# EFFICIENT PRECONDITIONER UPDATES FOR SHIFTED LINEAR SYSTEMS*

STEFANIA BELLAVIA†, VALENTINA DE SIMONE‡, DANIELA DI SERAFINO‡§, AND BENEDETTA MORINI†

**Abstract.** We present a technique for building effective and low cost preconditioners for sequences of shifted linear systems $(A+\alpha I)x_\alpha = b$, where $A$ is symmetric positive definite and $\alpha > 0$. This technique updates a preconditioner for $A$, available in the form of an $LDL^T$ factorization, by modifying only the nonzero entries of the $L$ factor in such a way that the resulting preconditioner mimics the diagonal of the shifted matrix and reproduces its overall behaviour. This approach is supported by a theoretical analysis as well as by numerical experiments, showing that it works efficiently for a broad range of values of $\alpha$.

**Key words.** Shifted linear systems, preconditioner updates, incomplete $LDL^T$ factorization.

**AMS subject classifications.** Primary: 65F08, 65F50. Secondary: 65K05, 65M22.

**1. Introduction.** We are concerned with the problem of building efficient preconditioners for the solution, by Krylov methods, of shifted linear systems of the form

$$(A+\alpha I)x_\alpha = b, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix, $I \in \mathbb{R}^{n\times n}$ is the identity matrix and $\alpha > 0$. Sequences of such linear systems arise in various fields, e.g. in trust-region and regularization techniques for nonlinear least-squares and other optimization problems, as well as in the application of implicit methods for the numerical solution of partial differential equations (PDEs).

The conditioning of the matrix $A+\alpha I$ may considerably change as $\alpha$ varies; thus, reusing the same preconditioner for different matrices is likely to work well for small values of $\alpha$, but may be inappropriate as $\alpha$ increases. On the other hand, recomputing the preconditioner from scratch for each shifted matrix may be too expensive, either when $\alpha$ is small or when it is large and the spectral properties of $A+\alpha I$ are more favourable.

In this paper we are interested in building efficient and low-cost preconditioners for the shifted matrices, through a suitable update of a seed preconditioner, i.e. of a preconditioner built for the matrix $A$. Specifically, we assume that the seed preconditioner is available in the form of an $LDL^T$ factorization, with $L$ unit lower triangular and $D$ diagonal positive definite. We note that in case the matrix $A$ is not explicitly available, but can be accessed by evaluating matrix-vector products, it is possible

*Work supported in part by INdAM-GNCS, under grant *Progetti 2010 - Analisi e risoluzione iterativa di sistemi lineari di grandi dimensioni in problemi di ottimizzazione*, and by MIUR, under PRIN grants no. 20079PLLN7 - *Nonlinear Optimization, Variational Inequalities and Equilibrium Problems*, and no. 20074TZWNJ - *Development and analysis of mathematical models and of numerical methods for partial differential equations for applications to environmental and industrial problems.*

†Dipartimento di Energetica "S. Stecco", Università degli Studi di Firenze, via C. Lombroso 6/17, 50134 Firenze, Italy, (stefania.bellavia@unifi.it, benedetta.morini@unifi.it).

‡Dipartimento di Matematica, Seconda Università degli Studi di Napoli, via A. Vivaldi 43, 81100 Caserta, Italy (valentina.desimone@unina2.it, daniela.diserafino@unina2.it).

§Istituto di Calcolo e Reti ad Alte Prestazioni, CNR, via P. Castellino 111, 80131 Napoli, Italy.

to obtain an incomplete $LDL^T$ factorization of $A$ through matrix-free procedures, without explicitly forming $A$ [8, 21].

Techniques for updating factorized preconditioners were proposed in [3, 6, 9, 10, 16, 17, 25]. More precisely, sequences of systems of the form (1.1) are specifically addressed in [6, 25], while complex matrices which differ by a diagonal matrix are considered in [9]; finally, the algorithms presented in [3, 10, 16, 17] apply to sequences of general nonsymmetric matrices. These techniques are based on the availability of an incomplete factorization of either $A$ or $A^{-1}$.

The procedures in [10] have been designed to improve the $L$ and $U$ factors of an incomplete $LU$ factorization of a general matrix $A$, and to build preconditioners for a sequence of slowly varying matrices. These procedures are based on the idea of alternately correcting already computed $L$ and $U$ matrices, through sparse approximate inverse techniques or sparse-sparse triangular system solves. However, the development of alternating correction techniques for symmetric matrices is not considered.

The algorithms in [3, 6, 9, 16, 17] share a common ground, since they update a factorized preconditioner $P = LDL^T$, or $P^{-1} = L^{-T}D^{-1}L^{-1}$, by leaving the unit lower triangular factor $L$ unchanged and modifying the factor $D$. The updating technique in [3, 6, 9] exploits the stabilized AINV preconditioner [5], which provides an approximation of $A^{-1}$:

$$A^{-1} \approx VD^{-1}V^T = L^{-T}D^{-1}L^{-1}.$$

It follows that

$$(A + \alpha I)^{-1} \approx V(D + \alpha V^T V)^{-1}V^T,$$

and a preconditioner for $A + \alpha I$ can be obtained as

$$(\overline{P}_\alpha)^{-1} = V(D + \alpha H)^{-1}V^T, \tag{1.2}$$

where $H$ is a symmetric approximation of $V^TV$. This can be formed by extracting an upper banded matrix $\tilde{V}$ from $V$ and setting $H = \tilde{V}^T\tilde{V}$, or simply taking the main diagonal of $V^TV$. In the latter case $D + \alpha H$ is diagonal and the updated preconditioner can be trivially applied; in the former, the solution of a banded system is required. This updating strategy is supported by a theoretical justification which ensures good spectral properties of the preconditioned matrices if the entries of $V$ decay fast enough from the main diagonal [6].

The approach presented in [16, 17] has been developed for the general case of preconditioning a sequence of nonsymmetric matrices which differ by a non-diagonal matrix, but we specialize it to the problems of interest in this paper. From

$$A \approx LDL^T,$$

it follows that

$$A + \alpha I \approx L(D + \alpha L^{-1}L^{-T})L^T. \tag{1.3}$$

Therefore, a preconditioner $\hat{P}_\alpha$ for $A + \alpha I$ can be defined as

$$\hat{P}_\alpha = LKL^T,$$

where $K$ is an easily invertible approximation to $D + \alpha L^{-1}L^{-T}$ that preserves the symmetry. Obtaining such an approximation may be quite expensive as it involves the computation of entries of $L^{-1}$. In the general case of a nonsymmetric matrix for which an incomplete factorization $LDU$ is available, it is tacitly assumed that either $L$ or $U$ more or less approximates the identity matrix and one of the two matrices $L^{-1}$ and $U^{-1}$, or both of them, are neglected [16]. Then, procedures for building a sparse approximation of the resulting middle factor are proposed and theoretically analyzed. However, in our case the previous assumption implies that $A + \alpha I$ is almost diagonal for any $\alpha \geq 0$, thus making the updating strategy not very meaningful.

To overcome the above mentioned drawback, we derive an update technique, for an $LDL^T$ factorization of $A$, that modifies only the nonzero entries of the $L$ factor while leaving $D$ unchanged. The modifications applied to $L$ force the preconditioner to mimic the diagonal of the shifted matrix and to reproduce its overall qualitative behaviour. The updating rule is motivated by a componentwise analysis of the preconditioner and the shifted matrix; remarkably, this analysis provides a deep understanding of the procedure. Besides theoretical properties, our updating strategy has nice computational features, as it preserves the sparsity pattern of $L$, has a low computational overhead and does not require tuning any algorithmic parameters, apart from those possibly used in the construction of a preconditioner of $A$. Numerical experiments show that it can work well for a broad range of values of $\alpha$ and is competitive with the approach given in [6]. A posteriori, we found a formal equivalence between our strategy and one of the procedures proposed by Meurant in [25]. In that paper, a preconditioner for $A + \alpha I$ is built by updating an incomplete Cholesky factorization of $A$ of the form $(L_M + \Sigma)\Sigma^{-1}(L_M + \Sigma)^T$, where $\Sigma$ is diagonal and $L_M$ is strictly lower triangular. Since the differences between the entries of the factors of $A$ and those of $A+\alpha I$ depend on $\alpha$, asymptotic expansions in terms of $\alpha$, of order 0 or 1, are applied to the latter to derive updates of the incomplete Cholesky factorization of $A$; furthermore, the recursiveness of the modifications undergone by the matrix entries during the Cholesky factorization is neglected. No theoretical results on the quality of the order 0 and 1 preconditioners are provided in [25]. Here we show the equivalence between our procedure and that based on the order 0 asymptotic expansion. As a consequence, our theoretical analysis offers new insights on the properties of that procedure and makes clear that an apparently low-accuracy approximation gives rise to a well-founded and efficient procedure.

The remainder of this paper is organized as follows. In section 2 we provide some examples of problems requiring the solution of sequences of shifted linear systems, which provide motivations for our work. In section 3 we present our procedure, while in section 4 we analyze the quality of the resulting preconditioner, by providing an estimate of its distance from $A + \alpha I$, as well as results on the clustering of the eigenvalues of the preconditioned matrix for small and large values of $\alpha$. In section 5 we show the equivalence between our update strategy and Meurant's order 0 strategy. In section 6 we discuss the results of numerical experiments, showing the effectiveness of the proposed technique. Finally, in section 7, we give some concluding remarks and outline future work.

In the following, $\|\cdot\|$ denotes the vector or matrix 2-norm. For any square matrix $B$, $diag(B)$ is the diagonal matrix with the same diagonal entries as $B$, $off(B)$ is the matrix with zero diagonal and the same off-diagonal entries as $B$, i.e. $off(B) = B - diag(B)$, and $[B]_{ij}$, or $b_{ij}$, represents the $(i,j)$-th entry of $B$. Furthermore, if $B$ is symmetric, $\lambda_{min}(B)$ and $\lambda_{max}(B)$ denote the minimum and the maximum

eigenvalue of $B$. Finally, $diag(b_1, \ldots, b_n)$ denotes the diagonal matrix with diagonal entries $b_1, \ldots, b_n$.

**2. Motivating applications.** We briefly describe some problems requiring the solution of sequences of shifted linear systems, which arise in numerical optimization and in the numerical solution of PDEs.

**2.1. Trust-region and regularized subproblems in optimization.** Let us consider the following problems which arise, e.g., as subproblems in unconstrained nonlinear least squares [14, Chapter 10]:

$$\min_p \mathcal{M}(p) = \frac{1}{2}\|F + Jp\|_2^2, \quad \|p\|_2 \leq \Delta, \tag{2.1}$$

and

$$\min_p \mathcal{M}(p) = \frac{1}{h}\|F + Jp\|_2^h + \frac{\sigma}{k}||p||_2^k, \tag{2.2}$$

where $F \in \mathbb{R}^m$, $J \in \mathbb{R}^{m\times n}$ is full rank, $m \geq n$, $\Delta > 0$, $\sigma > 0$, and $h$ and $k$ are integers such that $h > 0$ and $k > 1$. Specifically, the solution of (2.1) is required to obtain the trial step in trust-region methods [26], while the solution of (2.2) is needed to compute the trial step in recent regularization approaches where $h = 1$, $k = 2$, or $h = 2$, $k = 3$ [4, 12, 27]. Subproblems of the previous type may also arise in numerical methods for constrained optimization, when computing a "normal step" with the aim of reducing the constraint infeasibility (see, e.g., [28, Chapter 18]).

The solution of (2.1) and (2.2) can be accomplished by solving a so-called secular equation, that is by finding the unique positive root $\lambda_*$ of a secular function $\phi : \mathbb{R} \to \mathbb{R}$, whose form depends on the method used. For example, it is well known that any global minimizer $p_*$ of (2.1) satisfies, for a certain $\lambda_* \geq 0$,

$$(J^TJ + \lambda_* I)p_* = -J^TF, \quad \lambda_*(\|p_*\|_2 - \Delta) = 0 \tag{2.3}$$

(see [26, Lemma 2.1]). Letting $p(\lambda) : \mathbb{R} \to \mathbb{R}^n$ be such that

$$(J^TJ + \lambda I)p(\lambda) = -J^TF, \quad \lambda \geq 0, \tag{2.4}$$

it follows that either $\lambda_* = 0$, $p_* = -(J^TJ)^{-1}J^TF$ and $\|p_*\|_2 \leq \Delta$, or $p_* = p(\lambda_*)$, where $\lambda_*$ is the positive solution of the secular equation

$$\phi(\lambda) = \|p(\lambda)\|_2^2 - \Delta^2 = 0. \tag{2.5}$$

The Newton method or the secant method applied to (2.5) produce a sequence $\{\lambda_l\}$ of scalars and require the evalutation of $\phi$ at each iterate. This amounts to solve a sequence of shifted linear systems of the form (2.4).

Similarly, the characterization of the minimizer of (2.2) leads to the solution of a secular equation, which in turn requires the solution of a sequence of linear systems of the form (2.4) [4, 12, 27].

When the dimension $n$ of the problem is large, the linear systems (2.4) can be solved by the LSQR method [29] and our preconditioner update technique. It is worth mentioning that, alternatively, LSQR can be used to find constrained minimizers of (2.1) and (2.2) over a sequence of expanding subspaces associated with the Lanczos process for reducing $J$ to bi-diagonal form [12].

We conclude this section noting that shifted matrices of the form $J^TJ + \lambda I$ arise also in the very recent method presented in [19]. This method solves the inequality constrained problem (2.1) over a sequence of evolving low-dimensional subspaces and employs preconditioners for a sequence of shifted matrices in the calculation of one of the vectors spanning the subspaces [19, §3.4].

**2.2. Solution of PDEs by implicit methods.** Let us consider the well-known heat equation:

$$\frac{\partial u}{\partial t}(t,x) - \beta \Delta u(t,x) = f(t,x), \tag{2.6}$$

where $t \geq 0$, $x \in [a,b] \times [c,d]$ and $\beta$ is a constant, with Dirichlet boundary conditions and an initial condition $u(0,x) = u_0(x)$. By discretizing (2.6) in space by central differences on a Cartesian grid and in time by the backward Euler method, we obtain a sequence of linear systems of the form:

$$\left(I + \frac{\Delta t}{(\Delta x)^2} A\right) u^{k+1} = u^k + \Delta t\, f^k, \tag{2.7}$$

where $A$ is a symmetric positive definite matrix, $\Delta t$ and $\Delta x$ are the grid spacing and the time stepsize, respectively, and $k$ denotes the current time step. A simple multiplication by $\alpha = (\Delta x)^2/\Delta t$ yields a sequence of systems of the form (1.1). Since the stepsize is usually chosen adaptively at each time step by estimating the accuracy in the computed solution, the value of $\alpha$ changes during the integration process.

More generally, sequences of systems of type (2.7) arise in the solution of initial-boundary value problems for diffusion equations, when implicit methods such as, e.g., BDF and SDIRK are applied [1]. The involved matrices are usually very large and sparse, thus requiring the application of Krylov solvers with suitable preconditioners.

**3. A strategy for updating the preconditioner.** In this section we introduce a preconditioner $P_\alpha$ for the matrix $A+\alpha I$, $\alpha > 0$, which is an update of an incomplete factorization of $A$. More precisely, we assume that an incomplete $LDL^T$ factorization of $A$ is available, where $L$ is a unit lower triangular matrix and $D$ is a diagonal matrix with positive entries, i.e. $LDL^T$ is an incomplete square root - free Cholesky factorization of $A$.

We define $P_\alpha$ as follows:

$$P_\alpha = (L+G)D(L+G)^T, \tag{3.1}$$

where $G$ is the sum of two matrices:

$$G = E + F, \tag{3.2}$$

with $E$ diagonal and $F$ strictly lower triangular. The matrix $E = diag(e_{11}, \ldots, e_{nn})$ is defined by

$$e_{ii} = \sqrt{1 + \frac{\alpha}{d_{ii}}} - 1, \quad 1 \leq i \leq n, \tag{3.3}$$

while the nonzero entries of the matrix $F = (f_{ij})$ are given by

$$f_{ij} = \gamma_j\, l_{ij}, \quad 2 \leq i \leq n, \ 1 \leq j < i, \tag{3.4}$$

$$\gamma_j = \frac{1}{\sqrt{1 + \frac{\alpha}{d_{jj}}}} - 1 = \frac{1}{e_{jj}+1} - 1. \tag{3.5}$$

Here and in the following the dependence of $e_{ii}$, $f_{ij}$ and $\gamma_j$ on $\alpha$ is neglected for simplicity. Furthermore, we extend the definition of $\gamma_j$ in (3.5) to $j = n$. It is immediate to see that the entries of $E$ are positive, $\gamma_j$ is a decreasing function of $e_{jj}$, for $e_{jj} \in (0, +\infty)$, and $\gamma_j \in (-1, 0)$ for $\alpha > 0$. We note also that, for all $i$,

$$\lim_{\alpha \to 0} e_{ii} = 0, \qquad \lim_{\alpha \to +\infty} e_{ii} = +\infty, \tag{3.6}$$

$$\lim_{\alpha \to 0} \gamma_i = 0, \qquad \lim_{\alpha \to +\infty} \gamma_i = -1. \tag{3.7}$$

In practice, the update of the $LDL^T$ factorization of $A$ consists in adding the term $e_{ii}$ to the $i$-th unit diagonal entry of $L$ and scaling the subdiagonal entries of the $i$-th columns of $L$ with the scalar $\gamma_i + 1$. Remarkably, the sparsity pattern of the factors of $P_\alpha$ in (3.1) is the same of the factors of the incomplete $LDL^T$ factorization of $A$.

The construction of the matrix $G$ is inspired by the following observations. Suppose that $LDL^T$ is the exact factorization of $A$. The shift by $\alpha I$ modifies only the diagonal entries of $A$; furthermore, the larger is $\alpha$ the stronger is the effect of this modification on the matrix $A$. Therefore, we would like to have a preconditioner $\widetilde{P}_\alpha$ such that

$$[\widetilde{P}_\alpha]_{ii} = a_{ii} + \alpha.$$

This can be achieved by modifying the diagonal of $L$, i.e. by considering $\widetilde{P}_\alpha = (L + E)D(L + E)^T$, with $E$ diagonal, and imposing that

$$\sum_{j=1}^{i-1} l_{ij}^2 d_{jj} + (l_{ii} + e_{ii})^2 d_{ii} = \sum_{j=1}^{i-1} l_{ij}^2 d_{jj} + l_{ii}^2 d_{ii} + \alpha, \quad i = 1, \ldots, n. \tag{3.8}$$

From (3.8) and $l_{ii} = 1$ we obtain

$$(1 + e_{ii})^2 d_{ii} = d_{ii} + \alpha, \tag{3.9}$$

which is equivalent to (3.3). On the other hand, the off-diagonal entries of $\widetilde{P}_\alpha$ take the following form:

$$[\widetilde{P}_\alpha]_{ij} = [\widetilde{P}_\alpha]_{ji} = \sum_{k=1}^{j-1} l_{ik} d_{kk} l_{jk} + l_{ij} d_{jj}(1 + e_{jj}), \quad i > j, \tag{3.10}$$

and, by using the second limit in (3.6), we obtain

$$\lim_{\alpha \to +\infty} [\widetilde{P}_\alpha]_{ij} = +\infty. \tag{3.11}$$

In other words, the off-diagonal entries of $\widetilde{P}_\alpha$ increase with $\alpha$, and hence they depart from the corresponding entries of $A + \alpha I$, which are independent of $\alpha$. This means that $\widetilde{P}_\alpha$ cannot be a good approximation of $A + \alpha I$ when $\alpha$ is large.

To overcome the previous drawback, we relax the requirement that the preconditioner has the same diagonal as $A + \alpha I$. By defining the preconditioner as in (3.1), we have that its off-diagonal entries are kept bounded, while the diagonal ones satisfy

$$[P_\alpha]_{ii} - (a_{ii} + \alpha) = \mu, \tag{3.12}$$

where

$$\mu = \sum_{j=1}^{i-1} (1 - (1+\gamma_j)^2)\, l_{ij}^2\, d_{jj}.$$

By using (3.7), it is immediate to verify that the magnitude of $\mu$ is small compared to $a_{ii} + \alpha$ as $\alpha$ tends to zero or infinity. Furthermore, the stricly lower triangle of $L + F$ tends to $L$ when $\alpha$ tends to zero, and vanishes when $\alpha$ tends to infinity, thus preserving the qualitative behaviour of the off-diagonal part of $A + \alpha I$. We note also that the first row and column of $P_\alpha$ are equal to the corresponding row and column of $A + \alpha I$. A rigorous analysis of the behaviour of $P_\alpha$, showing that it can be effective for a broad range of values of $\alpha$, is carried out in the next section.

**4. Analysis of the preconditioner.** For the sake of simplicity, we assume that $LDL^T$ is the exact factorization of $A$, i.e.

$$A = LDL^T; \tag{4.1}$$

however, the following analysis can be easily extended to the case where the factorization of $A$ is incomplete.

We provide first an estimate of $\|P_\alpha - (A + \alpha I)\|$, i.e. of the accuracy of the preconditioner as an approximation of $A + \alpha I$, for all values of $\alpha$, showing that the distance between $P_\alpha$ and $(A + \alpha I)$ is bounded independently of $\alpha$. We study also the behaviour of the preconditioner as $\alpha$ tends to zero or infinity, demonstrating that the preconditioner behaves particularly well in these limiting cases. Finally, we analyse the spectrum of $P_\alpha^{-1}(A + \alpha I)$, showing that a certain number of eigenvalues may be equal to 1, depending on the rank of $P_\alpha - (A + \alpha I)$, and that all the eigenvalues are clustered in a neighbourhood of 1 when $\alpha$ is sufficiently small or large.

Letting

$$S = diag(\gamma_1, ..., \gamma_n), \tag{4.2}$$

we can write $F$ as

$$F = off(L)S. \tag{4.3}$$

Furthermore, it is easy to see that

$$S = (I + E)^{-1} - I \quad \text{and} \quad E + SE + S = 0. \tag{4.4}$$

The following Lemma provides an expression of the difference between $P_\alpha$ and $A + \alpha I$.

LEMMA 4.1. *The preconditioner $P_\alpha$ given in (3.1) satisfies*

$$P_\alpha = A + \alpha I + Z, \tag{4.5}$$

*where*

$$Z = diag(B_1) + off(B_2), \tag{4.6}$$

*and the diagonal entries of $B_1$ and the off-diagonal entries of $B_2$ are given by*

$$[B_1]_{ii} = \sum_{k=1}^{i-1} l_{ik}^2\, d_{kk}\, \gamma_k(\gamma_k + 2), \quad 1 \le i \le n, \tag{4.7}$$

$$[B_2]_{ij} = [B_2]_{ji} = \sum_{k=1}^{j-1} l_{ik}\, l_{jk}\, d_{kk}\, \gamma_k(\gamma_k + 2), \quad 2 \le i \le n,\ 1 \le j < i. \tag{4.8}$$

*Proof.* We start noting that

$$\begin{aligned} P_\alpha &= (L+G)D(L+G)^T \\ &= LDL^T + \alpha I + GDL^T + LDG^T + GDG^T - \alpha I \\ &= A + \alpha I + GDL^T + LDG^T + GDG^T - \alpha I. \end{aligned}$$

Then, letting

$$Z = GDL^T + LDG^T + GDG^T - \alpha I, \tag{4.9}$$

we have

$$P_\alpha = A + \alpha I + diag(Z) + off(Z).$$

We first prove that $diag(Z) = diag(B_1)$ with $B_1$ given by

$$B_1 = FDL^T + LDF^T + FDF^T. \tag{4.10}$$

From the definition of $E$ in (3.3) it follows that

$$diag(E^2D + LDE + EDL^T - \alpha I) = 0;$$

then, using (3.2) we have

$$\begin{aligned} diag(Z) &= diag(EDL^T + FDL^T + LDE + LDF^T + E^2D \\ &\qquad + EDF^T + FDE + FDF^T - \alpha I) \\ &= diag(FDL^T + LDF^T + EDF^T + FDE + FDF^T) \\ &= diag(FDL^T + LDF^T + FDF^T), \end{aligned}$$

where the last equality depends on the fact that $FDE$ and $EDF^T$ are strictly lower and upper triangular matrices, respectively. Hence, $diag(Z) = diag(B_1)$ and from (4.10) it is easy to see that the diagonal entries of $B_1$ have the form given in (4.7).

Now we show that $off(Z) = off(B_2)$ where $B_2$ is given by

$$B_2 = off(L)(2DS + DS^2)off(L)^T. \tag{4.11}$$

Using (3.2) we have

$$GDG^T = (E+F)D(E+F)^T = E^2D + FDE + EDF^T + FDF^T,$$

and, since $E^2D$ is diagonal, (4.9) yields

$$off(Z) = off(GDL^T + LDG^T + FDE + EDF^T + FDF^T).$$

From (3.2) and (4.3) it follows that $G = E + off(L)S$, and, using $L = I + off(L)$, we obtain

$$\begin{aligned} GDL^T + LDG^T + FDE + EDF^T + FDF^T = {} & 2DE + (E + S + ES)Doff(L)^T \\ & + off(L)D(E + S + ES) + 2off(L)SDoff(L)^T \\ & + off(L)SDSoff(L)^T. \end{aligned}$$

By taking into account that $DE$ is diagonal and by (4.4), we get

$$off(Z) = off(off(L)(2DS + SDS)off(L)^T) = off(B_2).$$

From the previous expression it follows that $B_2$ is symmetric and its off-diagonal entries have the form given in (4.8), which completes the proof. ∎

Note that the equalities (4.7) and (4.8) include also $[B_1]_{11} = 0$ and $[B_2]_{i1} = [B_2]_{1i} = 0$, $2 \le i \le n$, which, by (4.5), means that the first row and column of $P_\alpha$ are equal to those of $(A + \alpha I)$.

Next, we provide technical results.

LEMMA 4.2. *The matrices $L$ and $D$ in (4.1) satisfy*

$$\|off(L)Doff(L)^T\| \le 4\|LDL^T\|. \tag{4.12}$$

*Also, if $W$ is a positive semidefinite matrix, then*

$$\|off(W)\| \le \|W\|. \tag{4.13}$$

*Proof.* The matrix $R = LD^{\frac{1}{2}}$ is the Cholesky factor of $A$ and hence satisfies $\|A\| = \|R\|^2$. Furthermore,

$$\|off(L)Doff(L)^T\| = \|off(L)D^{\frac{1}{2}}D^{\frac{1}{2}}off(L)^T\| = \|off(R)off(R)^T\| = \|off(R)\|^2.$$

Then, (4.12) follows from

$$\|off(R)\| = \|R - diag(R)\| \le \|R\| + \|diag(R)\| \le \|A\|^{\frac{1}{2}} + \max_i a_{ii}^{\frac{1}{2}} \le 2\|A\|^{\frac{1}{2}}. \tag{4.14}$$

Now we prove (4.13). Since $W$ is positive semidefinite, we have $W = off(W) + diag(W)$, where $off(W)$ is indefinite with zero trace and $diag(W)$ is positive semidefinite. From [20, Theorem 8.1.5] we have

$$\lambda_{max}(W) \ge \lambda_{max}(off(W)) + \lambda_{min}(diag(W)),$$

which implies

$$\lambda_{max}(W) \ge \lambda_{max}(off(W)).$$

Furthermore, it is easy to see that $\|W\|I - diag(W)$ is positive semidefinite and hence $\|W\|I + off(W) = \|W\|I - diag(W) + W$ is positive semidefinite too. Thus, $\lambda_{min}(\|W\|I + off(W)) \ge 0$, i.e.

$$|\lambda_{min}(off(W))| = -\lambda_{min}(off(W)) \le \|W\|,$$

which completes the proof. ∎

The quality of $P_\alpha$ as a preconditioner of $A + \alpha I$ is studied in the following theorems. Let us introduce the quantities

$$\omega_k = \gamma_k(\gamma_k + 2) = -\frac{\alpha}{\alpha + d_{kk}}, \quad k = 1, \ldots, n, \tag{4.15}$$

where $\gamma_k$ is defined in (3.5). For $k = 1, \ldots, n$, $\omega_k$ is a monotonically decreasing function of $\alpha$ such that

$$\lim_{\alpha \to 0} \omega_k = 0, \qquad \lim_{\alpha \to \infty} \omega_k = -1, \tag{4.16}$$

and

$$\max_k |\omega_k| = \frac{\alpha}{\alpha + \min_k d_{kk}}. \tag{4.17}$$

THEOREM 4.3. *For all $\alpha > 0$, the matrix $Z$ given in (4.6) satisfies*

$$\|Z\| = \|P_\alpha - (A+\alpha I)\| \le \max_k |\omega_k| \,(\max_i (a_{ii} - d_{ii}) + \|\mathit{off}(L) D^{\frac{1}{2}}\|^2) \tag{4.18}$$

$$\le \max_k |\omega_k| (5\lambda_{max}(A) - \lambda_{min}(A)) \tag{4.19}$$

*and*

$$\frac{\|P_\alpha - (A+\alpha I)\|}{\|A+\alpha I\|} \le \max_k |\omega_k| \frac{5\lambda_{max}(A) - \lambda_{min}(A)}{\lambda_{max}(A) + \alpha} \tag{4.20}$$

*Proof.* Let us derive the inequality (4.18). By (4.7),

$$[B_1]_{ii} = \sum_{k=1}^{i-1} l_{ik}^2 \, d_{kk} \omega_k, \qquad |l_{ik}^2 \, d_{kk} \omega_k| = l_{ik}^2 \, d_{kk} |\omega_k|;$$

then, since $a_{ii} - d_{ii} = \sum_{k=1}^{i-1} l_{ik}^2 \, d_{kk}$, we have

$$\|\mathit{diag}(B_1)\| \le \max_k |\omega_k| \, \max_i (a_{ii} - d_{ii}). \tag{4.21}$$

Using (4.11) and (4.13) we obtain

$$\begin{aligned} \|\mathit{off}(B_2)\| &= \|\mathit{off}(\mathit{off}(L) D^{\frac{1}{2}} (2S + S^2) D^{\frac{1}{2}} \mathit{off}(L)^T)\| \\ &\le \|\mathit{off}(L) D^{\frac{1}{2}} (2S + S^2) D^{\frac{1}{2}} \mathit{off}(L)^T\| \\ &\le \|2S + S^2\| \, \|\mathit{off}(L) D^{\frac{1}{2}}\|^2. \end{aligned}$$

Hence, we can conclude that

$$\|\mathit{off}(B_2)\| \le \max_k |\omega_k| \, \|\mathit{off}(L) D^{\frac{1}{2}}\|^2. \tag{4.22}$$

The definition (4.6) of $Z$, along with (4.21) and (4.22), yields (4.18). Furthermore, from the equality $d_{ii} = (L^{-1})_i A (L^{-1})_i^T$, where $(L^{-1})_i$ is the $i$-th row of $L$, and the properties of the Rayleigh quotient it follows that

$$\max_i (a_{ii} - d_{ii}) \le \max_i a_{ii} - \min_i d_{ii} \le \lambda_{max}(A) - \lambda_{min}(A).$$

This inequality and (4.14) provide (4.19) and (4.20). ∎

The previous theorem shows that $\|P_\alpha - (A+\alpha I)\|$ remains bounded for all values of $\alpha$. Moreover, from (4.17) it follows that the absolute distance $\|P_\alpha - (A+\alpha I)\|$ goes to zero with $\alpha$, while the relative distance $\|P_\alpha - (A+\alpha I)\| / \|A + \alpha I\|$ tends to zero as $\alpha$ increases. A componentwise analysis of the behaviour of $P_\alpha$ for $\alpha \to 0$ and $\alpha \to +\infty$ is carried out in the next theorem.

THEOREM 4.4. *The matrix $P_\alpha$ satisfies*

$$\lim_{\alpha \to 0} [P_\alpha]_{ij} = a_{ij}, \qquad 1 \le i, j \le n, \tag{4.23}$$

$$\lim_{\alpha \to +\infty} \frac{[P_\alpha]_{ii}}{\alpha + d_{ii}} = 1, \qquad 1 \le i \le n, \tag{4.24}$$

$$\lim_{\alpha \to +\infty} [P_\alpha]_{ij} = l_{ij} d_{jj}, \qquad 2 \le i \le n, \ 1 \le j < i. \tag{4.25}$$

*Proof.* Let us consider (4.5) along with (4.6). From (4.7) and (3.7) we get

$$\lim_{\alpha\to 0}[B_1]_{ii} = 0, \tag{4.26}$$

$$\lim_{\alpha\to +\infty}[B_1]_{ii} = \sum_{j=1}^{i-1} -d_{jj}l_{ij}^2 = -a_{ii} + d_{ii}. \tag{4.27}$$

Thus, equality (4.5) implies (4.23), for $i = j$, and (4.24).

From the expression of the off-diagonal entries of $B_2$ in (4.8), by using (3.7), for $i > j$ we get

$$\lim_{\alpha\to 0}[off(B_2)]_{ij} = 0, \tag{4.28}$$

$$\lim_{\alpha\to +\infty}[off(B_2)]_{ij} = -a_{ij} + l_{ij}d_{jj}. \tag{4.29}$$

The above equalities, along with (4.5), yield (4.23), for $i \neq j$, and (4.25). ∎

The previous theorem shows that for $\alpha \to 0$ the preconditioner $P_\alpha$ behaves like $A+\alpha I$, since both approach the matrix $A$; furthermore, for $\alpha \to +\infty$ the diagonal of $P_\alpha$ behaves like the diagonal of $A + \alpha I$, which, in turn, dominates over the remaining entries, while the off-diagonal part of $P_\alpha$ is kept bounded.

The analysis of the eigenvalues of $P_\alpha^{-1}(A + \alpha I)$ is performed in the following theorem.

THEOREM 4.5. *For all $\alpha > 0$, if the matrix $Z$ has rank $n - k$, then $k$ eigenvalues of $P_\alpha^{-1}(A + \alpha I)$ are equal to 1. Furthermore, for sufficiently small values of $\alpha$, any eigenvalue $\lambda$ of $P_\alpha^{-1}(A + \alpha I)$ satisfies*

$$|\lambda - 1| = \frac{O(\alpha)}{\lambda_{min}(A) - O(\alpha)}, \tag{4.30}$$

*while, for sufficiently large values of $\alpha$, it satisfies*

$$|\lambda - 1| \leq \frac{||A^{-1}Z||}{1 + \dfrac{\alpha}{\lambda_{max}(A)} - \|A^{-1}Z\|}. \tag{4.31}$$

*Proof.* By using (4.5) we get

$$P_\alpha^{-1}(A + \alpha I) = (A + \alpha I + Z)^{-1}(A + \alpha I) = (I + \alpha A^{-1} + A^{-1}Z)^{-1}(I + \alpha A^{-1}).$$

Then, if $\lambda$ is an eigenvalue of $P_\alpha^{-1}(A + \alpha I)$, we have

$$(I + \alpha A^{-1})v = \lambda(I + \alpha A^{-1} + A^{-1}Z)v, \tag{4.32}$$

where $v$ is an eigenvector corresponding to $\lambda$. Without loss of generality, we assume $\|v\| = 1$. From (4.32) it follows that $\lambda = 1$ if and only if $A^{-1}Zv = 0$, i.e. $v$ belongs to the null space of $Z$. So, if $rank(Z) = n - k$, it follows that there are at least $k$ unit eigenvalues.

Now suppose that $A^{-1}Zv \neq 0$ and multiply (4.32) by $v^T$, obtaining

$$1 + \alpha v^T A^{-1} v = \lambda(1 + \alpha v^T A^{-1} v + v^T A^{-1} Z v), \tag{4.33}$$

and hence

$$|\lambda - 1| = \left|\frac{v^T A^{-1} Z v}{1 + \alpha v^T A^{-1} v + v^T A^{-1} Z v}\right| \leq \frac{\|A^{-1}Z\|}{|1 + \alpha v^T A^{-1} v + v^T A^{-1} Z v|}, \tag{4.34}$$

where $1 + \alpha v^T A^{-1} v + v^T A^{-1} Z v \neq 0$ by (4.33). First, we focus on the case where $\alpha$ is small. We observe that

$$\begin{aligned} |\alpha v^T A^{-1} v + v^T A^{-1} Z v| &\leq \|\alpha A^{-1}\| + \|A^{-1} Z\| \\ &\leq \|A^{-1}\|(\alpha + \|Z\|) \\ &= \frac{\alpha + \|Z\|}{\lambda_{min}(A)}. \end{aligned}$$

From (4.26) and (4.28), it easily follows that $Z = O(\alpha)$, therefore, for sufficiently small values of $\alpha$ we get

$$\begin{aligned} |1 + \alpha v^T A^{-1} v + v^T A^{-1} Z v| &\geq 1 - |\alpha v^T A^{-1} v + v^T A^{-1} Z v| \\ &\geq 1 - \frac{\alpha + \|Z\|}{\lambda_{min}(A)} > 0. \end{aligned}$$

Then, inequality (4.34) yields

$$|\lambda - 1| \leq \frac{\|A^{-1} Z\|}{1 - \dfrac{\alpha + \|Z\|}{\lambda_{min}(A)}}, \tag{4.35}$$

which implies (4.30), since $\|Z\| = O(\alpha)$ and $\|A^{-1}\| = 1/\lambda_{min}(A)$.

Let us consider now the case where $\alpha$ is large. From (4.18)-(4.19) we know that $\|Z\|$ is bounded for all $\alpha > 0$; therefore, for sufficiently large values of $\alpha$, the denominator in (4.34) is bounded as follows:

$$\begin{aligned} |1 + \alpha v^T A^{-1} v + v^T A^{-1} Z v| &\geq |1 + \alpha v^T A^{-1} v| - |v^T A^{-1} Z v| \\ &\geq \lambda_{min}(I + \alpha A^{-1}) - |v^T A^{-1} Z v| \\ &\geq 1 + \frac{\alpha}{\lambda_{max}(A)} - \|A^{-1} Z\| > 0, \end{aligned}$$

where the second inequality follows from the fact that $1 + \alpha v^T A^{-1} v$ is a Rayleigh quotient of $I + \alpha A^{-1}$. Then, (4.31) follows from (4.34). □

We note that $Z$ has rank at most $n - 1$, since the first row and column of $Z$ are zero; then, $P_\alpha^{-1}(A + \alpha I)$ has at least one unit eigenvalue. We observe also that (4.30) means that for small values of $\alpha$ the eigenvalues of $P_\alpha^{-1}(A + \alpha I)$ are clustered in a neighbourhood of 1. Analogously, from (4.31) it follows that for large values of $\alpha$ the eigenvalues are again clustered in a neighbourhood of 1, since $\|Z\|$ is bounded independently of $\alpha$.

The previous theorems hold under the assumption that we have the exact $LDL^T$ factorization of $A$, but the results can be easily extended to an incomplete factorization $A \approx LDL^T$. In this case, a term depending on $\|A - LDL^T\|$ must be added to the upper bound on $\|P_\alpha - (A + \alpha I)\|$; furthermore, for large values of $\alpha$ the eigenvalues of $P_\alpha^{-1}(A + \alpha I)$ are still clustered around 1, while for small values of $\alpha$ the distance of the eigenvalues from 1 depends on the accuracy of the incomplete factorization, i.e. the more accurate such factorization is, the smaller the distance is expected to be.

**5. Equivalence with Meurant's order 0 procedure.** For the sake of clarity, we first provide a brief description of Meurant's order 0 preconditioning technique, following the presentation in [25]. This technique starts from a factorization of the

matrix $A$ which is different from (4.1) (as in section 4, we consider an exact factorization of $A$ for simplicity). Since the procedure for obtaining this factorization is the basis for understanding the order 0 approximate factorization of $A + \alpha I$, we briefly recall it.

Let us consider the following partitioning of $A$:

$$A = A_1 = \begin{pmatrix} a_{11} & a_1^T \\ a_1 & Q_2 \end{pmatrix},$$

where $a_{11} \in \mathbb{R}$, $a_1 \in \mathbb{R}^{n-1}$ and $Q_2 \in \mathbb{R}^{(n-1)\times(n-1)}$. We can factorize $A_1$ as

$$A_1 = \begin{pmatrix} a_{11} & 0^T \\ l_1 & I \end{pmatrix} \begin{pmatrix} a_{11}^{-1} & 0^T \\ 0 & A_2 \end{pmatrix} \begin{pmatrix} a_{11} & l_1^T \\ 0 & I \end{pmatrix} = L_1 \Delta_1 L_1^T, \tag{5.1}$$

where

$$l_1 = a_1, \quad A_2 = Q_2 - \frac{1}{a_{11}} a_1 a_1^T,$$

and 0 and $I$ are the zero vector and the identity matrix of appropriate dimensions, respectively. By applying to $A_2$ the same factorization as in (5.1), we get

$$A_2 = \begin{pmatrix} a_{22}^{(2)} & a_2^T \\ a_2 & Q_3 \end{pmatrix} = \begin{pmatrix} a_{22}^{(2)} & 0^T \\ l_2 & I \end{pmatrix} \begin{pmatrix} (a_{22}^{(2)})^{-1} & 0^T \\ 0 & A_3 \end{pmatrix} \begin{pmatrix} a_{22}^{(2)} & l_2^T \\ 0 & I \end{pmatrix} = \widetilde{L}_2 \widetilde{\Delta}_2 \widetilde{L}_2^T,$$

where $a_{22}^{(2)} \in \mathbb{R}$, $a_2 \in \mathbb{R}^{n-2}$, $Q_3 \in \mathbb{R}^{(n-2)\times(n-2)}$, and

$$l_2 = a_2, \quad A_3 = Q_3 - \frac{1}{a_{22}^{(2)}} a_2 a_2^T$$

(the superscript $^{(2)}$ in $a_2$ has been neglected for simplicity). By defining

$$L_2 = \begin{pmatrix} 1 & 0^T \\ 0 & \widetilde{L}_2 \end{pmatrix} \quad \text{and} \quad \Delta_2 = \begin{pmatrix} a_{11}^{-1} & 0^T \\ 0 & \widetilde{\Delta}_2 \end{pmatrix},$$

we have

$$A = L_1 L_2 \Delta_2 L_2^T L_1^T.$$

By repeating this reasoning, after $n$ steps we obtain

$$A = L_1 L_2 \dots L_n \Delta_n L_n^T \dots L_2^T L_1^T,$$

where $\Delta_n = diag\left(a_{11}^{-1}, (a_{22}^{(2)})^{-1}, \dots, (a_{nn}^{(n)})^{-1}\right)$ and the $k$-th diagonal entry $a_{kk}^{(k)}$ is the upper diagonal entry of $A_k$. Letting

$$\Sigma = \Delta_n^{-1} \quad \text{and} \quad L_M = off(L_1 L_2 \dots L_n), \tag{5.2}$$

it follows that

$$diag(L_1 L_2 \dots L_n) = \Sigma,$$

and hence

$$A = (L_M + \Sigma)\Sigma^{-1}(L_M + \Sigma). \tag{5.3}$$

The same procedure can be applied to obtain a factorization of the matrix $A+\alpha I$. Specifically, at the first step we have:

$$\begin{aligned} A+\alpha I = A_1(\alpha) &= \begin{pmatrix} a_{11}+\alpha & a_1^T \\ a_1 & Q_2(\alpha) \end{pmatrix} \\ &= \begin{pmatrix} a_{11}+\alpha & 0^T \\ l_1 & I \end{pmatrix} \begin{pmatrix} (a_{11}+\alpha)^{-1} & 0^T \\ 0 & A_2(\alpha) \end{pmatrix} \begin{pmatrix} a_{11}+\alpha & l_1^T \\ 0 & I \end{pmatrix} \\ &= L_1(\alpha)\Delta_1(\alpha)L_1^T(\alpha), \end{aligned}$$

where

$$A_2(\alpha) = Q_2(\alpha) - \frac{1}{a_{11}+\alpha} a_1 a_1^T,$$

and the meaning of the remaining notations is clear from the context. By using an order 0 asymptotic expansion of $1/(a_{11}+\alpha)$, we can approximate $A_2$ as follows:

$$A_2(\alpha) \approx \bar{A}_2(\alpha) = Q_2(\alpha) - \frac{1}{a_{11}} a_1 a_1^T.$$

We can now perform the next step of the factorization procedure on the matrix $\bar{A}_2(\alpha)$ and apply the order 0 asymptotic expansion to the resulting matrix $A_3(\alpha)$. By repeating this process, at the $n$-th step we get Meurant's order 0 approximate factorization.

Before proving the equivalence of our preconditioner update strategy with Meurant's strategy, we make a few observations. By induction, we can see that the approximate matrix $\bar{A}_k(\alpha)$ obtained at the $(k-1)$-th step of the approximate factorization procedure, is

$$\bar{A}_k(\alpha) = Q_k(\alpha) - \frac{1}{a_{k-1\,k-1}^{(k-1)}} a_{k-1} a_{k-1}^T,$$

and hence

$$\bar{A}_k(\alpha) = A_k + \alpha I.$$

By using the previous equality and (5.3), we obtain the following expression of the order 0 preconditioner $P_M(\alpha)$ for $A+\alpha I$:

$$P_M(\alpha) = (L_M + \Sigma + \alpha I)(\Sigma + \alpha I)^{-1}(L_M + \Sigma + \alpha I)^T. \tag{5.4}$$

The following theorem states the equivalence between the update strategy discussed in this paper and Meurant's order 0 approximation procedure.

THEOREM 5.1. *For all $\alpha > 0$, $P_M(\alpha) = P_\alpha$.*

*Proof.* Since the matrix $A$ has a unique Cholesky factorization $A = RR^T$ where $R$ is lower triangular with positive diagonal entries, it is

$$R = (L_M + \Sigma)\Sigma^{-\frac{1}{2}} = LD^{\frac{1}{2}},$$

where $L_M$ and $\Sigma$ are the matrices in (5.2), and $L$ and $D$ are those in (4.1). The previous equality, together with the observation that $L$ has unit diagonal and $L_M$

is strictly lower triangular, yields $\Sigma = D$ and $L_M = \mathit{off}(L)D$. Then, from (5.4) it follows that

$$\begin{aligned} &P_M(\alpha) = \\ &= (\mathit{off}(L)D + D + \alpha I)\,(D + \alpha I)^{-1}\,(\mathit{off}(L)D + D + \alpha I)^T \\ &= \left(\mathit{off}(L)D^{\frac{1}{2}}(D + \alpha I)^{-\frac{1}{2}} + D^{-\frac{1}{2}}(D + \alpha I)^{\frac{1}{2}}\right) D \left(\mathit{off}(L)D^{\frac{1}{2}}(D + \alpha I)^{-\frac{1}{2}} + D^{-\frac{1}{2}}(D + \alpha I)^{\frac{1}{2}}\right)^T. \end{aligned}$$

By observing that the matrix $L + G$ defined by (3.1)-(3.5) can be decomposed as

$$L + G = \mathit{off}(L) + F + I + E$$

and that

$$\mathit{off}(L) + F = \mathit{off}(L)D^{\frac{1}{2}}(D + \alpha I)^{-\frac{1}{2}}, \quad I + E = D^{-\frac{1}{2}}(D + \alpha I)^{\frac{1}{2}},$$

we get the thesis. □

**6. Numerical experiments.** Numerical experiments were carried out on several sequences of shifted linear systems to evaluate the behaviour of our preconditioner update technique, as well as to compare it with other preconditioning strategies.

The systems of each sequence were solved by using the Conjugate Gradient (CG) method with zero initial guess. The CG method was terminated when the ratio between the 2-norm of the current and the initial residual was less than $10^{-6}$, or when a maximum of 1000 iterations was reached; in the latter case a failure was declared if the residual did not achieve the required reduction. Five preconditioning strategies were coupled with CG. Two of them are standard procedures: recomputing the incomplete $LDL^T$ factorization for each value of $\alpha$ (`RP`) and "freezing" the incomplete $LDL^T$ factorization of $A$ for all values of $\alpha$ (`FP`). The remaining ones are updating strategies: our technique for the incomplete $LDL^T$ factorization of $A$ (`UP`), Meurant's order 1 technique (`MP`) in [25], and the approximate inverse technique (`IP`) in [6]. The CG method was run also without preconditioning (`NP`).

All the experiments were performed using Matlab 7.7 on an Intel Core 2 Duo E8500 processor, with clock frequency of 3.16 GHz, 4 GB of RAM and 6 MB of cache memory. The preconditioned CG method was applied through the `pcg` function. The incomplete $LDL^T$ factorizations used in the `UP`, `RP`, and `FP` procedures were obtained using the `cholinc` function and recovering the $L$ and $D$ matrices from the Cholesky factor. We note that `cholinc` implements the zero fill-in factorization and that based on a drop tolerance, say *drop* [31]. For many test sequences, the former showed a very poor performance on the system $Ax = b$, therefore we chose the latter. In this case, the memory required by $L$ is generally unpredictable and the performance of `cholinc` as a function of *drop* may be erratic (see, e.g., [24]). The value of *drop* affects the effectiveness and the computational cost of the preconditioned CG iterations, as well as the cost for computing the seed preconditioner and applying the updating strategy. In particular, the updating strategy works with $L$ and its computational overhead increases with the fill-in in this matrix. However, for most of our test problems the computed factor $L$ required a reasonable amount of memory (see section 6.1).

For each sequence, the value of *drop* was fixed by trial in the solution of the system $Ax = b$. Specifically, a first attempt was made to solve this system using CG preconditioned by an incomplete $LDL^T$ factorization with $\mathit{drop} = 10^{-1}$; if the computation of the incomplete factorization failed or CG did not terminate successfully

within 1000 iterations, then *drop* was reduced by a factor 10. This procedure was repeated until the preconditioned CG achieved the required accuracy in solving $Ax = b$. We remark that in the `UP` and `FP` strategies one incomplete $LDL^T$ factorization was performed for each sequence, while in the `RP` strategy an $LDL^T$ factorization was computed from scratch for each matrix $A + \alpha I$, using the value of *drop* employed to compute the seed preconditioner.

The `MP` technique was applied using Matlab functions kindly provided by G. Meurant. The `IP` one was implemented closely following [6]. Specifically, the seed preconditioner was built using the right-looking SAINV preconditioner available in the `Sparselab` package [32] as a MEX-F90 source with m-file interface. The rule for choosing the drop tolerance for SAINV was the same employed with `cholinc`. Moreover, in (1.2) we used the matrix $V_\alpha$ defined in [6, p.243] instead of the matrix $V$:

$$V_\alpha = \begin{cases} (1-\alpha)V + \alpha I, & 0 \le \alpha < \beta, \\ I, & \alpha \ge \beta. \end{cases}$$

Following [6], we set $\beta = 10^{-1}$. The reason for this choice is that $\lim_{\alpha\to\infty} V_\alpha = I$ and this feature gives, for large values of $\alpha$, a faster convergence of CG than when $V$ is used. On the other hand, we verified that, on our test problems, using $V_\alpha$ instead of $V$ does not affect the performance of GC for small values of $\alpha$. The `IP` technique has a further algorithmic parameter which identifies the bandwith of the symmetric approximation $H$ in (1.2). In our experiments we considered $H = diag(V_\alpha^T V_\alpha)$ and $H = \tilde{V}_\alpha^T \tilde{V}_\alpha$, where $\tilde{V}_\alpha$ was obtained by extracting the diagonal and the first upper diagonal from $V_\alpha$.

The elapsed time required by the preconditioned CG was measured, in seconds, using the `tic` and `toc` Matlab commands. The time needed to compute the seed preconditioner was not included in the measurements reported in this paper since we were interested in evaluating the performance of our updating strategy given a seed preconditioner, and not the performance of the seed preconditioner itself.

To compare the various preconditioned CG implementations we used the performance profile proposed by Dolan and Moré [15]. Let $\mathcal{S}_{\mathcal{T},\mathcal{A}} \ge 0$ be a statistic corresponding to the successful solution of the test problem $\mathcal{T}$ by the algorithm $\mathcal{A}$ and suppose that the smaller this value is, the better the algorithm is considered. Moreover, let $\mathcal{S}_{\mathcal{T}}$ denote the smallest value attained on the test $\mathcal{T}$ by one of the algorithms under analysis and $\chi_M$ be a value such that $\chi_M > \log_2(\mathcal{S}_{\mathcal{T},\mathcal{A}}/\mathcal{S}_{\mathcal{T}})$ for all $\mathcal{T}$ and $\mathcal{A}$. The performance profile of the algorithm $\mathcal{A}$, in logarithmic scale, is defined as

$$\pi(\chi) = \frac{\text{number of tests such that } \log_2(\mathcal{S}_{\mathcal{T},\mathcal{A}}/\mathcal{S}_{\mathcal{T}}) \le \chi}{\text{number of tests}}, \quad \chi \ge 0,$$

where the ratio $\log_2(\mathcal{S}_{\mathcal{T},\mathcal{A}}/\mathcal{S}_{\mathcal{T}})$ is set equal to $\chi_M$ if the algorithm $\mathcal{A}$ fails in solving the test $\mathcal{T}$. Clearly, $\pi(\chi)$ is the fraction of problems for which $\mathcal{S}_{\mathcal{T},\mathcal{A}}$ is within a factor $2^\chi$ of the smallest value $S_{\mathcal{T}}$. At $\chi = 0$ the performance profile gives the percentage of problems for which the algorithm $\mathcal{A}$ is the best, while the percentage of problems that are successfully solved by the algorithm $\mathcal{A}$ is the value $\lim_{\chi\to\chi_M^-} \pi(\chi)$. Specifically, there exists $\hat{\chi} < \chi_M$ such that the performance profiles flattens in $[\hat{\chi}, \chi_M)$. Then, if $\pi(\chi)$ is plotted in $[0, \chi_M)$, the right side of the plot gives the robustness of the algorithm in solving the test set.

**6.1. Test sets.** A first set of test sequences was built by shifting twenty sparse symmetric positive definite matrices from the University of Florida Sparse Matrix

| Matrix | $n$ | $d_A$ | *drop* | $d_L$ | Matrix | $n$ | $d_A$ | *drop* | $d_L$ |
|---|---|---|---|---|---|---|---|---|---|
| 1138_bus | 1138 | 4.0e-3 | 1.e-1 | 3.8e-3 | cfd1 | 70656 | 3.8e-4 | 1.e-4 | 5.9e-3 |
| apache1 | 80800 | 9.5e-5 | 1.e-1 | 6.8e-5 | crystm03 | 24696 | 1.0e-3 | 1.e-1 | 3.1e-4 |
| bcsstk13 | 2003 | 2.1e-2 | 1.e-3 | 4.1e-2 | gyro_m | 17361 | 1.2e-3 | 1.e-1 | 6.3e-4 |
| bcsstk14 | 1806 | 2.0e-2 | 1.e-1 | 4.1e-3 | jnlbrng1 | 40000 | 1.5e-4 | 1.e-1 | 1.0e-4 |
| bcsstk15 | 3948 | 7.8e-3 | 1.e-1 | 1.5e-3 | kuu | 7102 | 6.9e-3 | 1.e-2 | 5.6e-3 |
| bcsstk16 | 4884 | 1.2e-2 | 1.e-1 | 1.3e-3 | nd3k | 9000 | 4.1e-2 | 1.e-4 | 2.1e-2 |
| bcsstk17 | 10974 | 3.7e-3 | 1.e-3 | 7.6e-3 | s1rmq4m1 | 5489 | 8.9e-3 | 1.e-1 | 2.2e-3 |
| bcsstk18 | 11948 | 1.1e-3 | 1.e-2 | 1.0e-3 | s2rmq4m1 | 5489 | 8.9e-2 | 1.e-1 | 1.8e-3 |
| bcsstk25 | 15439 | 1.1e-3 | 1.e-3 | 3.3e-3 | thermal1 | 82654 | 9.6e-5 | 1.e-1 | 8.2e-5 |
| bcsstk38 | 8032 | 5.6e-3 | 1.e-1 | 9.7e-4 | wathen100 | 30401 | 5.4e-4 | 1.e-1 | 2.9e-4 |

TABLE 6.1
*Characteristics of test problems from the University of Florida Sparse Matrix Collection.*

| Sequence | $d_A$ | *drop* | $d_L$ | $m$ |
|---|---|---|---|---|
| bratu_1 | 1.3e-3 | 1.e-3 | 4.2e-3 | 4 |
| bratu_2 | 1.3e-3 | 1.e-3 | 4.3e-3 | 4 |
| fpm_1 | 1.3e-4 | 5.e-5 | 2.8e-2 | 8 |
| fpm_2 | 1.3e-4 | 5.e-5 | 2.8e-2 | 8 |
| pdexs_1 | 1.3e-3 | 1.e-3 | 4.1e-3 | 6 |
| pdexs_2 | 1.3e-3 | 1.e-3 | 4.1e-3 | 5 |
| pdex1_1 | 1.3e-3 | 1.e-4 | 2.7e-2 | 6 |
| pdex1_2 | 1.3e-3 | 1.e-3 | 4.3e-3 | 6 |

TABLE 6.2
*Characteristics of test problems arising in the application of the RER algorithm.*

Collection [13]. In Table 6.1 we report the dimension $n$ of each matrix, the density $d_A$ of its lower triangular part, the drop tolerance *drop* used to compute its incomplete $LDL^T$ factorization, and the density $d_L$ of the corresponding factor $L$. The density of a triangular matrix is defined as the ratio between the number of nonzero entries and the total number of entries in the triangular part. We remark that $d_L$ is also the density of the preconditioners used by the `UP` and `FP` procedures for a whole sequence of shifted systems. Furthermore, we note that $d_L$ is about the same order of magnitude or smaller than $d_A$, except for `cfd1`.

Each matrix was normalized dividing it by the maximum diagonal entry, to have a largest entry equal to 1. Eleven values of $\alpha$ were used for all the matrices, i.e. $\alpha = 10^{-i}$, $i = 0, 1, \ldots, 5$, and $\alpha = 5 \cdot 10^{-i}$, $i = 1, \ldots, 5$, for a total of 220 linear systems. We point out that, because of the normalization of the matrices, preconditioning was no longer beneficial as soon as $\alpha$ was of order $10^{-1}$ (see also [6]). The right-hand sides of the systems were built to obtain the solution vector $(1, \ldots, 1)^T$.

A second set of test sequences was obtained by applying the Regularized Euclidean Residual (RER) algorithm described in [4] to three systems of nonlinear equations arising from the discretization of PDEs. Specifically, the RER algorithm was implemented using $h = 1$ and $k = 2$ in (2.2), fixing $\sigma = 1$ as initial parameter and updating $\sigma$ along the iterations as in [4]. Eight sequences of shifted linear systems arising in the computation of trial steps were selected. All the matrices of the sequences have dimension $n = 10^4$.

The first of the mentioned nonlinear systems comes from the discretization of the Bratu problem with $\lambda = 6.5$ [30]. We used a zero initial guess and extracted the sequences of shifted linear systems at the first and the second nonlinear iteration of RER; we refer to these sequences as `bratu_1` and `bratu_2`. The second nonlinear system was obtained by discretizing a PDE problem modelling a flow in a porous

medium as described in [30]. We applied the RER algorithm to the nonlinear system with the initial guess given in [30] and extracted the sequences arising in the first (`fpm_1`) and in the second (`fpm_2`) nonlinear iteration. The discretization of the PDE problem given in [18, §3.2.1] produced our third nonlinear system and gave rise to four sequences. The sequences `pdexs_1` and `pdexs_2` were generated at the first and second nonlinear iteration of RER with the standard initial guess given in [18] (with $\kappa = 100$), while the sequences `pdex1_1` and `pdex1_2` arose at the first and the second nonlinear iteration of RER, starting from the vector of all ones.

In Table 6.2 we specify some details of the sequences of linear systems associated with the RER algorithm, i.e. the density $d_A$ of the matrix $J^TJ$, the drop tolerance *drop* used in the incomplete $LDL^T$ factorization of $J^TJ$, the density $d_L$ of the matrix $L$ and the number $m$ of shifted systems forming the sequence. We note that the drop tolerance used with `fmp_1` and `fmp_2` was obtained by dividing by 2, instead of 10, the previous drop tolerance leading to failure, i.e. $10^{-4}$, to avoid an incomplete Cholesky factorization too close to the complete one. For these two problems and `pdex1_1` the small drop tolerance used gave rise to $L$ factors significantly denser than the matrix $J^TJ$.

Concerning the computation of $L$ and $D$, we are aware of approaches that do not need to form the matrix $J^TJ$, such as the incomplete Cholesky factorization proposed in [7], and the Threshold Incomplete Givens Orthogonalization (TIGO) method presented in [2], which provides an incomplete $QR$ factorization of $J$. On the other hand, avoiding the computation of $J^TJ$ was not crucial for our experiments and we did not address it.

**6.2. Comparison with standard preconditioning techniques.** We compare `UP` with the standard `RP` and `FP` techniques, as well as with `NP`, analysing both their robustness and efficiency.

We focus first on the ability of the four strategies to solve a whole sequence of shifted linear systems. The performance profiles in Figure 6.1 concern the total number of CG iterations and the execution times required to solve the whole sequences obtained from the set of matrices in Table 6.1. A failure is declared if the preconditioned CG fails in solving a system of the sequence. As expected, CG with `UP` requires more iterations than CG with `RP`, but it is the fastest among all the solvers. Specifically, `UP` is the most efficient in terms of execution time for 60% of the sequences. On the other hand, it fails in solving the sequence corresponding to the matrix `bcsstk17`, since in this case it fails for $\alpha = 10^{-5}$ and $\alpha = 5 \cdot 10^{-5}$. With the same matrix, `FP` fails in solving all the systems but the one corresponding to $\alpha = 10^{-5}$; it also fails on all the systems associated with `bcsstk18`, `bcsstk25` and `bcsstk38`. Furthermore, the performance of `FP` typically deteriorates as $\alpha$ increases, as confirmed by the iterations and timings profiles. More insight into the robustness of the various strategies can be get by looking at the number of successful solutions over the 20 sequences, which is 20 for `RP`, 19 for `UP`, 12 for `FP` and 11 for `NP`.

To better understand the behaviour of our preconditioning strategy versus the other ones, we analyse also the performance profiles of the CG iterations and the execution times in solving the shifted systems corresponding to three representative values of $\alpha$, i.e. $\alpha = 10^{-5},\ 5 \cdot 10^{-3},\ 1$ (Figures 6.2–6.4). As expected, for each value of $\alpha$, `RP` is the most efficient in terms of CG iterations, but not in terms of time. The performance of `FP` steadily deteriorates while $\alpha$ increases and, for some tests, it already gives poor results with the small value $\alpha = 10^{-5}$; more precisely, `FP` fails in solving 3, 6, 8 systems for $\alpha = 10^{-5},\ 5 \cdot 10^{-3},\ 1$, respectively. The `NP` strategy gives

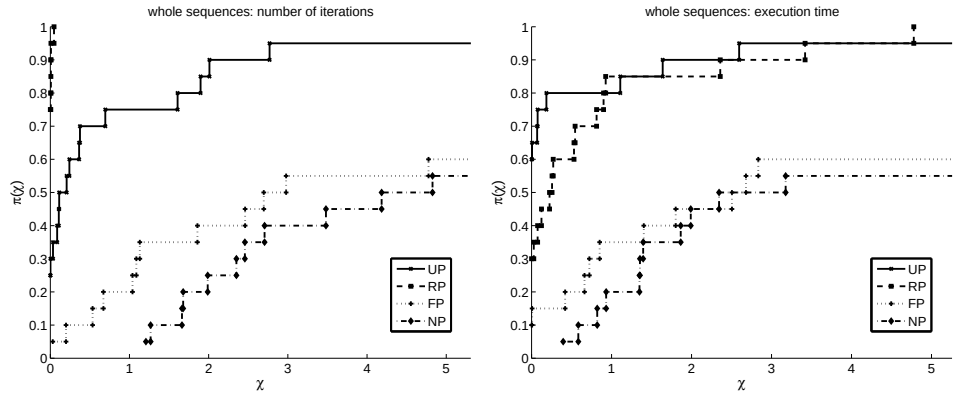


FIG. 6.1. *Performance profiles of* UP, RP, FP *and* NP *on the first test set, for the solution of the whole sequences: number of CG iterations (left) and execution times (right).*

rise to the slowest convergence rate and the largest execution time until $\alpha$ becomes sufficiently large, i.e. when preconditioning is no longer needed. Remarkably, the UP strategy is very effective in reducing the number of CG iterations and provides a good tradeoff between the cost of the linear solver and of the construction of the preconditioner. In particular, our strategy is the clear winner in terms of execution time for $\alpha = 10^{-5}$ and $\alpha = 5 \cdot 10^{-3}$, and shows a gain over RP and FP when $\alpha = 1$. In summary, we see that FP and NP work well for specific values of $\alpha$, RP is expensive in terms of time to solution, while the UP strategy shows a remarkable reliability and efficiency in solving sequences of shifted systems for a fairly broad range of values of $\alpha$.

We conclude the analysis of the first set of experiments reporting, in Tables 6.3, 6.4 and 6.5, the results for the sequences of systems built from the matrices nd3k, s1rmq4m1 and wathen100, respectively. The test problem nd3k is representative of the convergence history of CG with the various preconditioning techniques on several test problems. In this case, NP fails for the first value of $\alpha$, while it gains in iteration count (and time) as $\alpha$ increases; on the contrary, FP deteriorates for increasing values of $\alpha$. The UP strategy generally behaves well for all values of $\alpha$ and is considerably faster than RP; in fact, the time for computing the preconditioner from scratch is relatively high. In terms of execution time, the UP strategy outperforms the unpreconditioned CG as long as $\alpha \leq 10^{-3}$. Concerning the sequence associated with s1rmq4m1, the time for building the preconditioner from scratch for each matrix $A + \alpha I$ is small, so the RP strategy is fast. However, UP yields a number of CG iterations comparable with that of RP and hence has a smaller execution time. More precisely, the relative time saving of UP vs RP varies bewtween 9% and 46%. Furthermore, the execution times of UP and RP compare favourably with the times of FP and NP for all values of $\alpha$. Finally, we observe that for the test problem wathen100 the matrix $A$ and the factor $L$ of the seed preconditioner are much sparser than the matrices $A$ and $L$ of the previous tests, the sequence is not hard to solve by using the NP and FP strategies, and the performance of FP deteriorates slowly for increasing values of $\alpha$. Nevertheless, UP is the fastest strategy in solving the whole sequence and produces savings in terms of iterations and time over FP for $\alpha \geq 10^{-2}$, and over NP for $\alpha \leq 10^{-1}$. Furthermore, UP is as efficient as RP in terms of CG iterations.

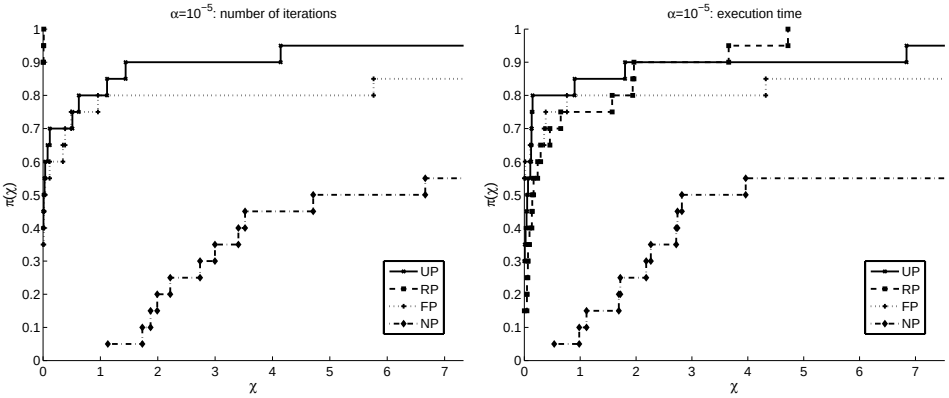


FIG. 6.2. *Performance profiles of* UP, RP, FP *and* NP *on the first test set, for* $\alpha = 10^{-5}$*: number of CG iterations (left) and execution times (right).*

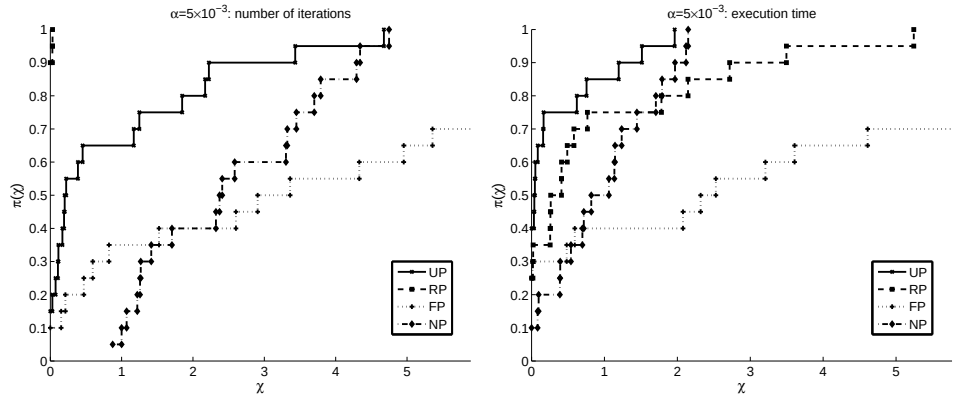


FIG. 6.3. *Performance profiles of* UP, RP, FP *and* NP *on the first test set, for* $\alpha = 5 \times 10^{-3}$*: number of CG iterations (left) and execution times (right).*

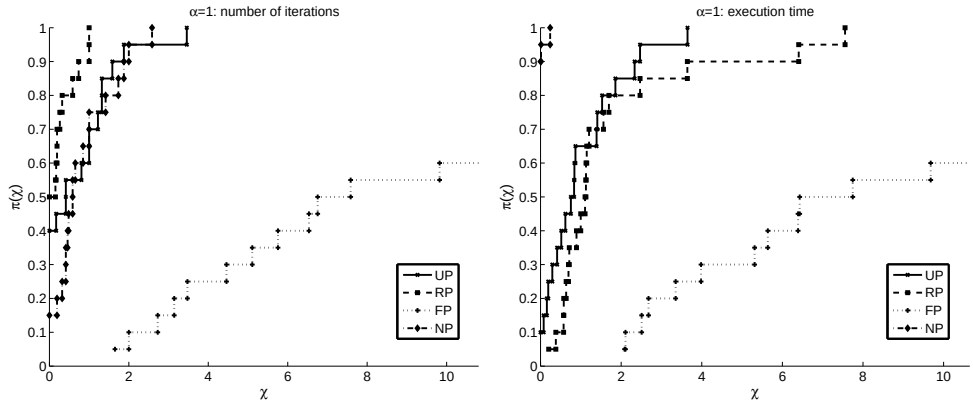


FIG. 6.4. *Performance profiles of* UP, RP, FP *and* NP *on the first test set, for* $\alpha = 1$*: number of CG iterations (left) and execution times (right).*

| | CG iterations | | | | execution time (sec.) | | | |
|---|---|---|---|---|---|---|---|---|
| $\alpha$ | UP | RP | FP | NP | UP | RP | FP | NP |
| 1.e-5 | 67 | 47 | 66 | * | 1.7e0 | 2.1e1 | 1.6e0 | * |
| 5.e-5 | 46 | 20 | 48 | 806 | 1.2e0 | 1.9e1 | 1.2e0 | 1.4e1 |
| 1.e-4 | 44 | 15 | 48 | 553 | 1.1e0 | 1.8e1 | 1.2e0 | 9.8e0 |
| 5.e-4 | 54 | 8 | 68 | 211 | 1.4e0 | 1.5e1 | 1.7e0 | 3.7e0 |
| 1.e-3 | 57 | 7 | 83 | 142 | 1.5e0 | 1.3e1 | 2.0e0 | 2.5e0 |
| 5.e-3 | 54 | 5 | 155 | 50 | 1.4e0 | 1.0e1 | 3.8e0 | 8.9e-1 |
| 1.e-2 | 49 | 4 | 197 | 33 | 1.3e0 | 8.9e0 | 4.8e0 | 6.0e-1 |
| 5.e-2 | 39 | 3 | 375 | 10 | 1.0e0 | 7.1e0 | 9.1e0 | 1.9e-1 |
| 1.e-1 | 31 | 3 | 481 | 6 | 8.2e-1 | 6.5e0 | 1.2e0 | 1.2e-1 |
| 5.e-1 | 15 | 2 | 751 | 2 | 4.2e-1 | 5.4e0 | 1.8e1 | 4.5e-2 |
| 1.e0 | 11 | 2 | 906 | 1 | 3.3e-1 | 5.0e0 | 2.2e1 | 2.7e-2 |

TABLE 6.3

*Results obtained with* UP, RP, FP *and* NP *on the sequence of shifted systems associated with* nd3k *(* denotes a CG failure).*

| | CG iterations | | | | execution time (sec.) | | | |
|---|---|---|---|---|---|---|---|---|
| $\alpha$ | UP | RP | FP | NP | UP | RP | FP | NP |
| 1.e-5 | 153 | 154 | 155 | * | 4.1e-1 | 4.5e-1 | 4.2e-1 | * |
| 5.e-5 | 103 | 103 | 109 | 937 | 2.8e-1 | 3.1e-1 | 2.9e-1 | 1.9e0 |
| 1.e-4 | 86 | 86 | 96 | 730 | 2.3e-1 | 2.7e-1 | 2.5e-1 | 1.5e0 |
| 5.e-4 | 58 | 54 | 110 | 389 | 1.6e-1 | 1.8e-1 | 2.9e-1 | 7.8e-1 |
| 1.e-3 | 49 | 45 | 131 | 295 | 1.3e-1 | 1.6e-1 | 3.5e-1 | 5.9e-1 |
| 5.e-3 | 32 | 28 | 170 | 145 | 9.0e-2 | 1.1e-1 | 4.5e-1 | 2.9e-1 |
| 1.e-2 | 27 | 24 | 180 | 109 | 7.7e-2 | 9.5e-2 | 4.8e-1 | 2.2e-1 |
| 5.e-2 | 18 | 21 | 183 | 51 | 5.3e-2 | 8.2e-2 | 4.9e-1 | 1.1e-1 |
| 1.e-1 | 14 | 20 | 187 | 36 | 4.3e-2 | 8.0e-2 | 5.0e-1 | 7.5e-2 |
| 5.e-1 | 9 | 12 | 312 | 16 | 3.0e-2 | 4.9e-2 | 8.2e-1 | 3.5e-2 |
| 1.e0 | 7 | 8 | 378 | 11 | 2.5e-2 | 3.7e-2 | 9.9e-1 | 2.5e-2 |

TABLE 6.4

*Results obtained with* UP, RP, FP *and* NP *on the sequence of shifted systems associated with* s1rmq4m1 *(* denotes a CG failure).*

| | CG iterations | | | | execution time (sec.) | | | |
|---|---|---|---|---|---|---|---|---|
| $\alpha$ | UP | RP | FP | NP | UP | RP | FP | NP |
| 1.e-5 | 20 | 20 | 20 | 212 | 1.7e-1 | 2.4e-1 | 1.5e-1 | 1.0e0 |
| 5.e-5 | 20 | 20 | 20 | 209 | 1.7e-1 | 2.4e-1 | 1.5e-1 | 1.0e0 |
| 1.e-4 | 20 | 20 | 20 | 206 | 1.7e-1 | 2.4e-1 | 1.5e-1 | 9.9e-1 |
| 5.e-4 | 20 | 20 | 20 | 174 | 1.7e-1 | 2.4e-1 | 1.5e-1 | 8.4e-1 |
| 1.e-3 | 20 | 20 | 20 | 146 | 1.7e-1 | 2.4e-1 | 1.5e-1 | 7.1e-1 |
| 5.e-3 | 20 | 19 | 22 | 95 | 1.7e-1 | 2.4e-1 | 1.7e-1 | 4.6e-1 |
| 1.e-2 | 19 | 19 | 24 | 76 | 1.6e-1 | 2.4e-1 | 1.8e-1 | 3.7e-1 |
| 5.e-2 | 16 | 16 | 35 | 42 | 1.4e-1 | 2.1e-1 | 2.7e-1 | 2.0e-1 |
| 1.e-1 | 14 | 14 | 43 | 32 | 1.2e-1 | 1.9e-1 | 3.3e-1 | 1.6e-1 |
| 5.e-1 | 10 | 10 | 71 | 16 | 9.7e-2 | 1.4e-1 | 5.3e-1 | 7.9e-2 |
| 1.e0 | 8 | 10 | 89 | 12 | 8.0e-2 | 1.2e-1 | 6.6e-1 | 6.5e-2 |

TABLE 6.5

*Results obtained with* UP, RP, FP *and* NP *on the sequence of shifted systems associated with* wathen100.

| | total CG iterations | | | | total execution time (sec.) | | | |
|---|---|---|---|---|---|---|---|---|
| Sequence | UP | RP | FP | NP | UP | RP | FP | NP |
| bratu_1 | 117 | 66 | 300 | * | 5.1e-1 | 1.2e0 | 1.1e1 | * |
| bratu_2 | 126 | 110 | 187 | * | 3.6e-1 | 1.6e0 | 5.5e-1 | * |
| fpm_1 | 443 | 103 | * | * | 5.9e0 | 5.6e0 | * | * |
| fpm_2 | 671 | 100 | 5250 | * | 8.6e0 | 5.9e0 | 6.2e1 | * |
| pdexs_1 | 114 | 59 | 586 | * | 5.5e-1 | 1.5e0 | 2.3e0 | * |
| pdexs_2 | 127 | 60 | 314 | * | 5.9e-1 | 1.4e0 | 1.2e0 | * |
| pdex1_1 | * | 209 | * | * | * | 5.3e0 | * | * |
| pdex1_2 | 96 | 57 | 850 | * | 4.8e-1 | 1.5e0 | 3.4e0 | * |

TABLE 6.6
*Results obtained with* UP, RP, FP *and* NP *on the second test set (* denotes a CG failure).*

The results of our experiments on the second set of sequences are summarized in Table 6.6, where the total number of CG iterations and the total execution times for solving all the systems of each sequence are reported. As for the first test set, we declare a failure when the preconditioned CG fails in solving at least one system of the sequence. We note that these sequences really need a preconditioning strategy; in fact, the unpreconditioned CG fails in solving the first linear system of each sequence, corresponding to the smallest value of $\alpha$. For five of the eight sequences, the UP strategy turns out to be the most efficient. Actually, for these sequences RP is the most effective in terms of CG iterations, but it requires a significantly larger execution time than UP, because of the computation of the incomplete $LDL^T$ factorization for each value of $\alpha$. The sequence pdex1_1 is hard to solve, as only the RP strategy succeeds in its solution; our update strategy fails in solving the first linear system of the sequence. We observe also that on the fpm_1 and fpm_2 sequences the UP strategy requires a large number of CG iterations compared to the RP one. Moreover, the small drop tolerance $5 \cdot 10^{-5}$ that is needed for computing an effective incomplete Cholesky preconditioner, produces a relatively dense factor $L$, hence the time required for applying the preconditioner, i.e. for solving the related triangular systems at each CG iteration, is significant. Therefore, for these two sequences, the considerable gain in terms of iterations obtained by the RP strategy offsets the cost of the computation of the preconditioner from scratch and makes RP the most efficient in the solution of these two sequences. The FP strategy is generally worse than UP and RP.

**6.3. Comparison with update techniques.** Now we focus on the comparison among the UP, MP and IP update strategies.

In accordance with the numerical experiments presented in [25], we verified that, in applying MP to the selected test problems, there is no significant reduction of CG iterations using the order 1 over the order 0 expansion. As a consequence, the order 0 strategy is generally faster than the order 1 strategy. For this reason, we do not report the results obtained with MP.

Concerning the IP strategy, we found that the two choices of the matrix $H$ give practically the same CG iteration counts, while using diagonal $H$ requires less execution time; this comment on the performance of IP was also raised in [6]. Hence, we report only the results obtained with diagonal $H$.

As in section 6.2, UP and IP are compared in terms of CG iterations and execution times. However, we are aware that there are strong differences in the seed preconditioners used by the two strategies and this affects the number of iterations performed with the updated preconditioners and their costs. Nevertheless, UP and IP were applied to the test sequences using the same stopping criterion and accuracy re-

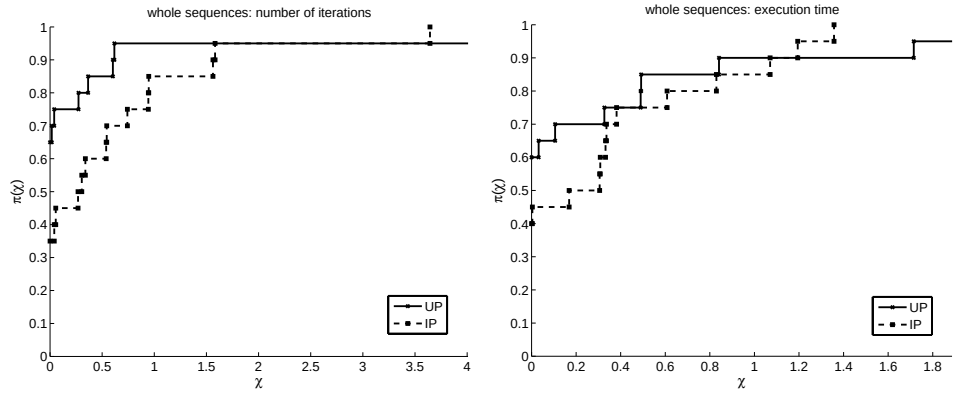


FIG. 6.5. *Performance profiles of* `UP` *and* `IP` *on the first test set, for the solution of the whole sequences: number of CG iterations (left) and execution times (right).*

quirement in the solution of the linear systems; thus, the comparison is not misleading as far as we evaluate the ability of the two different algorithms to solve a sequence within a prescribed level of accuracy.

We restrict our analysis to the first set of test sequences, since the construction of the approximate inverse seed preconditioner is very convenient in this case, while it is hard for the second set. Specifically, for the first test set, the SAINV seed preconditioner was built using $10^{-1}$ as drop tolerance for 18 out of 20 matrices, and $10^{-2}$ for the remaining two matrices, always obtaining a factor $V$ with a comparable or smaller density than $A$. For the second test set, a seed preconditioner could be formed for only 2 matrices out of 8. For `bratu_1, bratu_2, pdexs_1, pdexs_2, pdex1_1, pdex1_2`, a seed preconditioner with a too dense factor $V$ was needed to successfully solve $Ax = b$, and we could not compute it because of memory limitations.

The performance profiles in Figure 6.5 show the number of CG iterations and the execution times required to solve the whole sequences. We see that the `IP` strategy solves all the sequences and that it is slightly more expensive than the `UP` one. In particular, there is an evident gain of `UP` over `IP` in terms of CG iterations. This behaviour can be partially ascribed to the slower convergence of `IP` corresponding to the largest values of $\alpha$; in fact, although the modified matrix $V_\alpha$ is used, the number of CG iterations is not steadily decreasing for increasing values of $\alpha$. Concerning the execution time, `UP` is the fastest for 60% of the problems; however, the timings of `IP` are within a factor 2 of those of `UP` for 85% of the tests. The improved behaviour of `IP` in terms of time depends on the fact that the application of the preconditioner does not require solving linear systems and that the multiplicative application of the inverse preconditioners is cheap because of their high sparsity.

**7. Conclusions and future work.** We derived a preconditioning technique for the iterative solution of sequences of symmetric positive definite shifted linear systems. For each matrix $A + \alpha I$, a preconditioner was obtained by a suitable update of a preconditioner for $A$, available in the $LDL^T$ form. This update is low cost, preserves the sparsity pattern of the initial factors and is very easy to be implemented. A theoretical analysis shows that this technique provides preconditioners that are good approximations of the corresponding shifted matrices and it is able to mimic the behaviour of the matrix sequence not only for small values of $\alpha$, but also when $\alpha$

increases. Numerical experiments on several test problems confirm the theoretical results and show that our approach is more efficient than freezing the preconditioner or recomputing it from scratch.

Furthermore, we showed the equivalence between our strategy and Meurant's order 0 preconditioning strategy. As a consequence, we provided a theoretical understanding of the latter strategy.

Future work will be devoted to extending our procedure to sequences of symmetric indefinite matrices which differ by a diagonal matrix, such as the KKT matrices arising in interior point methods for linear and quadratic programming. We will also investigate preconditioner update strategies where both the matrices $L$ and $D$ are allowed to change.

**Acknowledgments.** We wish to thank the anonymous referees for their careful suggestions and comments. We are especially grateful to Prof. Gerard Meurant for making available Matlab implementations of the preconditioning techniques in [25], which helped us to understand the equivalence between our update procedure and his order 0 approximation procedure.

## REFERENCES

[1] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM, 1998.

[2] Z.-Z. Bai, I. S. Duff, and A. J. Wathen, *A Class of Incomplete Orthogonal Factorization Methods. I: Methods and Theories*, BIT Numerical Mathematics, 41 (2001), pp. 53–70.

[3] S. Bellavia, D. Bertaccini, and B. Morini, *Nonsymmetric preconditioner updates in Newton-Krylov methods for nonlinear systems*, Report 10/01, Dipartimento di Energetica "S.Stecco", Università di Firenze, 2010.

[4] S. Bellavia, C. Cartis, N. I. M. Gould, B. Morini, and Ph. L. Toint, *Convergence of a Regularized Euclidean Residual Algorithm for Nonlinear Least-Squares*, SIAM J. Num. Anal., 48 (2010), pp. 1–29.

[5] M. Benzi, J. K. Cullum, and M. Tůma, *Robust approximate inverse preconditioning for the conjugate gradient method*, SIAM J. Sci. Comput., 22 (2000), pp. 1318–1332.

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

[7] M. Benzi and M. Tůma, *A Robust Preconditioner with Low Memory Requirements for Large Sparse Least Squares Problems*, SIAM J. Sci. Comput., 25 (2003), pp. 499–512.

[8] M. Benzi and M. Tůma, *A Robust Incomplete Factorization Preconditioner for Positive Definite Matrices*, Numer. Linear Algebra with Appl., 10 (2003), pp. 385–400.

[9] D. Bertaccini, *Efficient preconditioning for sequences of parametric complex symmetric linear systems*, ETNA, 18 (2004), pp. 49–64.

[10] C. Calgaro, J. P. Chehab, and Y. Saad, *Incremental incomplete LU factorizations with applications*, Numer. Linear Algebra with Appl., 17 (2010), pp. 811–837.

[11] C. Cartis, N. I. M. Gould, and Ph. L. Toint, *Adaptive cubic overestimation methods for unconstrained optimization. Part I: motivation, convergence and numerical results*, Math. Prog. A, to appear.

[12] C. Cartis, N. I. M. Gould, and Ph. L. Toint, *Trust-region and other regularisations of linear least-squares problems*, BIT, 49 (2009), pp. 21-53.

[13] T. A. Davis and Y. F. Hu, *The University of Florida sparse matrix collection*, submitted to ACM Trans. Math. Soft., 2010, http://www.cise.ufl.edu/research/sparse/matrices.

[14] J. E. Dennis, Jr., and R. B. Schnabel *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, 1996.

[15] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with performance profiles*, Math. Prog., 91 (2002), pp. 201-213.

[16] J. Duintjer Tebbens and M. Tůma, *Efficient Preconditioning of Sequences of Nonsymmetric Linear Systems*, SIAM J. Sci. Comput., 29 (2007), pp. 1918–1941.

[17] J. Duintjer Tebbens and M. Tůma, *Preconditioner Updates for Solving Sequences of Linear Systems in Matrix-free Environment*, Numer. Linear Algebra with Appl., 17 (2010),

pp. 997–1019.
[18] S. C. Eisenstat and H. F. Walker, *Choosing the forcing term in an inexact Newton method*, SIAM J. Sci. Comput., 17 (1996), pp. 16–32.
[19] J. B. Erway and P. E. Gill, *A Subspace Minimization Method for the Trust-Region Step*, SIAM J. Optim., 20 (2009), pp. 1439–1461.
[20] G. H. Golub and C. F. Van Loan, *Matrix Computations*, third edition, The Johns Hopkins University Press, 1996.
[21] J. Gondzio, *Matrix-Free Interior Point Method*, Technical Report ERGO-09-012, School of Mathematics, The University of Edinburgh, revised 29 April 2010.
[22] N. I. M. Gould, D. P. Robinson and H. S. Thorne, *On solving trust-region and other regularised subproblems in optimization*, Math. Program. Comput., 2 (2010), pp. 21–57.
[23] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, 1991.
[24] C. J. Lin and J. Moré, *Incomplete Cholesky factorizations with limited memory*, SIAM J. Sci. Comput., 21 (1999), pp. 24–45.
[25] G. Meurant, *On the incomplete Cholesky decomposition of a class of perturbed matrices*, SIAM J. Sci. Comput., 23 (2001), pp. 419–429.
[26] J. J. Moré and D. C. Sorensen, *Computing a trust-region step*, SIAM J. Sci. Stat. Comput., 4 (1983), pp. 553–572.
[27] Y. Nesterov, *Modified Gauss-Newton scheme with worst-case guarantees for global performance*, Optim. Methods Softw., 22 (2007), pp. 469–483.
[28] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operation Research, Springer-Verlag, 1999.
[29] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Soft., 8 (1982), pp. 43–71.
[30] M. Pernice and H. F. Walker, *NITSOL: a new iterative solver for nonlinear systems*, SIAM J. Sci. Comput., 19 (1998), pp. 302–318.
[31] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd edition, SIAM, 2003.
[32] M. Tůma, *Sparselab*, available at http://www2.cs.cas.cz/~tuma/sparslab.html.