 & \epsilon & 0 & 0 \end{bmatrix}^T, \hat{Q}_1 = \begin{bmatrix} \epsilon & 0 & 0 & 0 & 0 \\ 0 & \epsilon & 0 & 0 & 0 \\ 0 & 0 & \epsilon & 0 & 0 \end{bmatrix}^T \text{ and } \hat{Q}_2 = \begin{bmatrix} 1/\epsilon & 0 & 0 & 0 & 0 \\ 0 & \epsilon^2 & 0 & 0 & 0 \\ 0 & 0 & \epsilon^2 & 0 & 0 \end{bmatrix}^T,$$

their associated error matrices $E_1 = Q - \hat{Q}_1$ and $E_2 = Q - \hat{Q}_2$ have rank 1 and 2, and spectral norm $\dfrac{1}{\epsilon} - \epsilon$ and $\epsilon - \epsilon^2$, respectively.

Table 1 show, for different values of ϵ, the rank (p) and spectral norm of the error matrix E ($\|E\|$), the condition number of the preconditioned augmented matrix $\mathbf{M^{-1}B}$ (κ), the bound for the complex ($\mathbb{C}_b$) and real eigenvalues, ($\mathbb{R}_b$) as predicted by Theorem 4, and also the measured extreme values for the real and complex eigenvalues ($|\lambda_{\mathbb{C}}|_{\max}$, $\lambda_{\mathbb{R}\min}$, $\lambda_{\mathbb{R}\max}$). Column Its. reports the number of iterations spent by the preconditioned conjugate gradient method to solve the updated linear system $(A + QQ^T)x = b$, with b corresponding to the solution of all ones. Two different approximations of the matrix A have been considered. The first block of results correspond to $\hat{A} = A$, and the second one to $\hat{A} = \operatorname{diag}(A)$. With these extreme choices we study the influence of the matrices $\hat{A}$ and $\hat{Q}$ on the spectral distribution of the preconditioned matrix. In addition Figures 2 and 3 show the complete eigenvalues distribution of the matrix $\mathbf{M^{-1}B}$ for $\epsilon = 10^{-2}$ and both approximations of A.

We observe that the distribution of the real eigenvalues depends on the matrix $\hat{A}$ that approximates A. For $\hat{A} = A$ the real eigenvalues are equal to 1 and convergence is achieved very fast for different values of the norm and rank of E. For $\hat{A} = \operatorname{diag} A$ the real eigenvalues are spread in the interval $[0.04, 1.96]$ and the convergence is degraded. Note that these bounds are tight since there are real eigenvalues at both extremes of the interval predicted by Theorem 4. With respect to the complex eigenvalues, its number is determined by the rank of E, that is there are $2p$ complex eigenvalues, while the norm of the error matrix has influence on how these eigenvalues are clustered. Thus, even for larger values of the rank of E, as the norm values become smaller the complex eigenvalues approach one and their number of iterations needed to converge decrease. Also

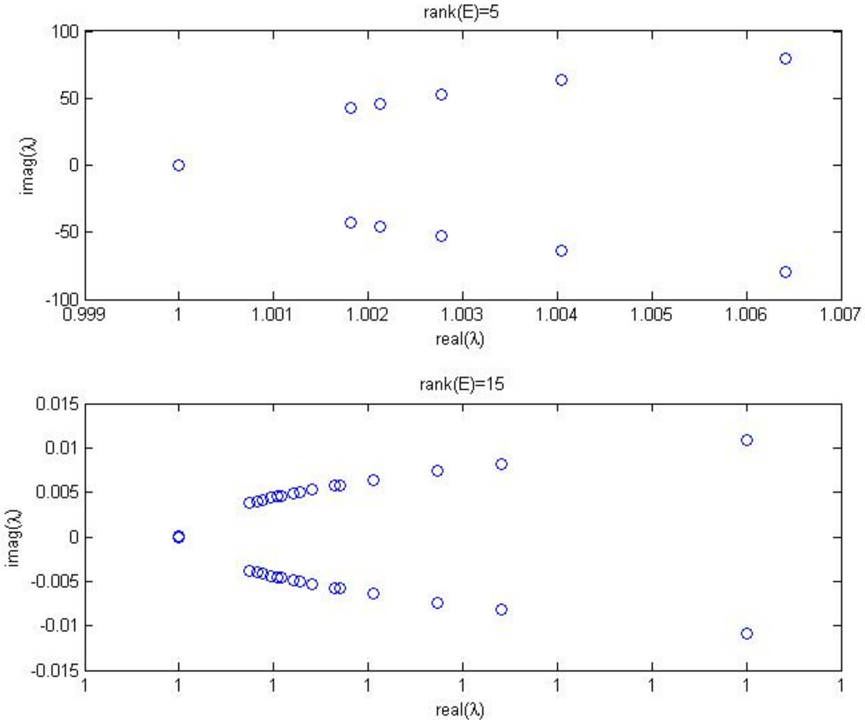


Figure 2: Eigenvalues distribution of the matrix $\mathbf{M^{-1}B}$ for Example 1 with $\epsilon = 10^{-2}$ and $\hat{A} = A$.

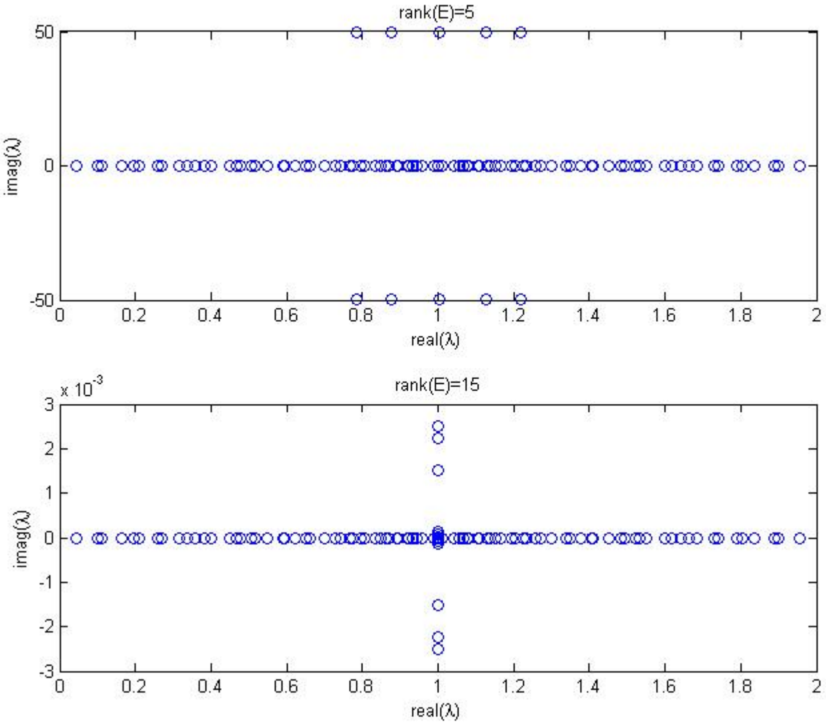


Figure 3: Eigenvalues distribution of the matrix $\mathbf{M^{-1}B}$ for Example 1 with $\epsilon = 10^{-2}$ and $\hat{A} = \mathrm{diag}\,(A)$.

ϵ	p	$\|E\|$	κ	$\mathbb{C}_b$	$\|\lambda_{\mathbb{C}}\|_{\max}$	$\mathbb{R}_b$	$\lambda_{\mathbb{R}\min}/\lambda_{\mathbb{R}\max}$	Its.
10^{-1}	5	10^{1}	200.7	10.99	8.06	[1.0,1.0]	1.0/1.0	6
	15	10^{-1}	1.88	1.09	1.01	[1.0,1.0]	1.0/1.0	4
10^{-2}	5	10^{2}	1799	101.0	80.08	[1.0,1.0]	1.0/1.0	6
	15	10^{-2}	1.071	1.01	1.0001	[1.0,1.0]	1.0/1.0	2
10^{-3}	5	10^{3}	17723	1001.0	800.71	[1.0,1.0]	1.0/1.0	7
	15	10^{-3}	1.001	1.001	1.0	[1.0,1.0]	1.0/1.0	1
10^{-1}	5	10^{1}	393.71	10.99	5.13	[0.04,1.96]	0.04/1.96	53
	15	10^{-1}	69.90	1.09	1.01	[0.04,1.96]	0.04/1.96	40
10^{-2}	5	10^{2}	3371.80	101.0	50.01	[0.04,1.96]	0.04/1.96	54
	15	10^{-2}	66.60	1.01	1.0001	[0.04,1.96]	0.04/1.96	34
10^{-3}	5	10^{3}	33172	1001.0	500.0	[0.04,1.96]	0.04/1.96	43
	15	10^{-3}	66.26	1.001	1.0	[0.04,1.96]	0.04/1.96	24

Table 1: Spectral analysis of the matrix $\mathbf{M^{-1}B}$ for Example 1 according to Theorem 4. Matrix A has size 100 and Q has rank 20. The first block of rows correspond to $\hat{A} = A$ and the second one corresponds to $\hat{A} = \mathrm{diag}(A)$.

note that the bound stated for complex eigenvalues by Theorem 4 is sharper for small norms of the error matrix. Finally, observe that for a given ϵ, i.e., a given norm of the update Q, the condition number of the preconditioned matrix increases with the norm of E.

7 Numerical experiments

In this section we study the numerical performance of the preconditioner update method proposed. We present results obtained with nonsymmetric matrices arising in different areas of scientific computing. Our numerical experiments point out that the proposed algorithm is competitive and robust.

The performance of the updated preconditioner is compared with two different preconditioning strategies. The first one consists in reusing the initial preconditioner for the matrix A, while the second strategy corresponds to the computation of a new BIF preconditioner for the updated matrix $A+PQ^T$ from scratch. Clearly, the former strategy is the one we wanted to avoid and it is included as a benchmark.

The experiments test the performance for different sizes (rank) of the matrices P and Q and different values of the Frobenius norm of the updating matrix PQ^T. Moreover, we considered two different nonzero patterns of PQ^T. For the first one the matrices P and Q were generated such that only nonzero elements of A are modified, i.e., the sparse pattern of A does not change with the update. For the second pattern this restriction was not applied and any entry of the matrix could be updated. We think that these experiments, although are somehow artificial, provide a good insight into the performance of the update method and can be representative for a wide range of scientific applications.

Matrix name	n	nz	cond	Application
1138_BUS	1138	2596	10^7	Power systems admittance matrices
ADD20	2395	17319	10^4	Computer component design
FS_541_4	541	4285	10^{11}	Atmospheric pollution problem
HOR131	434	4710	10^5	Flow network problem
JPWH_991	991	6027	10^3	Circuit physics modelling
NOS7	729	2673	10^9	Diffusion equation
ORSIRR1	1030	6858	10^5	Oil reservoir simulation
ORSREG1	2205	14133	10^4	Oil reservoir simulation
RDB968	968	5632	10^1	Reaction-diffusion Brusselator Model
SHERMAN1	1000	3750	10^4	Oil reservoir simulation
WATHEN(40,20)	2521	38081	10^3	Matlab gallery finite element matrix
MEMPLUS	17758	126150	10^5	Computer component design
CHEM_MASTER1	40401	201201	10^3	Markov model

Table 2: Set of tested matrices

The test matrices are shown in table 2. Except for the matrix WATHEN, all the problems tested belong to the University of Florida Sparse Matrix Collection [15]. For each matrix we provide its dimension n, the number of its nonzero entries nz, its condition number rounded to the nearest power of 10, and the application field.

The ISM shift parameter s was set to 1. The preconditioned BiCGStab method [30], full GMRES and restarted GMRES(m) [28] iterative methods were used for a relative initial residual decrease of 10^{-8}, allowing a maximum number of $2,000$ iterations. Results for full GMRES are included as a reference for GMRES(m) that is commonly used in practice. The right hand side vector was computed as $b = Ae$, where e is the vector of all ones. The initial approximation to the solution x was the vector of all zeros. The experiments where obtained with MATLAB. The BIF algorithm was coded according to Algorithm 3.1 in [12], and the update method as described in Algorithm 1. For the application of the update, in step 2 of Algorithm 2 the inversion of the matrix R was done by computing first its LU factors with the function **lu**(), and then solving the corresponding triangular systems at each iteration. The dropping parameter for computing the BIF preconditioner for A was also used in Algorithm 1 for computing the update. For simplicity, it was set to 0.1 for almost all the tested matrices.

We first conduct a detailed analysis of the results obtained with the smallest matrices tested and the BiCGStab method. The effect of the rank when the updated matrix has the same pattern as A is reported in table 3. This table shows the rank of the update k and the number of iterations needed to converge. The preconditioner density is indicated with ρ^1 and ρ^2. In the first case the size of the preconditioner is relative to the number of nonzero elements of the matrices A, P and Q, and in the second the density is relative to the number of nonzeros of the full matrix $B = A + PQ^T$. Moreover, it is indicated the number α for which $\mathrm{cond}(B) \approx 10^{\alpha}\,\mathrm{cond}(A)$. The updating matrix PQ^T is generated at random in such a way that only the nonzero elements of a subset of k rows of A are modified. This update is done in such a way that the sign of the elements are preserved and its Frobenius norm is equal to the norm of the updated submatrix. Results for the non-preconditioned iterations are also reported.

The first consideration that can be made is that in general the results obtained with the updating method proposed are better than the ones obtained with the initial one for the majority of the tested matrices. Without preconditioning the iterative methods performed poorly and failed to converge in most cases. The reduction observed in the number of iterations tends to be larger with the rank of the update. The exception are the matrices JPWH991, RDB968 and WATHEN(40,10), for which the performance of the initial preconditioner does not degrade with the rank and norm of the update. A possible explanation is that the condition number of these matrices is quite small and does not increase with the rank of the update. Under this situation the initial preconditioner remains competitive and there is not too much room for improvement. Also note that with the updated preconditioner the number of iterations are not far from the results obtained when computing from scratch a new preconditioner and, in some cases, they are even better as will be further explained later. On the other hand, although the size of the preconditioner increases with the rank, we think that the storage requirements are not prohibitive for relative small values of k.

The effect of the norm of the update, for a fixed value of the rank $k = 10$, is illustrated in Figures 4 to 6 for the matrices FS_541_4, JPWH_991 and HOR131. For the rest of the matrices tested a similar behaviour was observed. It can be seen how increasing the norm of the matrix PQ^T the number of iterations of the initial preconditioner deteriorates progressively, while the behaviour of the updated preconditioner remains closer to the recomputed one. Observe that the condition number of the updated matrix tends to increase with the norm and it appears to affect adversely the performance of the initial preconditioner. For larger values of k the effect is more pronounced. In [18] the authors report a similar behaviour for PGMRES in the context of skew-hermitian updates.

Table 4 shows the effect of the rank when the update changes not only the elements but also the sparse pattern of the matrix. In this case, the matrix PQ^T is also generated at random but in such a way that any entry of a subset of k rows of A may be replaced, the modified rows are kept sparse and its norm remains approximately equal. First, as for the previous sparse pattern, it can be observed that the updated preconditioner performs better than the initial one but in this case the benefits seem to be bigger for some matrices. Note that in this case the increment of the rank of the update is paired with a bigger increment of the condition number of the updated matrix. This fact seems to impact more negatively the performance of the initial preconditioner. Moreover, although the performance of the updated preconditioner remains closer to the recomputed one as before, there are some cases where its performance was considerably better. For instance, the results for the matrix NOS7 show that if the increment of the rank leads to a very bad conditioned matrix it can be a source of instabilities.

For this matrix we computed a second denser preconditioner to get a similar iteration count for $k = 25, 50$. Figures 7 and 8 show the sparse pattern of the incomplete LU factors computed for the augmented matrix $\mathbf{B}$ and for the updated matrix $B = A + PQ^T$, respectively. It can be seen that new fill-in in the updated preconditioner is quite moderate and located into the last rows and columns of these factors. By contrast, it appears that the increment on the condition number of the updated matrix makes difficult to compute from scratch a robust preconditioner with a moderate amount of fill-in. Thus, a possible advantage obtained by working with the augmented matrix is that the structure of the preconditioner helps to preserve a good deal of sparsity since new fill-in can occur only in the last k rows and columns. Moreover, because all the new computed entries correspond to the approximation of the rank-k update part of the matrix, a more robust preconditioner may be obtained for a given amount of fill-in. The same can be said for the matrices FS_541_4 and HOR131.

Tables 5 and 6 report the results for the GMRES and GMRES(30) methods for a subset of the matrices tested, respectively. With full GMRES convergence is obtained easily with all the methods and a significant improvement with the updated preconditioner is only attained for the larger values of the rank of the update for which a bigger reduction of the number of iterations is observed. In contrast, there is a considerable improvement for GMRES(30) with the updated preconditioner in number of iterations and time for the largest rank values. It showed to be fairly robust and only failed to converge in 3 occasions.

The updated BIF preconditioner was also compared with ILUT [27] and the results for the BiCGSTAB and GMRES(30) preconditioned methods are shown in Table 7. Results with full GMRES are not reported since similar conclusions to the case mentioned above can be made. We used the Matlab implementation without pivoting and with drop tolerance equal to 0.1. The results listed correspond to ILUT applied to A and also to the updated matrix, namely non-updated and recomputed preconditioners. In general, ILUT performed similarly to BIF and therefore, a similar comparison with the proposed update preconditioner algorithm can be made.

Finally, Tables 8 and 9 show the results for the larger matrices of the test MEMPLUS and CHEM_MASTER1, for the three iterative methods considered preconditioned with BIF and ILUT. The iterative solution CPU time is also reported. Preconditioner computation and setup timings are not included because the initial computations are common to the three strategies considered and in principle can be used for many updates. Moreover, the cost of computing the update is negligible for the rank of the updates and problems tested, especially for these matrices which are the biggest ones of the set. We observe that the results are in line with the analysis of the data for the smaller matrices. The updated BIF preconditioner is almost always better than the non-updated ones, especially for BiCGSTAB and GMRES(30). In addition, it shows a similar performance compared with the recomputed BIF and ILUT preconditioners and in some cases it can be even better, probably due to the reasons explained before. According to Proposition 1 it is observed that the reduction on the number of iterations is large enough to compensate the application of the update and it leads to a better overall performance of the preconditioned iterative method. Of course, there are cases where the additional cost of computing a new preconditioner can be amortized by the reduction on the solution time as, for instance, in the case of CHEM_MASTER1 and preconditioned GMRES(50) with ILUT, but overall we think that the results show that the technique proposed for updating a preconditioner is a good alternative.

8 Conclusions

In this paper we have presented a method for updating an initial BIF preconditioner after a rank-k update of the coefficient matrix. Numerical experiments for a variety of matrices and different iterative methods preconditioned with BIF and ILUT have been obtained. The results show that the strategy proposed is effective and robust. The observed reduction on the number of iterations and solution time for most of the problems tested is enough to compensate the computational cost of the algorithm even for relative small values of k. In general, the performance benefits increase with the rank and also with the norm of the updating matrix. Moreover, its convergence rate is close to the one obtained with a new preconditioner recomputed from scratch. Indeed, it can present even a better performance when the update produces an increment on the condition number of the matrix, probably due to the structure of the preconditioner that helps to confine all the new fill-in induced by the update into a set of $2k$ sparse vectors. This behaviour was observed for different nonzero patterns of the updating matrix.

Acknowledgements

The authors wish to express their thanks to the referees for their helpful suggestions which greatly improved the paper.

References

[1] S. Bellavia, D. Bertaccini, and B Morini. Nonsymmetric preconditioner updates in Newton-Krylov methods for nonlinear systems. *SIAM J. Sci. Comput.*, 33(5):2595–2619, 2011.

[2] M. Benzi and D. Bertaccini. Approximate inverse preconditioning for shifted linear systems. *BIT*, 43(2):231–244, 2003.

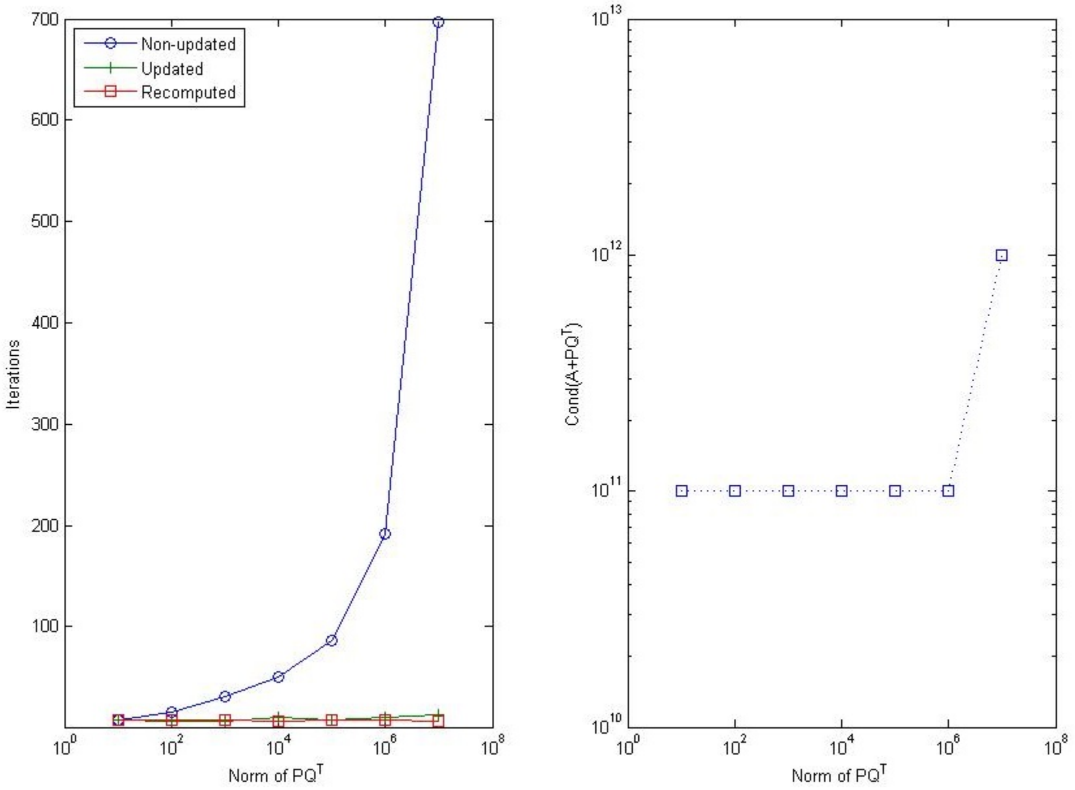


Figure 4: Matrix: FS_541_4, $k = 10$. Effect of the norm of PQ^T on the number of iterations for the three preconditioners (left) and the condition number of the updated matrix (right).

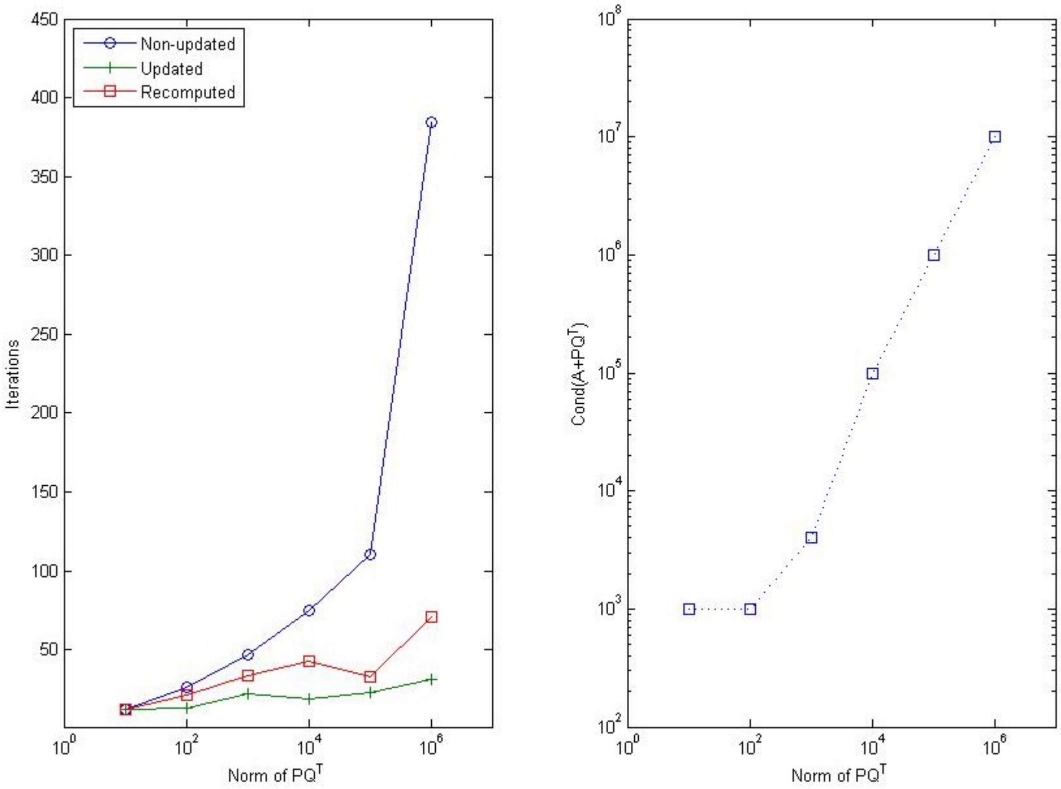


Figure 5: Matrix: JPWH_991, $k = 10$. Effect of the norm of PQ^T on the number of iterations for the three preconditioners (left) and the condition number of the updated matrix (right).

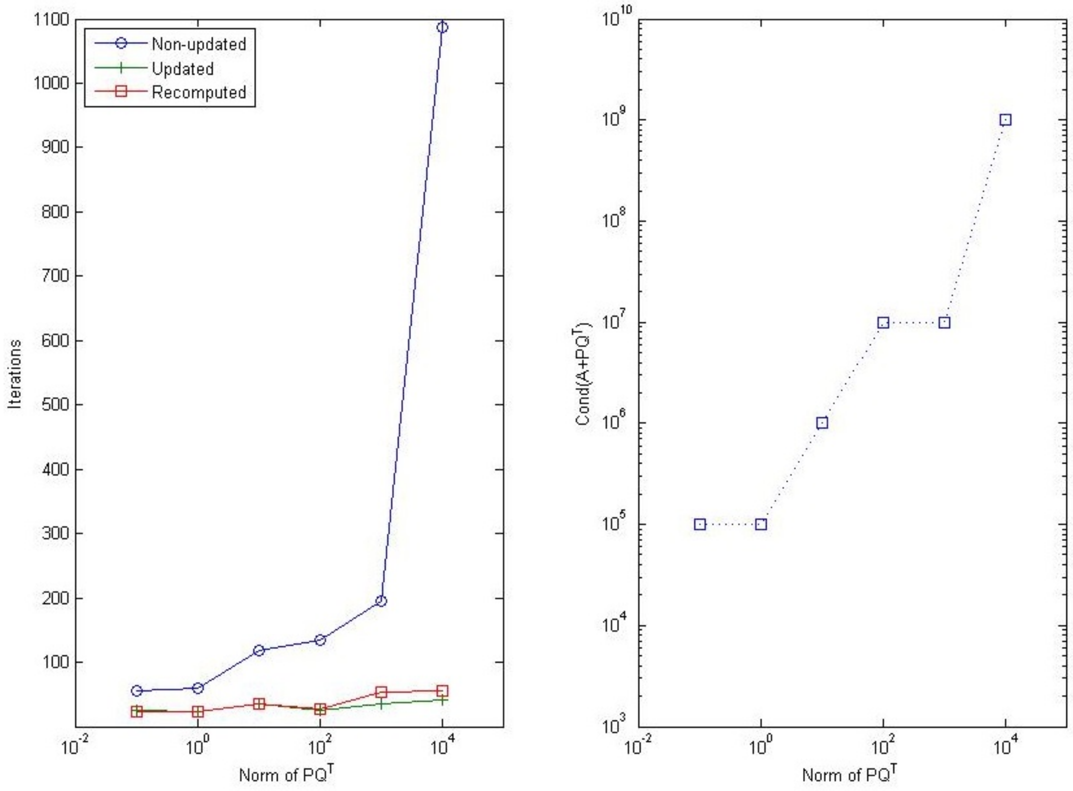


Figure 6: Matrix: HOR_131, $k = 10$. Effect of the norm of PQ^T on the number of iterations for the three preconditioners (left) and the condition number of the updated matrix (right).

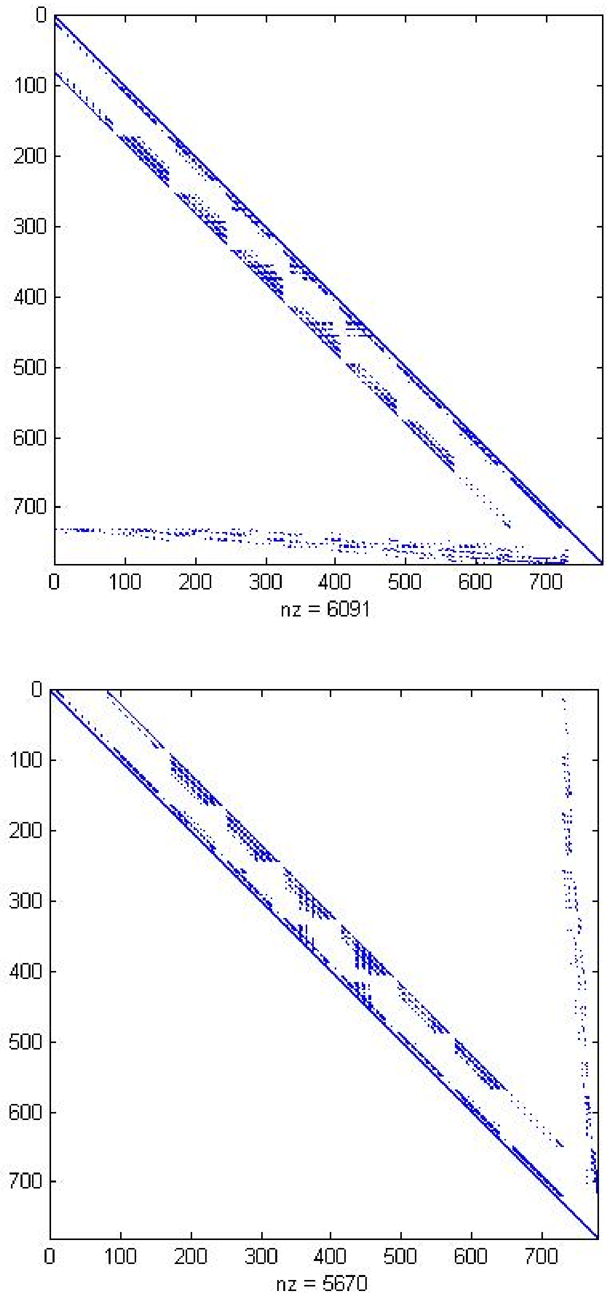
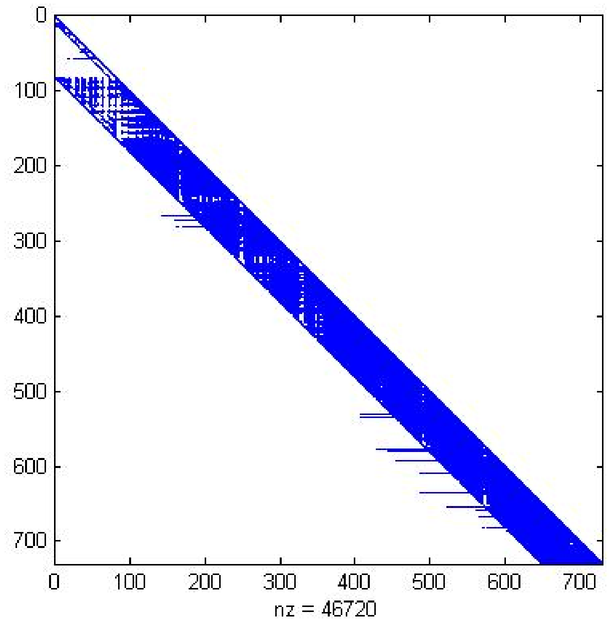


Figure 7: Sparse pattern of the incomplete L and U factors for the augmented matrix $\mathbf{B}$ with the matrix NOS7, $k = 50$



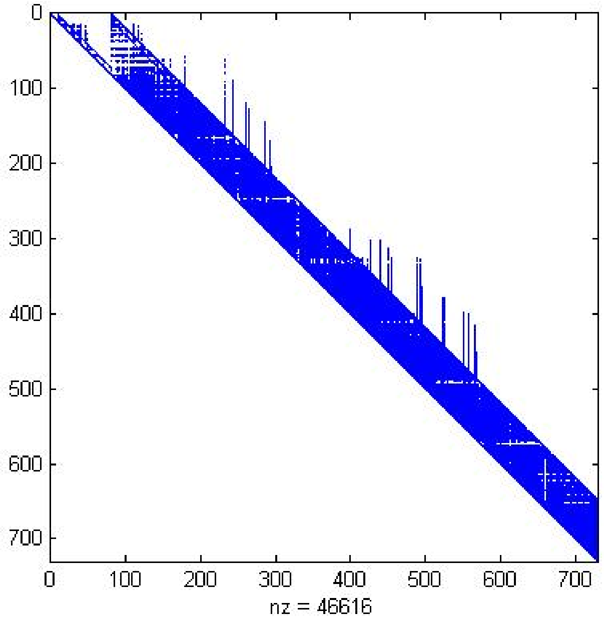


Figure 8: Sparse pattern of the incomplete L and U factors for the matrix $A + PQ^T$. A is the NOS7 matrix, $k = 50$.

[3] L. Bergamaschi, R. Bru, and A Martínez. Low-rank update of preconditioners for the inexact Newton method with SPD Jacobian. *Math. Comput. Model.*, 54:1863–1873, 2011.

[4] L. Bergamaschi, R. Bru, A. Martínez, J. Mas, and M. Putti. Low-rank update of preconditioners for the nonlinear Richards Equation. *Math. Comput. Model.*, 57:1933–1941, 2013.

[5] L. Bergamaschi, J. Gondzio, M. Venturin, and G. Zilli. Inexact constraint preconditioners for linear systems arising in interior point methods. *Comput. Optim. Appl.* , 36(2-3):137–147, 2007.

[6] M. Beroiz, T. Hagstrom, S.R. Lau, and R.H. Price. Multidomain, sparse, spectral-tau method for helically symmetric flow. *Comput. & Fluids*, 102(0):250 – 265, 2014.

[7] D. Bertaccini. Efficient preconditioning for sequences of parametric complex symmetric linear systems. *Electron. Trans. Numer. Anal.*, 18:49–64, 2004.

[8] M. Bollhöfer. A robust and efficient *ILU* that incorporates the growth of the inverse triangular factors. *SIAM J. Sci. Comput.*, 25(1):86–103, 2003.

[9] M. Bollhöfer and Y. Saad. On the relations between ILUs and factored approximate inverses. *SIAM J. Matrix Anal. Appl.*, 24(1):219–237, 2002.

[10] R. Bru, J. Cerdán, J. Marín, and J. Mas. Preconditioning sparse nonsymmetric linear systems with the Sherman-Morrison formula. *SIAM J. Sci. Comput.*, 25(2):701–715, 2003.

[11] R. Bru, J. Marín, J. Mas, and M. Tůma. Balanced incomplete factorization. *SIAM J. Sci. Comput.*, 30(5):2302–2318, 2008.

[12] R. Bru, J. Marín, J. Mas, and M. Tůma. Improved balanced incomplete factorization. *SIAM J. Matrix Anal. Appl.*, 31(5):2431–2452, 2010.

[13] J. Cerdán, T. Faraj, N. Malla, J. Marín, and J. Mas. Block approximate inverse preconditioners for sparse nonsymmetric linear systems. *Electron. Trans. Numer. Anal.*, 37:23–40, 2010.

[14] J. Cerdán, J. Marín, J Mas, and M. Tůma. Block balanced incomplete factorization. Technical Report No. TR-IMM2015/04, Polytechnic University of Valencia, Spain, 2015.

[15] T. A. Davis. *University of Florida Sparse Matrix Collection.* available online at http://www.cise.ufl.edu/∼davis/sparse/, NA Digest, vol. 94, issue 42, October 1994.

[16] J. D. Tebbens and M. Tůma. Efficient preconditioning of sequences of nonsymmetric linear systems. *SIAM J. Sci. Comput.*, 29(5):1918–1941, 2007.

[17] J. D. Tebbens and M. Tůma. Preconditioner updates for solving sequences of linear systems in matrix-free environment. *Numer. Linear Algebra Appl.*, 17:997–1019, 2010.

[18] M. Embree, J. A. Sifuentes, K. M. Soodhalter, D. B. Szyld, and F. Xue. Short-term recurrence Krylov subspace methods for nearly hermitian matrices. *SIAM. J. Matrix Anal. and Appl.*, 33-2:480–500, 2012.

[19] B. Engquist and L. Ying. Sweeping preconditioner for the Helmholtz equation: hierarchical matrix representation. *Commun. Pure Appl. Math.*, 64:697–735, 2011.

[20] P. Gatto, R. E. Christiansen, and J. S. Hesthaven. A preconditioner based on a low-rank approximation with applications to topology optimization. Technical Report EPFL-ARTICLE-207108, École polytechnique fédérale de Lausanne, EPFL, CH-1015 Lausanne, 2015.

[21] L. Grasedyck and W. Hackbusch. Construction and arithmetics of H-matrices. *Computing*, 70(4):295–334, 2003.

[22] L. Grasedyck, D. Kressner, and C. Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, 36(1):53–78, 2013.

[23] L. Greengard and V. Rokhlin. A new version of the fast multipole method for the Laplace equation in three dimensions. *Acta Numer.*, 6:229–269, 1 1997.

[24] W. W. Hager. Updating the inverse of matrix. *SIAM Rev.*, 31(2):221–239, 1989.

[25] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, May 2011.

[26] C. T. Kelley. *Solving nonlinear equations with Newton's method*. Fundamentals of Algorithms. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2003.

[27] Y. Saad. ILUT: a dual threshold incomplete *LU* factorization. *Numer. Linear Algebra Appl.*, 1(4):387–402, 1994.

[28] Y. Saad and M. H. Schulz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.

[29] V. Simoncini and D. B.Szyld. The effect of non-optimal bases on the convergence of Krylov subspace methods. *Numer. Math.*, 100(4):711–733, 2005.

[30] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 12:631–644, 1992.

Matrix	k	α	non-prec	non-updated	updated	recomputed
			Its.	Its./ρ^1	Its./ρ^1	Its./ρ^2
1138_BUS	1	0	†	69/1.51	60/1.51	60/1.51
	5	0	†	89/1.51	70/1.51	70/1.52
	10	1	†	193/1.49	101/1.52	105/1.53
	25	1	1582	460/1.47	115/1.55	128/1.54
	50	2	†	1006/1.43	153/1.58	148/1.60
ADD20	1	0	366	9/0.71	8/0.71	8/0.71
	5	1	398	14/0.71	8/0.71	8/0.71
	10	1	565	31/0.71	9/0.72	24/0.72
	25	1	†	54/0.70	10/0.72	10/0.74
	50	1	†	238/0.70	25/0.74	39/0.72
FS_541_4	1	0	1148	13/0.83	8/0.83	8/0.83
	5	0	1044	30/0.83	8/0.83	8/0.84
	10	0	1348	77/0.81	13/0.86	17/0.88
	25	0	†	203/0.79	21/0.91	24/0.93
	50	1	†	613/0.78	25/0.97	21/1.00
HOR131	1	0	†	20/1.08	19/1.09	19/1.09
	5	0	†	30/1.07	23/1.14	25/1.16
	10	1	†	58/1.05	43/1.17	31/1.26
	25	1	†	110/1.01	40/1.42	35/1.47
	50	2	†	281/0.96	123/1.67	357/2.75
JPWH991	1	0	34	13/1.18	13/1.18	13/1.18
	5	0	35	13/1.17	12/1.18	12/1.18
	10	0	37	13/1.17	12/1.18	12/1.18
	25	0	42	15/1.15	13/1.18	12/1.18
	50	0	36	14/1.12	12/1.19	14/1.12
NOS7	1	1	†	18/0.98	21/0.98	25/0.98
	5	1	†	38/0.97	29/0.98	21/0.98
	10	1	†	49/0.96	31/0.99	29/1.13
	25	1	1994	1545/0.94	27/1.06	31/1.14
	50	1	†	†/0.91	66/1.15	76/1.21
ORSREG1	1	0	368	36/0.58	30/0.58	33/0.58
	5	0	501	86/0.58	50/0.58	99/0.59
	10	0	740	133/0.58	76/0.58	94/0.59
	25	1	†	427/0.57	263/0.59	246/0.61
	50	2	†	1446/0.57	544/0.59	625/0.62
ORSIRR1	1	0	1170	36/0.56	35/0.56	30/0.56
	5	1	1652	66/0.56	39/0.56	65/0.57
	10	0	†	136/0.55	53/0.57	50/0.59
	25	1	†	316/0.54	105/0.57	118/0.60
	50	1	†	1025/0.53	269/0.59	285/0.69
RDB968	1	0	†	6/2.27	5/2.27	5/2.27
	5	0	†	6/2.26	6/2.27	5/2.28
	10	0	†	7/2.24	6/2.26	5/2.28
	25	0	†	7/2.21	6/2.25	6/2.28
	50	0	†	6/2.14	6/2.24	6/2.29
SHERMAN1	1	0	337	25/1.13	23/1.14	25/1.13
	5	0	504	50/1.13	37/1.14	34/1.14
	10	0	528	43/1.12	28/1.16	38/1.16
	25	3	1840	73/1.10	52/1.20	47/1.22
	50	3	1226	123/1.06	49/1.23	48/1.22
WATHEN	1	0	84	9/0.94	9/0.94	7/0.91
	5	0	88	8/0.94	8/0.95	7/0.96
	10	0	86	8/0.94	8/0.94	7/0.91
	25	0	98	7/0.95	8/0.96	7/0.96
	50	0	103	7/0.95	8/0.96	7/0.96

Table 3: Effect of the rank of the update when the matrices A and $A + PQ^T$ have the same sparse pattern. Solution obtained with BiCGStab.

Matrix	k	α	non-prec	non-updated	updated	recomputed
			Its.	Its./ρ^1	Its./ρ^1	Its./ρ^2
1138_BUS	1	0	†	54/1.51	54/1.51	54/1.51
	5	1	†	60/1.50	54/1.51	53/1.52
	10	1	1345	120/1.49	100/1.51	112/1.66
	25	1	†	129/1.47	100/1.56	110/1.71
	50	1	†	332/1.43	201/1.58	291/1.90
ADD20	1	1	447	10/0.71	7/0.71	7/0.71
	5	1	402	9/0.70	8/0.72	8/0.72
	10	1	434	9/0.70	8/0.72	9/0.71
	25	2	561	13/0.70	9/0.73	13/0.72
	50	2	1910	113/0.69	24/0.78	48/0.73
FS_541_4	1	1	808	8/0.83	8/0.83	7/1.21
	5	2	†	24/0.82	14/0.84	13/1.56
	10	3	†	35/0.81	14/0.85	11/1.62
	25	5	†	400/0.79	147/0.91	1755/2.70
						221/13.15
	50	9	†	†/0.75	113/1.51	†/6.75
						731/17.23
HOR131	1	0	†	24/1.07	22/1.08	23/1.15
	5	0	1727	33/1.06	25/1.10	32/1.10
	10	1	†	43/1.05	24/1.18	23/1.64
	25	2	862	101/1.01	44/1.35	71/1.89
	50	3	†	116/0.96	39/1.71	410/2.80
						82/7.08
JPWH991	1	0	36	13/1.18	13/1.18	13/1.18
	5	0	41	12/1.18	12/1.18	12/1.18
	10	0	43	13/1.17	13/1.18	13/1.18
	25	0	40	12/1.15	12/1.15	12/1.17
	50	0	43	13/1.11	13/1.16	12/1.14
NOS7	1	0	913	14/1.26	15/1.26	15/1.26
	5	1	1684	20/1.25	22/1.27	39/1.63
	10	4	1922	40/1.24	32/1.27	53/1.63
	25	6	†	564/1.21	96/1.36	†/7.83
				504/2.18	44/2.69	203/12.42
	50	7	†	†/2.09	65/2.81	†/11.74
	50			1960/4.31	47/5.28	588/19.56
ORSREG1	1	2	406	48/0.58	48/0.58	48/0.58
	5	2	983	75/0.58	63/0.58	65/0.59
	10	2	†	128/0.58	97/0.58	91/0.60
	25	3	†	354/0.57	240/0.59	235/0.61
	50	3	†	1023/0.57	535/0.59	561/0.62
ORSIRR1	1	2	1827	40/0.56	37/0.56	37/0.56
	5	2	†	55/0.56	46/0.56	44/0.57
	10	2	†	186/0.55	104/0.56	110/0.55
	25	3	†	322/0.54	154/0.58	144/0.66
	50	3	†	653/0.53	325/0.59	320/0.74
RDB968	1	0	†	6/2.27	5/2.27	5/2.27
	5	0	†	6/2.26	6/2.27	5/2.28
	10	0	†	7/2.24	6/2.26	5/2.28
	25	0	†	7/2.21	6/2.25	6/2.28
	50	0	†	6/2.14	6/2.24	6/2.29
SHERMAN1	1	1	350	24/1.13	24/1.14	23/1.13
	5	1	401	24/1.13	27/1.15	27/1.16
	10	1	438	26/1.13	28/1.15	28/1.16
	25	1	730	42/1.10	33/1.18	30/1.17
	50	2	614	50/1.07	35/1.25	36/1.26
WATHEN	1	0	74	7/0.96	7/0.95	7/0.96
	5	0	76	7/0.96	7/0.95	7/0.96
	10	0	68	7/0.95	7/0.94	7/0.95
	25	0	71	7/0.95	7/0.95	7/0.94
	50	0	73	7/0.95	7/0.96	7/0.94

Table 4: Effect of the rank of the update when the matrices A and $A + PQ^T$ have a different sparse pattern. Solution obtained with BiCGStab.

Matrix	k	α	non-prec	non-updated	updated	recomputed
			Its.	Its./ρ^1	Its./ρ^1	Its./ρ^2
1138_BUS	1	0	481	74/1.51	73/1.51	72/1.51
	5	0	517	71/1.51	72/1.51	74/1.51
	10	0	535	83/1.49	75/1.52	79/1.53
	25	0	558	87/1.47	71/1.55	72/1.54
	50	0	519	102/1.43	75/1.58	79/1.60
ADD20	1	0	301	16/0.71	16/0.71	16/0.71
	5	1	313	18/0.71	16/0.71	16/0.71
	10	1	327	21/0.71	17/0.72	16/0.72
	25	1	420	33/0.70	18/0.72	18/0.74
	50	0	618	56/0.70	23/0.74	30/0.72
FS_541_4	1	0	217	15/0.83	15/0.83	15/0.83
	5	0	233	20/0.82	17/0.83	19/0.84
	10	0	250	28/0.81	21/0.86	23/0.88
	25	0	252	36/0.79	24/0.91	24/0.93
	50	1	278	60/0.78	30/0.97	25/1.00
HOR_131	1	0	421	32/1.08	32/1.09	32/1.09
	5	0	420	32/1.07	26/1.14	31/1.14
	10	1	424	38/1.05	35/1.17	36/1.28
	25	1	425	49/1.01	41/1.42	37/1.47
	50	2	432	67/0.95	53/1.67	74/2.75
NOS7	1	1	351	24/0.98	24/0.98	24/0.98
	5	1	401	22/0.97	24/0.98	24/0.98
	10	1	390	24/0.96	24/0.99	24/1.13
	25	1	477	38/0.94	29/1.06	31/1.34
	50	1	415	59/0.91	49/1.15	56/2.02
ORSREG1	1	0	285	48/0.58	48/0.58	57/0.58
	5	0	363	64/0.58	59/0.58	57/0.59
	10	0	393	74/0.58	60/0.58	50/0.59
	25	1	625	109/0.57	97/0.59	91/0.61
	50	2	922	152/0.57	124/0.59	118/0.62
ORSIRR1	1	0	437	46/0.56	45/0.56	51/0.56
	5	1	470	49/0.56	47/0.56	51/0.57
	10	0	503	56/0.55	51/0.57	51/0.59
	25	1	658	83/0.54	69/0.57	64/0.60
	50	1	823	120/0.53	100/0.59	90/0.69
SHERMAN1	1	0	324	35/1.13	35/1.14	35/1.13
	5	0	372	44/1.13	39/1.14	41/1.14
	10	0	356	40/1.12	35/1.16	36/1.16
	25	3	387	53/1.10	42/1.20	54/1.22
	50	3	419	63/1.06	46/1.23	49/1.22

Table 5: Effect of the rank of the update when the matrices A and $A+PQ^T$ have the same sparse pattern. Solution obtained with full GMRES. The density of the preconditioners and the condition number of the updated matrices (parameter α) are the same which are listed in table 3.

Matrix	k	α	non-prec	non-updated	updated	recomputed
			Its.	Its./ρ^1	Its./ρ^1	Its./ρ^2
1138_BUS	1	0	†	397/1.51	384/1.51	382/1.51
	5	0	†	201/1.51	234/1.51	205/1.51
	10	0	†	1110/1.49	240/1.52	335/1.53
	25	0	†	†/1.47	326/1.55	320/1.54
	50	0	†	†/1.43	319/1.58	1399/1.60
ADD20	1	0	301	16/0.71	16/0.71	16/0.71
	5	1	313	48/0.71	46/0.71	46/0.71
	10	1	†	51/0.71	46/0.72	47/0.72
	25	1	†	80/0.70	48/0.72	48/0.74
	50	0	†	988/0.69	53/0.74	60/0.72
FS_541_4	1	0	†	15/0.83	15/0.83	15/0.83
	5	0	†	20/0.82	17/0.83	19/0.84
	10	0	†	28/0.81	19/0.86	23/0.88
	25	0	†	232/0.79	24/0.91	24/0.93
	50	0	†	†/0.78	30/0.96	25/1.00
HOR_131	1	0	†	66/1.08	66/1.09	66/1.09
	5	0	†	71/1.07	59/1.14	66/1.30
	10	1	†	81/1.07	78/1.17	77/1.28
	25	1	†	180/1.01	100/1.42	85/1.47
	50	2	†	†/0.95	980/1.67	†/2.75
NOS7	1	1	†	53/0.98	54/0.98	54/0.98
	5	1	†	52/0.97	53/0.98	54/0.98
	10	1	†	54/0.96	54/0.99	54/1.13
	25	1	†	†/0.94	57/1.06	67/1.34
	50	1	†	†/0.91	59/1.15	†/2.02
ORSREG1	1	0	748	81/0.58	80/0.58	88/0.58
	5	0	†	116/0.58	111/0.58	116/0.59
	10	0	†	169/0.58	104/0.58	89/0.59
	25	1	†	†/0.57	†/0.59	†/0.61
	50	2	†	†/0.57	†/0.59	†/0.62
ORSIRR1	1	0	†	79/0.56	78/0.56	84/0.56
	5	1	†	99/0.56	98/0.56	95/0.57
	10	0	†	141/0.55	101/0.57	8/0.59
	25	1	†	†/0.54	323/0.57	268/0.60
	50	1	†	†/0.53	†/0.59	†/0.69
SHERMAN1	1	0	†	70/1.13	70/1.14	67/1.13
	5	0	†	115/1.13	90/1.14	90/1.14
	10	0	†	88/1.12	72/1.16	81/1.16
	25	3	†	226/1.10	1181.22	299/1.22
	50	3	†	474/1.06	111/1.23	110/1.22

Table 6: Effect of the rank of the update when the matrices A and $A + PQ^T$ have the same sparse pattern. Solution obtained with GMRES(30). The density of the preconditioners and the condition number of the updated matrices (parameter α) are the same which are listed in table 3.

			BiCGSTAB		GMRES(30)	
Matrix	k	$[\rho_{\min}/\rho_{\max}]$	non-updated	recomputed	non-updated	recomputed
			Its.	Its.	Its.	Its.
1138_BUS	1	$[1.42, 1.46]$	56	66	270	272
	5		98	113	232	238
	10		168	82	567	176
	25		405	100	†	310
	50		1018	144	†	326
ADD20	1	$[0.64,\ 0.66]$	36	30	47	52
	5		44	37	83	80
	10		76	40	90	84
	25		213	45	207	93
	50		908	157	†	1015
FS_541_4	1	$[0.62, 0.79]$	10	6	11	10
	5		26	9	17	12
	10		63	12	22	15
	25		185	9	120	14
	50		500	13	†	20
HOR_131	1	$[0.97, 2.01]$	28	25	89	89
	5		51	36	104	86
	10		69	32	101	80
	25		124	36	239	89
	50		300	102	†	833
NOS7	1	$[0.92, 1.00]$	19	21	55	55
	5		43	22	53	56
	10		43	23	56	58
	25		1686	53	†	150
	50		†	129	†	†
ORSREG1	1	$[0.55, 0.57]$	36	36	82	91
	5		87	93	119	121
	10		147	80	173	119
	25		431	316	†	†
	50		1636	616	†	†
ORSIRR1	1	$[0.54, 0.59]$	37	37	81	86
	5		68	42	104	92
	10		147	57	143	105
	25		369	123	†	778
	50		1182	519	†	†
SHERMAN1	1	$[1.02, 1.07]$	26	22	76	77
	5		54	37	122	101
	10		45	29	100	85
	25		83	53	190	119
	50		126	73	447	159

Table 7: Effect of the rank of the update when the matrices A and $A + PQ^T$ have the same sparse pattern. Solution obtained with ILUT as preconditioner.

			BIF			ILUT	
It. Method	k	non-prec	non-updated	updated	recomputed	non-updated	recomputed
			Its./Time	Its./Time	Its./Time	Its./Time	Its./Time
BiCGSTAB	5	1882/3.25	305/0.89	163/0.52	218/0.64	320/0.92	227/0.71
	10	†	700/1.90	139/0.42	156/0.46	662/1.87	166/0.47
	25	†	†	460/1.42	445/1.34	†	484/1.40
	50	†	†	453/1.49	443/1.45	†	412/1.35
GMRES	5	730 / 80.28	146/3.57	151/3.98	171/4.88	146/3.51	166/4.65
	10	779 / 87.02	139/3.17	165/4.25	177/4.85	142/3.25	174/4.55
	25	1085/168.16	166/4.32	166/4.17	214/6.92	162/4.16	208/6.05
	50	1183/219.46	217/6.99	209/6.07	223/6.55	213/6.88	221/6.45
GMRES(30)	5	†	727/4.30	746/4.31	907/5.65	727/4.22	918/5.72
	10	†	465/2.75	378/2.10	439/2.37	455/2.52	427/2.31
	25	†	†	†	†	†	†
	50	†	†	†	†	†	†

Table 8: Effect of the rank of the update when the matrices A and $A + PQ^T$ have different sparse pattern for the matrix MEMPLUS. The relative density for all the preconditioners is in the range $[0.69, 0.73]$.

			BIF			ILUT	
It. Method	k	non-prec	non-updated	updated	recomputed	non-updated	recomputed
			Its./Time	Its./Time	Its./Time	Its./Time	Its./Time
BiCGSTAB	5	536/2.10	160/1.15	135/0.97	151/1.06	125/0.80	123/0.80
	10	†	224/1.56	186/1.29	182/1.29	218/1.51	216/1.44
	25	†	513/2.95	289/1.81	369/2.34	347/2.20	298/1.79
	50	†	774/5.38	579/3.69	619/3.83	603/3.86	499/2.93
GMRES	5	789/207.5	194/13.34	176/10.29	199/13.84	186/12.24	185/11.87
	10	832/231.2	204/14.41	195/13.08	204/14.42	192/12.61	185/11.97
	25	821/230.1	226/15.81	200/13.15	226/15.74	216/14.54	193/12.14
	50	948/300.0	255/20.61	217/15.08	248/19.35	246/18.98	197/13.36
GMRES(50)	5	†	794/14.39	583/9.89	741/13.31	650/11.49	554/9.79
	10	†	1282/23.51	1091/19.90	1350/25.08	1045/18.66	900/16.46
	25	†	†	1446/24.25	†	†	1353/23.17
	50	†	†	†	†	†	†

Table 9: Effect of the rank of the update when the matrices A and $A + PQ^T$ have the same sparse pattern for the matrix CHEM_MASTER1. The relative density for all the preconditioners is around 1.14 for BIF and 1.05 for ILUT.