GROUPE D'ÉTUDES ET DE RECHERCHE
EN ANALYSE DES DÉCISIONS

Les Cahiers du GERAD

G-2013-87
December 2013

CITATION ORIGINALE / ORIGINAL CITATION

Orban, D., Limited-memory LDL^T factorization of symmetric quasi-definite matrices, Numerical Algorithms, 70(1), 9-41, October 2014.
Doi: 10.1007/s11075-014-9933-x

GERAD HEC Montréal
3000, ch. de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Les Cahiers du GERAD ISSN: 0711–2440

Limited-Memory LDL^T Factorization of Symmetric Quasi-Definite Matrices

D. Orban

G–2013–87

December 2013

Limited-Memory LDL^T Factorization of Symmetric Quasi-Definite Matrices

Dominique Orban

GERAD & Department of Mathematics and Industrial Engineering
Polytechnique Montréal
Montréal (Québec) Canada, H3C 3A7

dominique.orban@gerad.ca

December 2013

Les Cahiers du GERAD

G–2013–87

Abstract: We propose a generalization of the limited-memory Cholesky factorization of Lin and Moré (1999) to the symmetric indefinite case with special interest in symmetric quasi-definite matrices. We use this incomplete factorization to precondition two formulations of linear systems arising from regularized interior-point methods for quadratic optimization. An advantage of the limited-memory approach is predictable memory requirements. We establish existence of incomplete factors when the input matrix is an H-matrix but our numerical results illustrate that the factorization succeeds more generally. An appropriate diagonal shift is applied whenever the input matrix is not quasi definite. As the memory parameter increases an efficiency measure of the preconditioner suggested by Scott and Tůma (2013) improves. The combination of the 3×3 block formulation analyzed by Greif, Moulding, and Orban (2012), the SYMAMD ordering, and a moderate memory parameter results in encouraging performance.

Key Words: Preconditioning, Symmetric Quasi-Definite, Incomplete Factorization, Limited-Memory Factorization, Interior-Point Methods.

Résumé : Nous proposons une généralisation de la factorisation de Cholesky incomplète de Lin and Moré (1999) au cas symétrique indéfini en nous penchant particulièrement sur les matrices symétriques et quasi définies. Cette factorisation incomplète est utilisée pour préconditionner deux formulations des systèmes linéaires provenant des méthodes de points intérieurs régularisées pour l'optimisation quadratique. Un avantage de l'approche à mémoire limitée est que l'on peut prévoir à l'avance la quantité de mémoire nécessaire. L'existence d'une factorisation incomplète est établie sous l'hypothèse que la matrice initiale est une H-matrice. Nos résultats numériques suggèrent cependant que la factorisation se termine avec succès même en l'absence de cette propriété. Un décalage diagonal est appliqué lorsque la matrice initiale n'est pas quasi définie. Lorsque le paramètre de mémoire augmente, une mesure d'efficacité du préconditionneur proposée par Scott and Tůma (2013) augmente elle aussi. On obtient une performance encourageante en combinant la formulation bloc 3×3 analysée par Greif, Moulding, and Orban (2012), l'ordonnancement SYMAMD et une valeur modeste du paramètre de mémoire.

Acknowledgments: Research partially supported by an NSERC Discovery Grant. We express our sincere gratitude to Chen Greif for valuable comments and enjoyable discussions that lead to substantial improvements in the contents and presentation of this paper.

1 Introduction

Symmetric quasi-definite (SQD) matrices are symmetric permutations of a matrix of the form

$$\begin{bmatrix} \mathbf{E} & \mathbf{C}^\mathsf{T} \\ \mathbf{C} & -\mathbf{F} \end{bmatrix}, \tag{1.1}$$

where $\mathbf{E} = \mathbf{E}^\mathsf{T}$ and $\mathbf{F} = \mathbf{F}^\mathsf{T}$ are positive definite. Definite matrices are special cases of SQD matrices corresponding to either $\mathbf{E}$ or $\mathbf{F}$ being vacuous. SQD matrices arise in a number of practical applications, among which interior-point methods for nonlinear optimization (Friedlander and Orban, 2012; Greif et al., 2012) and the solution of stabilized Stokes problems (Elman, Silvester, and Wathen, 2005). Vanderbei (1995) established that they have a very specific structure beyond symmetry and indefiniteness that sets them apart from classical saddle-point matrices with $\mathbf{F} = \mathbf{0}$. Chiefly, they are *strongly factorizable*, i.e., every symmetric permutation leads to an LDL^T factorization without resort to block 2×2 pivots. Gill, Saunders, and Shinnerl (1996) provide stability results on this type of factorization. In addition, SQD matrices are always nonsingular so that the $\mathbf{D}$ factor is nonsingular.

The main idea of this paper is that if $\mathbf{A}$ is SQD and $\mathbf{L}$ and $\mathbf{D}$ are its exact factors, the preconditioned operator

$$|\mathbf{D}|^{-1/2}\mathbf{L}^{-1}\mathbf{A}\mathbf{L}^{-\mathsf{T}}|\mathbf{D}|^{-1/2} = |\mathbf{D}|^{-1}\mathbf{D},$$

where $|\mathbf{D}|$ is the diagonal matrix whose elements are the absolute values of those of $\mathbf{D}$, possesses only two distinct eigenvalues: $+1$ and -1. The hope is therefore that limited-memory factors have the potential to yield a preconditioned system with favorable spectral structure while keeping the computational effort reasonable. When $\mathbf{L}$ and $\mathbf{D}$ are limited-memory factors of $\mathbf{A}$, the preconditioned operator remains symmetric and indefinite so that it is possible to use MINRES or SYMMLQ (Paige and Saunders, 1975).

Our main result states that if $\mathbf{A}$ is an SQD H-matrix, a limited-memory LDL^T factorization always exists in which only the $n_k + p$ largest elements of $\mathbf{L}$ are retained in column k, where n_k is the number of nonzero elements in the strict lower triangle of the k-th column of $\mathbf{A}$ and where $p \in \mathbb{N}$ is a limited-memory factor specified by the user. Note that this is quite different from specifying a drop tolerance. A diagonal update strategy modifies the input matrix to encourage the H-matrix property to hold.

The main message from our numerical experiments is that our LDL^T preconditioner exhibits very encouraging performance in MINRES when combined with the symmetric AMD ordering of Amestoy, Davis, and Duff (1996) and a memory parameter around 10. Our implementation is a modification of that of Lin and Moré (1999) with an updated object-oriented MATLAB interface. It is available from `github.com/optimizers/lldl`.

This research is strongly inspired by the limited-memory Cholesky factorization of Lin and Moré (1999) and is a natural generalization to SQD matrices. In particular, if the input matrix is positive definite, our incomplete factorization coincides with that of Lin and Moré (1999) with the exception that an LDL^T factorization is produced instead of an LL^T factorization. It differs from previous approaches in that it attempts to exploit the SQD structure specifically. The factorization of Lin and Moré (1999) applied to an SQD matrix would induce far too large a perturbation of the matrix to yield a useful preconditioner since it would compute $\alpha > 0$ large enough so that $\mathbf{A} + \alpha\mathbf{I}$ is sufficiently positive definite, and compute an incomplete factorization of the latter matrix. Other factorizations, e.g., based on block pivoting are attractive but ignore the inherent strong factorizability property.

Limited-memory factorizations may not always be the best choice for problems arising from the discretization of PDEs, where preconditioners based on the underlying functional spaces and mesh are often preferred. We do not investigate those issues here and concentrate on algebraic problems arising from optimization applications.

This paper is organized as follows. The rest of this section covers related research and notation. Section 2 provides background results on SQD, M- and H-matrices. Section 3 describes our limited-memory LDL^T factorization. Section 4 is a brief discussion on the stability of the incomplete LDL^T factorization of SQD M-and H-matrices. Section 5 describes our implementation in relation to that of Lin and Moré (1999). Section 6 provides background on interior-point methods for constrained optimization and the form of the systems encountered in a typical implementation. Section 7 reports numerical results on systems of the form described in §6. We briefly mention extensions and the applicability of our limited-memory factorization to more general symmetric indefinite, but not SQD, matrices in §8. We conclude and give potential research avenues for the future in §9.

1.1 Related Research

Numerous incomplete factorization schemes have been proposed in the past, especially in the context of non-symmetric systems (Meijerink and van der Vorst, 1977; Gustafsson, 1978), and enforce a fixed level of fill or control the fill by way of a parameter. They either have unpredictable memory requirements or enforce a specific structure in the factors independently of the entries. Drop tolerance strategies, such as that of Munksgaard (1980) have similar shortcomings. The ILUT factorization of Saad (1994) has predictable memory requirements but typically does not produce a symmetric factorization even when the input matrix is symmetric.

The incomplete factorization proposed in the present paper is a generalization of the incomplete Cholesky factorization of Lin and Moré (1999), which itself improves over that of Jones and Plassmann (1995) by allowing an additional amount of fill controlled via a memory parameter. The main benefit of both is their predictable memory requirements. They are based on a shift strategy proposed by Manteuffel (1979, 1980). For a given symmetric matrix $\mathbf{A}$, they compute and incomplete lower triangular factor $\mathbf{L}$ such that $\mathbf{L}\mathbf{L}^{\mathsf{T}} \approx \mathbf{A} + \alpha\mathbf{I}$ for a shift $\alpha \geq 0$ chosen to ensure that $\mathbf{A} + \alpha\mathbf{I}$ is sufficiently positive definite. If $\mathbf{A}$ is already sufficiently positive definite, $\alpha = 0$ and as the memory parameter increases, the incomplete factor $\mathbf{L}$ converges to the exact Cholesky factor of $\mathbf{A}$. If n_j is the number of nonzero entries in the lower triangular part of the j-th column of $\mathbf{A}$, Lin and Moré (1999) retain the $n_j + p$ largest entries in the j-th column of $\mathbf{L}$, where $p \in \mathbb{N}$ is a user-specified memory parameter. The Jones and Plassmann (1995) factorization corresponds to $p = 0$.

Scott and Tůma (2013) also build upon the incomplete Cholesky factorization of Lin and Moré (1999) but require additional memory during the computation of the factors that is later discarded, following an approach suggested by Tismenetsky (1991) and Kaporin (1998). The input matrix $\mathbf{A}$ is shifted so as to be sufficiently positive definite.

Greif and Liu (2013) propose an incomplete factorization of general symmetric indefinite and skew-symmetric matrices. Though their software, named SYM-iLDL, also refers to the factorization as $\mathrm{LDL}^{\mathsf{T}}$, their factorization is inspired by that of Bunch and Kaufman (1977) and $\mathbf{D}$ is block diagonal with blocks of size 1×1 and 2×2. Their implementation builds upon ideas proposed by Li, Saad, and Chow (2003) and Li and Saad (2005). As the memory parameter increases, the incomplete factors converge to the exact factors.

To the best of our knowledge, no incomplete factorization specifically aimed at the SQD structure has been proposed so far.

1.2 Notation

Matrices and vectors are denoted by uppercase and lowercase latin letters typeset in bold math font, respectively. Capital Latin letters in standard font are reserved for index sets.

The i-th component of a vector $\mathbf{x}$ is x_i. Vector inequalities are understood componentwise, so $\mathbf{x} > \mathbf{0}$ is equivalent to $x_i > 0$ for all indices i. The notation $\mathrm{diag}(\mathbf{x})$ refers to the diagonal matrix whose diagonal entries are the components x_i of $\mathbf{x}$.

If $\mathbf{K}$ is a n-by-n matrix, $I \subseteq \{1, \dots, n\}$ and $J \subseteq \{1, \dots, n\}$, the submatrix of $\mathbf{K}$ whose row indices are in I and column indices are in J is denoted $\mathbf{K}(I, J)$. The complements of I and J in $\{1, \dots, n\}$ are denoted $\bar{I}$ and $\bar{J}$ respectively. For such an index set I and provided $\mathbf{K}(I, I)$ is nonsingular, the Schur complement of $\mathbf{K}(I, I)$ in $\mathbf{K}$ is the matrix $\mathbf{K}(\bar{I}, \bar{I}) - \mathbf{K}(\bar{I}, I)\mathbf{K}(I, I)^{-1}\mathbf{K}(I, \bar{I})$, denoted $\mathbf{K}/\mathbf{K}(I, I)$ for short.

If $\mathbf{K}$ is a n-by-n matrix and $\mathbf{S}$ is a nonsingular principal submatrix of $\mathbf{K}$, the Schur complement of $\mathbf{S}$ in $\mathbf{K}$ is simply denoted $\mathbf{K}/\mathbf{S}$.

Throughout this paper, the terminology LDL^T refers to the factorization of a symmetric, not necessarily definite, matrix $\mathbf{K}$ into the product $\mathbf{LDL}^\mathsf{T}$ where $\mathbf{L}$ is unit lower triangular and $\mathbf{D}$ is diagonal.

The absolute value of a vector $|\mathbf{x}|$ or of a matrix $|\mathbf{A}|$ is the vector or matrix whose elements are the absolute values of the elements of the original vector or matrix. The spectral condition number of $\mathbf{A}$ is denoted $\kappa_2(\mathbf{A})$.

2 Background

We begin with a few definitions and properties.

Definition 2.1 *A matrix $\mathbf{K}$ is symmetric and quasi definite (*SQD*) if $\mathbf{K} = \mathbf{K}^\mathsf{T}$ and there exists a permutation matrix $\mathbf{P}$ such that $\mathbf{P}^\mathsf{T}\mathbf{KP}$ may be partitioned as (1.1) where $\mathbf{E} = \mathbf{E}^\mathsf{T}$ and $\mathbf{F} = \mathbf{F}^\mathsf{T}$ are positive definite.*

Vanderbei (1995) establishes that SQD matrices are always nonsingular and have an SQD inverse. In addition, SQD matrices have the following important property of strong factorizability.

Theorem 2.2 (Vanderbei, 1995) *If $\mathbf{K}$ is* SQD*, the matrix $\mathbf{P}^\mathsf{T}\mathbf{KP}$ possesses an* LDL^T *factorization without pivoting for any permutation matrix $\mathbf{P}$.*

Symmetric indefinite matrices are not strongly factorizable in general. We note however that Tůma (2002) establishes that saddle-point matrices possess an LDL^T factorization for a well-chosen symmetric permutation.

The following result will be useful. Its proof follows from the proof of (Vanderbei, 1995, Theorem 2.1) and basic properties of the Schur complement. Our inspiration for this result is the proof of (George and Ikramov, 2000, Theorem 3.1).

Lemma 2.3 *Suppose $\mathbf{K}$ is an* SQD *matrix and $\mathbf{S}$ is a principal submatrix of $\mathbf{K}$. Then both $\mathbf{S}$ and $\mathbf{K}/\mathbf{S}$ are* SQD.

Proof. That $\mathbf{S}$ is SQD is established in the proof of (Vanderbei, 1995, Theorem 2.1). Observe now that if $\mathbf{S}$ is a principal submatrix of $\mathbf{K}$, then $(\mathbf{K}/\mathbf{S})^{-1}$ is a principal submatrix of $\mathbf{K}^{-1}$. Since the latter is SQD, so is $(\mathbf{K}/\mathbf{S})^{-1}$ by the first part of the lemma, and therefore, so is $\mathbf{K}/\mathbf{S}$. □

We will make use of a special case of Lemma 2.3 stated as the next result.

Corollary 2.4 *Suppose that the n-by-n matrix*

$$\begin{bmatrix} \alpha & \mathbf{w}^\mathsf{T} \\ \mathbf{w} & \mathbf{B} \end{bmatrix} \tag{2.1}$$

is SQD*, where $\alpha \in \mathbb{R}$, $\mathbf{w} \in \mathbb{R}^{n-1}$ and $\mathbf{B} = \mathbf{B}^\mathsf{T}$ is an $(n-1)$-by-$(n-1)$ matrix. Then the Schur complement $\mathbf{B} - \alpha^{-1}\mathbf{w}\mathbf{w}^\mathsf{T}$ is an* SQD *matrix.*

Definition 2.5 *A nonsingular real square matrix $\mathbf{A} = (a_{ij})$ is called an M-matrix if $a_{ij} \le 0$ for $i \ne j$ and $\mathbf{A}^{-1} \ge 0$, i.e., all entries of $\mathbf{A}^{-1}$ are nonnegative.*

Definition 2.6 *A real square matrix* $\mathbf{A} = (a_{ij})$ *is called an H-matrix if the* comparison matrix $\mathcal{M}(\mathbf{A}) = (m_{ij}(\mathbf{A}))$ *defined as*

$$m_{ij}(\mathbf{A}) = \begin{cases} |a_{ij}| & \text{if } i = j, \\ -|a_{ij}| & \text{if } i \neq j, \end{cases}$$

is an M-matrix.

Lemma 2.7 *Let* $\mathbf{K}$ *be an* SQD *matrix that is strictly diagonally dominant. Then* $\mathbf{K}$ *is an H-matrix.*

Proof. The result follows directly by applying (Axelsson, 1994, Lemma 6.4) to $\mathcal{M}(\mathbf{K})$. □

Other properties of M-matrices and H-matrices are used by Lin and Moré (1999) and stated in (Axelsson, 1994). We restate them here for reference.

Lemma 2.8 *Let* $\mathbf{A} = (a_{ij})$ *be a square matrix. Then*

1. *If* $a_{ij} \leq 0$ *for* $i \neq j$, $\mathbf{A}$ *is an M-matrix if and only if there exists* $\mathbf{x} > \mathbf{0}$ *such that* $\mathbf{Ax} > \mathbf{0}$.
2. $\mathbf{A}$ *is an H-matrix if and only if* $\mathbf{A}$ *is generalized strictly diagonally dominant, i.e., there exists* $\mathbf{x} > \mathbf{0}$ *such that* $\mathbf{A}\,\mathrm{diag}(\mathbf{x})$ *is strictly diagonally dominant.*
3. *If* $\mathbf{A}$ *is an M-matrix and* $\mathbf{B} \geq \mathbf{A}$ *is such that* $b_{ij} \leq 0$ *for* $i \neq j$, *then* $\mathbf{B}$ *is an M-matrix.*
4. *If* $\mathbf{A}$ *is an M-matrix, Any symmetric permutation of* $\mathbf{A}$ *is an M-matrix.*
5. *If* $\mathbf{A}$ *is an M-matrix, so is the Schur complement of any principal submatrix of* $\mathbf{A}$.

Proof. Items 1–3 appear as Lemma 6.4 and Remark 6.8 in (Axelsson, 1994).

Let $\mathbf{A}$ be an M-matrix. Item 1 shows that there exists $\mathbf{x} > \mathbf{0}$ such that $\mathbf{Ax} > \mathbf{0}$. Let $\mathbf{P}$ be a permutation matrix. Then $\mathbf{y} := \mathbf{P}^\top \mathbf{x} > \mathbf{0}$ and $\mathbf{P}^\top \mathbf{APy} = \mathbf{P}^\top \mathbf{Ax} > \mathbf{0}$ because this is the permutation of a positive vector. Item 1 then shows that $\mathbf{P}^\top \mathbf{AP}$ is an M-matrix.

Item 5 follows from Item 4 and (Axelsson, 1994, Theorem 6.10). □

3 Incomplete LDL$^\top$ Factorization

For a given n-by-n symmetric matrix $\mathbf{A} = (a_{ij})$, the LDL$^\top$ factorization attempts to compute a unit lower triangular matrix $\mathbf{L} = (\ell_{ij})$ and a diagonal matrix $\mathbf{D}$ represented by a vector $\mathbf{d} = (d_1, \ldots, d_n)$ such that $\mathbf{A} = \mathbf{LDL}^\top$. Typically, the unit diagonal of $\mathbf{L}$ is not kept in memory explicitly, the strict lower triangle of $\mathbf{A}$ is overwritten with $\mathbf{L}$ and the diagonal of $\mathbf{A}$ is overwritten with $\mathbf{d}$. The well-known LDL$^\top$ factorization in column form, suitable if matrices are held in compressed sparse column format, is described in Algorithm 3.1—see (Ortega, 1988) for various implementations of the Cholesky factorization, from which corresponding implementations of the LDL$^\top$ factorization may be derived.

The first iteration of Algorithm 3.1 may be represented as the decomposition

$$\begin{bmatrix} a_{11} & \mathbf{w}^\top \\ \mathbf{w} & \mathbf{B} \end{bmatrix} = \begin{bmatrix} 1 & \\ a_{11}^{-1}\mathbf{w} & \mathbf{I} \end{bmatrix} \begin{bmatrix} a_{11} & \\ & \mathbf{B} - a_{11}^{-1}\mathbf{w}\mathbf{w}^\top \end{bmatrix} \begin{bmatrix} 1 & a_{11}^{-1}\mathbf{w}^\top \\ & \mathbf{I} \end{bmatrix}, \tag{3.1}$$

where $\mathbf{B}$ is the bottom principal submatrix of $\mathbf{A}$ consisting in all rows and columns of $\mathbf{A}$ except for the first, and $\mathbf{w}$ contains the $(n-1)$ last elements of the first column of $\mathbf{A}$. If $\mathbf{A}$ is SQD, Corollary 2.4 ensures that $\mathbf{B} - a_{11}^{-1}\mathbf{w}\mathbf{w}^\top$ is also SQD. Therefore, the leading element of the latter matrix must be nonzero and the second

Algorithm 3.1 LDL^T Factorization

Require: $\mathbf{A}$

Initialize $\mathbf{L}$ and $\mathbf{d}$ to the strict lower triangle and diagonal of $\mathbf{A}$.
for $j = 1, \ldots, n$ **do**
    for $k = 1, \ldots, j-1$ **do**
        for $i = j+1, \ldots, n$ **do**
            $\ell_{ij} = \ell_{ij} - \ell_{ik} d_k \ell_{jk}$
    for $i = j+1, \ldots, n$ **do**
        $\ell_{ij} = \ell_{ij} / d_j$
        $d_i = d_i - d_j \ell_{ij}^2$

iteration will complete. This reasoning may be applied recursively to confirm that Algorithm 3.1 terminates successfully when initialized with an SQD matrix, as anticipated by Theorem 2.2.

The limited-memory variant of Algorithm 3.1 follows the principles set out by Lin and Moré (1999) and is described as Algorithm 3.2.

Algorithm 3.2 Limited-Memory Sparse LDL^T Factorization

Require: $\mathbf{A}$, $p \in \mathbb{N}$

1: Initialize $\mathbf{L}$ and $\mathbf{d}$ to the strict lower triangle and diagonal of $\mathbf{A}$.
2: **for** $j = 1, \ldots, n$ **do**
3:     $q = |\{i > j \mid \ell_{ij} \neq 0\}|$ *// Number of nonzeros in column j of $\mathbf{L}$*
4:     **for** $k = 1, \ldots, j-1$ such that $\ell_{jk} \neq 0$ **do**
5:         **for** $i = j+1, \ldots, n$ such that $\ell_{ik} \neq 0$ **do**
6:             $\ell_{ij} = \ell_{ij} - \ell_{ik} d_k \ell_{jk}$
7:     **for** $i = j+1, \ldots, n$ such that $\ell_{ij} \neq 0$ **do**
8:         $\ell_{ij} = \ell_{ij} / d_j$
9:         $d_i = d_i - d_j \ell_{ij}^2$
10:     Retain the largest $q + p$ elements in absolute value in the j-th column of $\mathbf{L}$.

The proof of existence of the limited-memory LDL^T factorization for H-matrices follows the proof of existence of the limited-memory Cholesky factorization of Lin and Moré (1999) and is included for completeness. As indicated by Lin and Moré (1999), the last loop of Algorithm 3.2 updates the future pivots based on *all* nonzero elements in the j-th column of $\mathbf{L}$. A variant of this algorithm may also be of interest, in which line 10 appears just before the last loop, causing the future pivots to be updated based on the retained nonzero elements only. In the next result, we establish that both versions compute an incomplete LDL^T factorization in the sense that Algorithm 3.2 does not break down.

Theorem 3.1 *If $\mathbf{K}$ is a symmetric n-by-n H-matrix with no zeros on the diagonal, Algorithm 3.2 computes an incomplete LDL^T factorization.*

Proof. Iteration j of Algorithm 3.2 consists in computing the vector $\mathbf{w}$ in a matrix of the form (2.1), where $\alpha = d_j \neq 0$ is the current pivot, $\mathbf{w}$ consists in the $(n-j)$ last elements of the j-th column of $\mathbf{L}$ before they are divided by d_j, and $\mathbf{B}$ is the portion of $\mathbf{K}$ that is yet to be eliminated. Note however that the diagonal elements of $\mathbf{B}$, i.e., $d_{j+1}, \ldots, d_n$, were updated in the previous iterations. Next, some components of $\mathbf{w}$ are discarded, resulting in the sparser vector $\bar{\mathbf{w}}$. In the variant described by Algorithm 3.2, we next perform the decomposition (3.1), i.e.,

$$\begin{bmatrix} \alpha & \mathbf{w}^\mathsf{T} \\ \mathbf{w} & \mathbf{B} \end{bmatrix} = \begin{bmatrix} 1 & \\ \alpha^{-1}\bar{\mathbf{w}} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \alpha & \\ & \mathbf{B} - \alpha^{-1}(\bar{\mathbf{w}}\bar{\mathbf{w}}^\mathsf{T} + \Delta\mathbf{W}^2) \end{bmatrix} \begin{bmatrix} 1 & \alpha^{-1}\bar{\mathbf{w}}^\mathsf{T} \\ & \mathbf{I} \end{bmatrix} + \mathbf{E} \tag{3.2}$$

where $\Delta\mathbf{w} := \mathbf{w} - \bar{\mathbf{w}}$, $\Delta\mathbf{W} = \operatorname{diag}(\Delta\mathbf{w})$, and

$$\mathbf{E} := \begin{bmatrix} & \Delta\mathbf{w}^{\mathsf{T}} \\ \Delta\mathbf{w} & \alpha^{-1}\Delta\mathbf{W}^2 \end{bmatrix}$$

is the error due to substituting $\bar{\mathbf{w}}$ for $\mathbf{w}$ and is discarded. Note that the Schur complement above corresponds to the elimination with respect to $\bar{\mathbf{w}}$ but contains the updated diagonal elements based on the nonzeros in $\mathbf{w}$. Note also that the only nonzero components of the vector $\Delta\mathbf{w}$ are w_i in locations that were not discarded. In particular, the diagonal elements of the modified Schur complement are

$$\left(\mathbf{B} - \alpha^{-1}(\bar{\mathbf{w}}\bar{\mathbf{w}}^{\mathsf{T}} + \Delta\mathbf{W}^2)\right)_{ii} = b_{ii} - \alpha^{-1}(\bar{w}_i^2 + \Delta w_i^2) = b_{ii} - \alpha^{-1}w_i^2, \tag{3.3}$$

and coincide with the diagonal elements of the original Schur complement.

Suppose now that $\mathbf{K}$ is an H-matrix. At the first iteration of Algorithm 3.2, $\mathbf{K}$ is the matrix on the left-hand side of (3.2). By definition, $\mathcal{M}(\mathbf{K})$ is the M-matrix

$$\begin{bmatrix} |\alpha| & -|\mathbf{w}|^{\mathsf{T}} \\ -|\mathbf{w}| & \mathcal{M}(\mathbf{B}) \end{bmatrix}$$

and Lemma 2.8 states that the Schur complement $\mathcal{M}(\mathbf{B}) - |\alpha|^{-1}|\mathbf{w}||\mathbf{w}|^{\mathsf{T}}$ is also an M-matrix. But we have

$$\mathcal{M}(\mathbf{B}) - |\alpha|^{-1}|\mathbf{w}||\mathbf{w}|^{\mathsf{T}} \le \mathcal{M}\left(\mathbf{B} - \alpha^{-1}(\bar{\mathbf{w}}\bar{\mathbf{w}}^{\mathsf{T}} + \Delta\mathbf{W}^2)\right) \tag{3.4}$$

componentwise because $\bar{w}_i$ is either zero or w_i. Indeed, the (i,j)-th element of the left-hand side of (3.4) has the form

$$m_{ij}(\mathbf{B}) - |\alpha|^{-1}|w_i||w_j| = \begin{cases} |b_{ii}| - |\alpha|^{-1}w_i^2 & (i = j), \\ -|b_{ij}| - |\alpha|^{-1}|w_i||w_j| & (i \ne j). \end{cases}$$

On the other hand, we see from (3.3) that diagonal elements of the right-hand side of (3.4) have the form

$$|b_{ii} - \alpha^{-1}w_i^2| \ge |b_{ii}| - |\alpha|^{-1}w_i^2,$$

while, since $\Delta\mathbf{W}$ is diagonal, off-diagonal elements have the form

$$-|b_{ij} - \alpha^{-1}\bar{w}_i\bar{w}_j| \ge -|b_{ij}| - |\alpha|^{-1}|\bar{w}_i||\bar{w}_j| \ge -|b_{ij}| - |\alpha|^{-1}|w_i||w_j|,$$

where we used the triangle inequality and the fact that $\bar{\mathbf{w}} \le \mathbf{w}$. We have established (3.4) and Lemma 2.8 states that the updated Schur complement is an H-matrix with a nonzero diagonal.

The proof of the second variant of Algorithm 3.2 is established by noting that $\Delta\mathbf{W}$ only affects the diagonal of the right-hand side of (3.4). The elements (3.3) become $b_{ii} - \alpha^{-1}\bar{w}_i^2$. Using again the inequality $\bar{\mathbf{w}} \le \mathbf{w}$, we have

$$|b_{ii} - \alpha^{-1}\bar{w}_i^2| \ge |b_{ii}| - |\alpha|^{-1}\bar{w}_i^2 \ge |b_{ii}| - |\alpha|^{-1}w_i^2,$$

which completes the proof. □

According to Lemma 2.7, Theorem 3.1 applies in particular to strictly diagonally dominant SQD matrices. In order to ensure successful computation of an incomplete LDL$^{\mathsf{T}}$ factorization for general SQD matrices, we employ Algorithm 3.3. Because $\mathbf{A}$ is SQD, the diagonal scaling matrix $\mathbf{S}$ is nonsingular. The scaled matrix $\hat{\mathbf{A}}$ at line 2 is still SQD and so is each matrix of the form $\hat{\mathbf{A}} + \alpha_k\hat{\mathbf{D}}$. Note also that since $\hat{\mathbf{A}}$ is SQD, it has no zero diagonal elements, its positive diagonal elements belong to the positive-definite block $\mathbf{E}$, in the notation of (1.1), and its negative diagonal elements belong to the negative-definite block $-\mathbf{F}$. In essence, Algorithm 3.3 performs the separate updates $\mathbf{E} \leftarrow \mathbf{E} + \alpha_k\mathbf{I}$ and $\mathbf{F} \leftarrow \mathbf{F} + \alpha_k\mathbf{I}$.

That Algorithm 3.3 terminates finitely is established by the following result, whose proof follows directly from the proof of (Lin and Moré, 1999, Theorem 3.2) and from Lemma 2.7.

Algorithm 3.3 Limited-Memory LDL^T Factorization for General SQD Matrices

Require: $\mathbf{A}$ SQD, $\alpha_{\min} > 0$

1: Compute the scaling matrix $\mathbf{S} := \mathrm{diag}(\|\mathbf{A}\mathbf{e}_i\|_2)$.
2: Let $\hat{\mathbf{A}} := \mathbf{S}^{-1/2}\mathbf{A}\mathbf{S}^{-1/2}$.
3: Define $\hat{\mathbf{D}} := \mathrm{diag}(\mathrm{sign}(\hat{\mathbf{A}}_{ii}))$ and set $\alpha_0 := 0$.
4: **for** $k = 0, 1, \ldots$ **do**
5: If Algorithm 3.2 applied to $\hat{\mathbf{A}} + \alpha_k\hat{\mathbf{D}}$ succeeds, set $\alpha_F := \alpha_k$ and exit.
6: Set $\alpha_{k+1} := \max(2\alpha_k, \alpha_{\min})$.

Theorem 3.2 *Let $\mathbf{A}$ be* SQD. *If β is the maximum number of nonzeros in any column of $\mathbf{A}$, and $\hat{\mathbf{A}}$ and $\hat{\mathbf{D}}$ are defined as in Algorithm 3.3, then $\hat{\mathbf{A}} + \alpha\hat{\mathbf{D}}$ is* SQD *and strictly diagonally dominant for $\alpha > \beta^{1/2}$, and is therefore an H-matrix.*

4 Stability Considerations

In this section, we review stability results on the full LDL^T factorization of a SQD matrix, and specialize known results on the stability of incomplete factorizations to the context of Algorithm 3.2.

We refer to the LDM^T factorization of a matrix as one with $\mathbf{L}$ and $\mathbf{M}$ unit lower triangular and $\mathbf{D}$ diagonal (Golub and Van Loan, 1996). Gill, Saunders, and Shinnerl (1996) observe that the LDL^T factorization of (1.1) is a simple transformation of the LDM^T factorization of the unsymmetric but positive-definite matrix

$$\bar{\mathbf{K}} := \begin{bmatrix} \mathbf{E} & -\mathbf{C}^\mathsf{T} \\ \mathbf{C} & \mathbf{F} \end{bmatrix} \tag{4.1}$$

and that the stability of one is equivalent to the stability of the other. In particular, upon defining $\bar{\mathbf{I}} := \mathrm{blkdiag}(\mathbf{I}, -\mathbf{I})$,

$$\mathbf{K} = \mathbf{L}\mathbf{D}\mathbf{L}^\mathsf{T} \quad \text{if and only if} \quad \bar{\mathbf{K}} = \mathbf{L}\bar{\mathbf{D}}\mathbf{M}^\mathsf{T},$$

where $\bar{\mathbf{D}} := \mathbf{D}\bar{\mathbf{I}}$ and $\mathbf{M} := \bar{\mathbf{I}}\mathbf{L}\bar{\mathbf{I}}$. Because $\bar{\mathbf{K}}$ is positive definite, it may be factored without row interchanges.

We say that the factorization is stable if there exists a moderate constant $\gamma > 0$ such that

$$\| |\mathbf{L}|\ |\bar{\mathbf{D}}|\ |\mathbf{M}^\mathsf{T}| \| \le \gamma(\|\bar{\mathbf{K}}_S\| + \|\bar{\mathbf{K}}_U\bar{\mathbf{K}}_S^{-1}\bar{\mathbf{K}}_U\|), \tag{4.2}$$

where $\bar{\mathbf{K}}_S$ and $\bar{\mathbf{K}}_U$ are the symmetric and unsymmetric parts of $\bar{\mathbf{K}}$, respectively (Higham, 2002). Formulated in terms of the components of (1.1), Gill et al. (1996) establish the following stability result.

Theorem 4.1 (Gill et al., 1996, Result 4.2) *The factorization $\mathbf{P}^\mathsf{T}\mathbf{K}\mathbf{P}$ is stable for every permutation $\mathbf{P}$ if*

$$\theta(\mathbf{K}) := \left(\frac{\|\mathbf{C}\|_2}{\max(\|\mathbf{E}\|_2, \|\mathbf{F}\|_2)}\right)^2 \max(\kappa_2(\mathbf{E}), \kappa_2(\mathbf{F}))$$

is not too large.

Note that Theorem 4.1 gives a sufficient condition. There may exist permutations that lead to more stable factorizations than others.

Let $\gamma > 0$ be a stability constant associated the the SQD matrix $\mathbf{K}$ in the sense of (4.2). We say that the incomplete factorization $\mathbf{L}_p\mathbf{D}_p\mathbf{L}_p^\mathsf{T}$ produced by Algorithm 3.2 with memory parameter p, if it succeeds, is *at least as stable* as the complete factorization $\mathbf{LDL}^\mathsf{T}$ of $\mathbf{K}$ if $|\mathbf{L}_p| \le |\mathbf{L}|$ and $|\mathbf{D}_p| \le |\mathbf{D}|$. In other words, (4.2) continues to hold if the exact factors are replaced with the incomplete factors in the left-hand side. A number of results on the stability of incomplete factorizations may be found in the literature. For instance, Buoni (1990) establishes stability of incomplete factorization of H-matrices with column-diagonally-dominant pivoting. Our incomplete factorization, and that of Lin and Moré (1999), fit in the framework of Meijerink and van der Vorst (1977) who establish the following result.

Theorem 4.2 (Meijerink and van der Vorst, 1977, Theorem 3.2) *If $\mathbf{A}$ is an M-matrix, then the construction of an incomplete* LU *decomposition is at least as stable as the construction of a complete* LU *decomposition of $\mathbf{A}$ without pivoting.*

When specialized to the present context, Theorem 4.2 yields the following corollary.

Corollary 4.3 *Let $\mathbf{K}$ be an SQD M-matrix, and $\bar{\mathbf{K}}$ be given by (4.1). The incomplete factorization of $\mathbf{K}$ described by Algorithm 3.2 is at least as stable as the* LDL^T *factorization of $\mathbf{K}$ without pivoting.*

In Corollary 4.3, we used the fact that the LU (or LDM^T) factorization of $\bar{\mathbf{K}}$ is precisely the LDL^T factorization of $\mathbf{K}$. Therefore, for M-matrices, stability is guaranteed provided that $\theta(\mathbf{K})$ is not too large because Theorem 4.1 is independent of the ordering.

Messaoudi (1995) generalizes Theorem 4.2 to H-matrices.

Theorem 4.4 (Messaoudi, 1995, Theorem 2.8) *If $\mathbf{A}$ is an H-matrix, then the construction of an incomplete* LU *decomposition is at least as stable as the construction of a complete* LU *decomposition of $\mathcal{M}(\mathbf{A})$ without pivoting.*

Theorem 4.4 and the identity $\mathcal{M}(\bar{\mathbf{K}}) = \mathcal{M}(\mathbf{K})$ yield the following corollary.

Corollary 4.5 *If $\mathbf{K}$ is a SQD H-matrix, the incomplete factorization described by Algorithm 3.2 is at least as stable as the* LU *(or* LDM^T*) factorization of $\mathcal{M}(\mathbf{K})$ without pivoting.*

In the admittedly special case where $\mathbf{K}$ is strictly diagonally dominant, so is $\mathcal{M}(\mathbf{K})$. Because the latter is symmetric and has a positive diagonal, it is symmetric and positive definite, and Corollary 4.5 guarantees stability.

5 Implementation

Our implementation, code-named LLDL, is a modification of that of Lin and Moré (1999), including a revision of the dated MATLAB interface. The factorization part essentially follows the original implementation with obvious changes to implement the LDL^T factorization instead of the LL^T factorization. The input matrix is held in compressed sparse column format. The shifting strategy described in Algorithm 3.3 is implemented by adding α_k to positive diagonal elements and subtracting α_k from negative diagonal elements.

In the rest of this section, we concentrate on invoking our library from MATLAB. Because Algorithm 3.2 only need access the lower triangle of the input matrix $\mathbf{A}$, it is technically possible to devise a MATLAB implementation of an iterative solver that only requires storage of this triangle. Recall that MATLAB has no concept of a symmetric matrix and needs to store both triangles to perform, e.g., matrix-vector products. For

```
K = mmread('matrix.mtx');          % Lower triangle in MM format.
rhs = load('rhs.txt');
P = symamd(K);                     % Only references lower triangle.
PKP = K(P,P);                      % Permuted matrix.
M = opLLDL(PKP, 10)                % Memory factor p=10.
[x, flags, resids] = minres(rhs(P), opSymMatrix(PKP), M);
```

Listing 1: Example usage of the LLDL MATLAB interface

this reason we introduce a new operator class `opSymMatrix` based on the SPOT abstract linear operator library for MATLAB (Friedlander and van Den Berg, 2013). This operator embodies and behaves like a symmetric matrix but only stores its lower triangle, i.e., if `A` is an `opSymMatrix` object representing $\mathbf{A}$ using only its lower triangle, `A*x` returns the expected value. Similarly, an operator class `opLLDL` embodies the limited-memory factorization—instantiating it performs the limited-memory factorization with the requested value of p and stores the factors inside the object. If `M` is an `opLLDL` instance, the operation `M*x` is equivalent to the forward and backward sweeps $\mathbf{L}^{-\mathsf{T}}|\mathbf{D}|^{-1}\mathbf{L}^{-1}\mathbf{x}$. This allows us to write preconditioned iterative methods for linear systems without the need to pass explicit matrices or functions as arguments— we merely need to pass objects that support multiplication. This is more natural from both an iterative method and a programming point of view. Listing 1 gives an example where the lower triangle of a matrix stored in MatrixMarket format is loaded along with a right-hand side vector, a symmetric AMD permutation is computed, a limited-memory factorization of the permuted matrix is performed with $p = 10$ and finally, our implementation of the preconditioned MINRES method is called to solve the permuted system. Note that the coefficient matrix is wrapped into an `opSymMatrix` operator and the preconditioner is given as an `opLLDL` operator. Internally, our implementation of MINRES only performs products `A*x` with the operator supplied and `M*x` with the preconditioner; it does not assume that one or the other is an explicit matrix. Our implementation is available from `github.com/optimizers/lldl`.

6 Interior-Point Methods

One of the main motivations for this work is the iterative solution of linear systems arising from interior-point methods for inequality-constrained optimization problems. We test our limited-memory preconditioner on systems generated in the course of the iterations of the method proposed by Friedlander and Orban (2012) for convex quadratic programs

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} \;\; \mathbf{g}^{\mathsf{T}}\mathbf{x} + \tfrac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{H}\mathbf{x} \quad \text{subject to } \; \mathbf{J}\mathbf{x} = \mathbf{b}, \; \mathbf{x} \geq 0, \tag{6.1}$$

where $\mathbf{H} = \mathbf{H}^{\mathsf{T}}$ is positive semi-definite. Applying a primal-dual interior-point method to (6.1) leads to symmetric linear systems with coefficient

$$\mathbf{K}_3 := \begin{bmatrix} \mathbf{H} & \mathbf{J}^{\mathsf{T}} & -\mathbf{Z}^{\frac{1}{2}} \\ \mathbf{J} & & \\ -\mathbf{Z}^{\frac{1}{2}} & & -\mathbf{X} \end{bmatrix} \quad \text{or} \quad \mathbf{K}_2 := \begin{bmatrix} \mathbf{H} + \mathbf{X}^{-1}\mathbf{Z} & \mathbf{J}^{\mathsf{T}} \\ \mathbf{J} & \end{bmatrix}, \tag{6.2}$$

where $\mathbf{X} := \operatorname{diag}(\mathbf{x})$ and $\mathbf{Z} := \operatorname{diag}(\mathbf{z})$ are positive definite and $\mathbf{z}$ is the vector of dual variables associated to the bound constraints of (6.1). The above matrices are symmetric and indefinite but not SQD. We refer the interested reader to the overviews of Forsgren, Gill, and Wright (2002) and Gould, Orban, and Toint (2005) for more information and numerous pointers to the literature.

In order to account for situations where the leading block of (6.2) is not positive definite and/or $\mathbf{J}$ does not have full row rank, Friedlander and Orban (2012) propose to solve instead the regularized problem

$$\begin{aligned} &\underset{\mathbf{x}\in\mathbb{R}^n,\mathbf{r}\in\mathbb{R}^m}{\text{minimize}} && \mathbf{g}^{\mathsf{T}}\mathbf{x} + \tfrac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{H}\mathbf{x} + \tfrac{1}{2}\rho\|\mathbf{x}-\mathbf{x}_k\|_2^2 + \tfrac{1}{2}\delta\|\mathbf{r}+\mathbf{y}_k\|_2^2 \\ &\text{subject to} && \mathbf{J}\mathbf{x} + \delta\mathbf{r} = \mathbf{b}, \; \mathbf{x} \geq 0, \end{aligned} \tag{6.3}$$

where $\rho > 0$ and $\delta > 0$ are iteration-dependent regularization parameters, $\mathbf{r}$ is a vector of auxiliary variables, $\mathbf{x}_k$ is the current primal estimate and $\mathbf{y}_k$ is the current estimate of the Lagrange multipliers associated to the equality constraints of (6.1). An interior-point method applied to (6.3) leads to systems with coefficient

$$\mathbf{K}_3(\rho,\delta) := \begin{bmatrix} \mathbf{H}+\rho\mathbf{I} & \mathbf{J}^\top & -\mathbf{Z}^{\frac{1}{2}} \\ \mathbf{J} & -\delta\mathbf{I} & \\ -\mathbf{Z}^{\frac{1}{2}} & & -\mathbf{X} \end{bmatrix} \quad \text{or} \quad \mathbf{K}_2(\rho,\delta) := \begin{bmatrix} \mathbf{H}+\mathbf{X}^{-1}\mathbf{Z}+\rho\mathbf{I} & \mathbf{J}^\top \\ \mathbf{J} & -\delta\mathbf{I} \end{bmatrix}, \tag{6.4}$$

which are SQD. Owing to their block dimension, we refer to the above matrices as the 3×3 and 2×2 block formulations. Greif et al. (2012) conduct a spectral analysis of both regularized matrices, as well as their unregularized counterparts, and conclude that if strict complementarity is satisfied at a solution of (6.1), i.e., $x_i + z_i > 0$ for all $i = 1,\dots,n$, the 3×3 block matrix has unconditionally uniformly bounded condition number. By contrast, the condition number of the 2×2 block formulation increases without bound provided at least one component of $\mathbf{x}$ vanishes in the limit, which is the typical situation.

Gill et al. (1996) establish that the $\mathrm{LDL}^\top$ factorization of $\mathbf{K}_2(\rho,\delta)$ is stable after an appropriate scaling, except perhaps in the last few iterations of the interior-point method.

In order to determine when a general SQD matrix such as (1.1) is an H-matrix, it is instructive to use the block inverse formula

$$\begin{aligned}\mathcal{M}(\mathbf{K})^{-1} &= \begin{bmatrix} \mathcal{M}(\mathbf{E}) & -|\mathbf{C}|^\top \\ -|\mathbf{C}| & \mathcal{M}(\mathbf{F}) \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \mathbf{S}^{-1} & \mathbf{S}^{-1}|\mathbf{C}|^\top\mathcal{M}(\mathbf{F})^{-1} \\ \mathcal{M}(\mathbf{F})^{-1}|\mathbf{C}|\mathbf{S}^{-1} & \mathcal{M}(\mathbf{F})^{-1}+\mathcal{M}(\mathbf{F})^{-1}|\mathbf{C}|\mathbf{S}^{-1}|\mathbf{C}|^\top\mathcal{M}(\mathbf{F})^{-1} \end{bmatrix},\end{aligned}$$

where $\mathbf{S} := \mathbf{K}/\mathcal{M}(\mathbf{F}) = \mathcal{M}(\mathbf{E}) - |\mathbf{C}|^\top\mathcal{M}(\mathbf{F})^{-1}|\mathbf{C}|$. If both $\mathbf{E}$ and $\mathbf{F}$ are H-matrices, note that the off-diagonal elements of $\mathbf{S}$ are all nonpositive. General conditions for $\mathbf{K}$ to be an H-matrix impose that $\mathbf{S}^{-1} \geq \mathbf{0}$, which occurs if $\mathbf{S}$ is an M-matrix. In turn, (Axelsson, 1994, Lemma 6.2) guarantees that this occurs if $\mathbf{S}$ is strictly diagonally dominant. In the case of $\mathbf{K}_2$, $\mathbf{K}_3$ and their regularized counterparts, we can be more precise. Note first that if $\mathbf{K}_3$, respectively $\mathbf{K}_3(\rho,\delta)$, is an H-matrix, item 5 of Lemma 2.8 guarantees that $\mathbf{K}_2$, respectively $\mathbf{K}_2(\rho,\delta)$, is an also H-matrix.

In the remainder of this section, we establish the following proposition. Note that in the special case of linear programming, $\mathbf{H} = \mathbf{0}$ and the assumption on $\mathbf{H}$ may be dropped. The proposition gives sufficient conditions on ρ and δ for $\mathbf{K}_3(\rho,\delta)$ to be an H-matrix.

Proposition 6.1 *Assume* $\mathbf{H}$ *is a symmetric positive semi-definite H-matrix. Let* $\mathbf{S} := \mathcal{M}(\mathbf{H}) + \rho\mathbf{I} - \mathbf{Z}^{1/2}\mathbf{X}^{-1}\mathbf{Z}^{1/2}$ *and assume that* $\mathbf{S}$ *is nonsingular, that* $(\mathbf{x},\mathbf{z}) > \mathbf{0}$*, that* $\rho > \min_i z_i/x_i$*, and that* $\tilde{\mathbf{S}} := \delta\mathbf{I} - |\mathbf{J}|\mathbf{S}^{-1}|\mathbf{J}|^\top$ *is an M-matrix. Then* $\mathbf{K}_3(\rho,\delta)$*, and therefore also* $\mathbf{K}_2(\rho,\delta)$*, is an H-matrix.*

Proof. Let

$$\mathbf{W} := \begin{bmatrix} \mathbf{H}+\rho\mathbf{I} & -\mathbf{Z}^{1/2} \\ -\mathbf{Z}^{1/2} & -\mathbf{X} \end{bmatrix}. \tag{6.5}$$

Then

$$\mathcal{M}(\mathbf{W}) = \begin{bmatrix} \mathcal{M}(\mathbf{H})+\rho\mathbf{I} & -\mathbf{Z}^{1/2} \\ -\mathbf{Z}^{1/2} & \mathbf{X} \end{bmatrix}, \tag{6.6}$$

and

$$\mathbf{M} := \mathcal{M}(\mathbf{W})^{-1} = \begin{bmatrix} \mathbf{S}^{-1} & \mathbf{S}^{-1}\mathbf{Z}^{1/2}\mathbf{X}^{-1} \\ \mathbf{X}^{-1}\mathbf{Z}^{1/2}\mathbf{S}^{-1} & \mathbf{X}^{-1}\mathbf{Z}^{1/2}\mathbf{S}^{-1}\mathbf{Z}^{1/2}\mathbf{X}^{-1}+\mathbf{X}^{-1} \end{bmatrix}. \tag{6.7}$$

Now $\mathcal{M}(\mathbf{K}_3(\rho,\delta))^{-1}$ is a symmetric permutation of

$$\mathcal{M}(\tilde{\mathbf{K}}_3(\rho,\delta))^{-1} = \begin{bmatrix} \mathcal{M}(\mathbf{W}) & -|\tilde{\mathbf{J}}|^\top \\ -|\tilde{\mathbf{J}}| & \delta\mathbf{I} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{M} + \mathbf{M}|\tilde{\mathbf{J}}|^\top \tilde{\mathbf{S}}^{-1}|\tilde{\mathbf{J}}|\mathbf{M} & \mathbf{M}\tilde{\mathbf{J}}^\top \tilde{\mathbf{S}}^{-1} \\ \tilde{\mathbf{S}}^{-1}\tilde{\mathbf{J}}\mathbf{M} & \tilde{\mathbf{S}}^{-1} \end{bmatrix}, \tag{6.8}$$

where $\tilde{\mathbf{J}} := \begin{bmatrix}\mathbf{J} & \mathbf{0}\end{bmatrix}$ and $\tilde{\mathbf{S}} = \delta\mathbf{I} - |\tilde{\mathbf{J}}|\mathbf{M}|\tilde{\mathbf{J}}|^\top = \delta\mathbf{I} - |\mathbf{J}|\mathbf{S}^{-1}|\mathbf{J}|^\top$.

Consider first the case of linear programming where $\mathbf{H} = \mathbf{0}$. Then $\mathbf{S}$ is an M-matrix provided that $\rho > \min_i z_i/x_i$. Under the latter assumption, $\mathbf{S}^{-1} > \mathbf{0}$. Since $(\mathbf{x},\mathbf{z}) > \mathbf{0}$, all blocks of (6.7) are nonnegative and $\mathbf{W}$ is an H-matrix. Consider now (6.8). Since $\mathbf{M} \geq 0$, it is clear that $\mathbf{K}_3(\rho,\delta)$ is an H-matrix provided that $\tilde{\mathbf{S}}$ is an M-matrix.

In the general case where $\mathbf{H} \neq \mathbf{0}$, the same reasoning as above applies under the additional assumption that $\mathbf{H}$ is an H-matrix. Indeed, item 1 of Lemma 2.8 implies that there exists $\mathbf{v} > \mathbf{0}$ such that $\mathcal{M}(\mathbf{H})\mathbf{v} > \mathbf{0}$. If $\rho > \min_i z_i/x_i$, then $\mathbf{S}\mathbf{v} > \mathbf{0}$ and $\mathbf{S}$ is an M-matrix. □

In the absence of regularization, i.e., $\rho = \delta = 0$, $\mathbf{W}$ and $\mathbf{K}_3$ are in general not H-matrices because the Schur complements $\mathbf{S} = -\mathbf{Z}^{1/2}\mathbf{X}^{-1}\mathbf{Z}^{1/2}$ and $\tilde{\mathbf{S}} = -|\mathbf{J}|\mathbf{S}^{-1}|\mathbf{J}|^\top$ are nonpositive and item 1 of Lemma 2.8 implies that they cannot be M-matrices.

Note that the off-diagonal elements of $\tilde{\mathbf{S}}$ are all nonnegative. Its diagonal elements have the form

$$\delta - |\mathbf{a}_j|^\top \mathbf{S}^{-1}|\mathbf{a}_j| = \delta - \sum_{i=1}^{n} s_i^{-1} a_{ji}^2, \quad j = 1,\dots,m,$$

where s_i is the i-th diagonal of $\mathbf{S}$ and $\mathbf{a}_j^\top$ is the j-th row of $\mathbf{J}$, i.e., the j-th constraint gradient. From (Axelsson, 1994, Lemma 6.2), $\tilde{\mathbf{S}}$ is an M-matrix if its diagonal elements are positive and if it is strictly diagonally dominant, i.e.,

$$\delta > \sum_{i=1}^{n} |\mathbf{a}_j|^\top \mathbf{S}^{-1}|\mathbf{a}_i|, \quad j = 1,\dots,n. \tag{6.9}$$

Under the assumption (6.9), $\tilde{\mathbf{S}}$ is an M-matrix and as a consequence, $\mathbf{K}_3(\rho,\delta)$ and $\mathbf{K}_2(\rho,\delta)$ are H-matrices.

The above may be construed as a proof of Theorem 3.2 for the special case of $\mathbf{K}_3(\rho,\delta)$ and $\mathbf{K}_2(\rho,\delta)$.

As we illustrate in the next section, the lower bounds on the regularization parameters found in the present section for a limited-memory factorization to exist are pessimistic overestimates in practice and our implementation identifies effective limited-memory preconditioners even for very small values of ρ and δ.

7 Numerical Experiments

Given an SQD system $\mathbf{A}\mathbf{x} = \mathbf{b}$, a permutation matrix $\mathbf{P}$ and $p \in \mathbb{N}$, we compute the limited-memory factors $\mathbf{L}$ and $\mathbf{D}$ of $\mathbf{P}^\top\mathbf{A}\mathbf{P}$ and solve the preconditioned system

$$|\mathbf{D}|^{-1/2}\mathbf{L}^{-1}\mathbf{P}^\top\mathbf{A}\mathbf{P}\mathbf{L}^{-\top}|\mathbf{D}|^{-1/2}\bar{\mathbf{x}} = |\mathbf{D}|^{-1/2}\mathbf{L}^{-1}\mathbf{P}^\top\mathbf{b},$$

where $\bar{\mathbf{x}} := |\mathbf{D}|^{1/2}\mathbf{L}^\top\mathbf{P}^\top\mathbf{x}$ with MINRES. As this system is still symmetric and the preconditioner is positive definite, MINRES may be equivalently applied to $\mathbf{P}^\top\mathbf{A}\mathbf{P}\mathbf{x} = \mathbf{P}^\top\mathbf{b}$ with preconditioner $\mathbf{L}^{-\top}|\mathbf{D}|^{-1}\mathbf{L}^{-1}$.

We generated linear systems at iterations 0, 5 and 10 of the interior-point method of Friedlander and Orban (2012), implemented as part of the NLPy[1] framework. The test problems are sparse convex quadratic

[1] github.com/dpo/nlpy

programs taken from the CUTEr collection (Gould, Orban, and Toint, 2003). Iteration 0 uses the initial primal-dual guess as computed by the standard procedure suggested by Mehrotra (1992). At iteration 0, far from a solution, the matrices (6.4) should both be reasonably well conditioned and $\rho_k = \delta_k = 1$. At iterations 5 and 10, the method uses $\rho_k = \delta_k = 10^{-5}$ and 10^{-8}, respectively, and the conditioning of the 2×2 formulation worsens. This ill-conditioning is not particularly due to the values of the regularization parameters, except when $\mathbf{J}$ is rank deficient, but mostly to components of $\mathbf{x}$ and $\mathbf{z}$ converging to zero (Greif et al., 2012, Section 4). Table 7.1 summarizes our test problems and some of their characteristics. The matrices come with their corresponding right-hand side, also output during the iterations of the interior-point method. The table reports the systems size and the density of the strict lower triangle computed as $\mathrm{nnz}(\mathbf{K})/(n(n-1)/2)$, where n is the size of $\mathbf{K}$ and $\mathrm{nnz}(\mathbf{K})$ represents the number of nonzeros in the strict lower triangle of $\mathbf{K}$. Some problems do not have inequality constraints. For those, the two formulations (6.4) are the same.

The aim of the experiments below is to evaluate the potential of limited-memory preconditioners in the context of interior-point methods. The current implementation of the interior-point method rests upon Theorem 2.2 and solves the systems with a full $\mathbf{LDL}^\mathsf{T}$ factorization. Some problems require less than 5 or 10 iterations to be solved, so that 76 systems were output at iteration 0, 58 at iteration 5, and 42 at iteration 10.

Table 7.1: Test linear systems arising in quadratic programs from the CUTEr collection. The problem size and density of the strict lower triangle are reported for the 2×2 and 3×3 block formulations (6.4).

Name	2×2	Density	3×3	Density
AUG2D	30200	8.8e−05	30200	8.8e−05
AUG2DC	30200	8.8e−05	30200	8.8e−05
AUG2DCQP	70600	3.2e−05	90800	2.4e−05
AUG2DQP	70600	3.2e−05	90800	2.4e−05
AUG3D	4873	5.5e−04	4873	5.5e−04
AUG3DC	4873	5.5e−04	4873	5.5e−04
AUG3DCQP	12619	1.8e−04	16492	1.3e−04
AUG3DQP	12619	1.8e−04	16492	1.3e−04
CVXQP1_L	55000	5.6e−05	75000	3.7e−05
CVXQP1_M	5500	5.6e−04	7500	3.7e−04
CVXQP1_S	550	5.5e−03	750	3.7e−03
CVXQP2_L	52500	5.6e−05	72500	3.7e−05
CVXQP2_M	5250	5.6e−04	7250	3.7e−04
CVXQP2_S	525	5.5e−03	725	3.7e−03
CVXQP3_L	57500	5.6e−05	77500	3.7e−05
CVXQP3_M	5750	5.6e−04	7750	3.7e−04
CVXQP3_S	575	5.5e−03	775	3.7e−03
DUAL1	426	4.3e−02	596	2.3e−02
DUAL2	481	4.2e−02	673	2.2e−02
DUAL3	556	4.2e−02	778	2.2e−02
DUAL4	376	4.4e−02	526	2.4e−02
DUALC1	474	2.0e−02	706	9.9e−03
DUALC2	492	1.6e−02	734	7.9e−03
DUALC5	595	1.4e−02	888	7.2e−03
DUALC8	1045	8.4e−03	1563	4.2e−03
GENHS28	18	2.2e−01	18	2.2e−01
GOULDQP2	3844	5.7e−04	5242	4.1e−04
GOULDQP3	3844	6.1e−04	5242	4.3e−04
HS118	133	1.7e−02	192	1.2e−02
HS21	12	1.7e−01	17	1.2e−01
HS21MOD	25	6.3e−02	34	5.0e−02
HS268	15	3.8e−01	20	2.4e−01
HS35	11	2.2e−01	15	1.5e−01

Continued on next page

Table 7.1 – Continued from previous page

Name	2×2	Density	3×3	Density
HS35MOD	9	2.8e−01	12	2.0e−01
HS51	8	3.2e−01	8	3.2e−01
HS52	8	3.2e−01	8	3.2e−01
HS53	28	7.7e−02	38	5.5e−02
HS76	18	1.5e−01	25	1.0e−01
HUES-MOD	30002	8.9e−05	40002	6.2e−05
HUESTIS	30002	8.9e−05	40002	6.2e−05
KSIP	2022	1.0e−02	3023	4.8e−03
LISWET1	30002	8.9e−05	40002	6.2e−05
LISWET10	30002	8.9e−05	40002	6.2e−05
LISWET11	30002	8.9e−05	40002	6.2e−05
LISWET12	30002	8.9e−05	40002	6.2e−05
LISWET2	30002	8.9e−05	40002	6.2e−05
LISWET3	30002	8.9e−05	40002	6.2e−05
LISWET4	30002	8.9e−05	40002	6.2e−05
LISWET5	30002	8.9e−05	40002	6.2e−05
LISWET6	30002	8.9e−05	40002	6.2e−05
LISWET7	30002	8.9e−05	40002	6.2e−05
LISWET8	30002	8.9e−05	40002	6.2e−05
LISWET9	30002	8.9e−05	40002	6.2e−05
LOTSCHD	43	8.6e−02	55	6.1e−02
MOSARQP1	8900	2.3e−04	12100	1.7e−04
MOSARQP2	3900	7.1e−04	5400	4.7e−04
POWELL20	30000	6.7e−05	40000	5.0e−05
PRIMAL1	497	4.8e−02	583	3.5e−02
PRIMAL2	843	2.3e−02	940	1.9e−02
PRIMAL3	969	4.6e−02	1081	3.7e−02
PRIMAL4	1641	1.2e−02	1717	1.1e−02
PRIMALC1	678	1.1e−02	902	6.7e−03
PRIMALC2	703	8.4e−03	939	5.3e−03
PRIMALC5	859	7.8e−03	1145	4.8e−03
PRIMALC8	1542	4.4e−03	2053	2.7e−03
QPCBLEND	354	1.1e−02	468	7.3e−03
QPCBOEI1	2335	2.0e−03	3306	1.2e−03
QPCBOEI2	903	4.6e−03	1281	2.7e−03
QPCSTAIR	1740	3.2e−03	2272	2.1e−03
S268	15	3.8e−01	20	2.4e−01
STCQP1	22537	2.1e−04	30731	1.3e−04
STCQP2	22537	2.1e−04	30731	1.3e−04
TAME	7	3.3e−01	9	2.5e−01
UBH1	54021	4.9e−05	66027	3.9e−05
YAO	6004	4.4e−04	8005	3.1e−04
ZECEVIC2	14	1.5e−01	20	1.1e−01

In the following numerical experiments, we evaluate the quality of limited-memory preconditioners in terms of the factor density and the number of iterations taken by MINRES to solve the linear system to within a relative stopping tolerance of 10^{-6} with a limit of $\min(n, 500)$ iterations. More precisely, our performance measure is the *efficiency*, suggested by Scott and Tůma (2013), and defined as the product of the number of nonzeros in $\mathbf{L}$ with the number of preconditioned MINRES iterations. All our tests are run with MATLAB R2012a on a MacBook Pro equipped with a 2.6GHz Intel Core i7 processor and 16 GB of RAM.

We examine the impact of two reorderings on the incomplete factorization and the subsequent system solve. The first is the symmetric approximate minimum degree reordering of Amestoy et al. (1996). This ordering aims to minimize the fill-in in factors and the MATLAB function `symamd()` plays well with our implementation since it only references the lower triangle of the coefficient matrix. The second is the reverse Cuthill and McKee (1969) ordering—see also (Chan and George, 1980). Again, the MATLAB function `symrcm()` is in line with our implementation because `symrcm(A)` works with the sparsity pattern of $\mathbf{A} + \mathbf{A}^\mathsf{T}$ and therefore is appropriate if only the lower triangle of $\mathbf{A}$ is stored. We tested other orderings, notably those included in the packages HSL_MC64 and HSL_MC80 from the HSL (2013) but we do not consider them in the numerical results below as they consistently yielded denser factors and more MINRES iterations in our experiments.

Our results are presented in the form of performance profiles (Dolan and Moré, 2002) on a logarithmic scale. Recall that for competing algorithms $\mathcal{A}_i$, $i = 1, \dots, N$, the performance profile of algorithm $\mathcal{A}_j$ is the step function

$$\tau \in [1, \infty) \mapsto \rho_j(\tau) := \left| \left\{ s \in \mathcal{S} \mid \mu_{j,s} \le \tau \min_i \mu_{i,s} \right\} \right|,$$

where $\mathcal{S}$ is the set of test systems considered, and $\mu_{i,s}$ is the performance measure of algorithm $\mathcal{A}_i$ on system s. In our case, the performance measure is the efficiency. In other words, for a given $\tau \ge 1$, $\rho_j(\tau)$ is the proportion of problems on which $\mathcal{A}_j$ is within a factor τ of the best performing algorithm. In the results below, a logarithmic scale in base 2 is used for τ, which helps differentiate the curves.

Figure 7.1 reports numerical results on the problems of Table 7.1 with $p = 0$, 10 and 20 and at iteration 0 of the interior-point method. The first row of profiles corresponds to the SYMAMD ordering while the second row corresponds to SYMRCM. Hence, the top left plot of Figure 7.1 indicates that the unpreconditioned variant and the preconditioned variants with $p = 0$, 10 and 20 are the best in terms of efficiency on about 5%, 25%, 65%, and 80% of the test problems, respectively. Large values of τ indicate that the unpreconditioned variant solves almost all test problems, while all preconditioned variants solve all test problems, i.e., identify a solution with a relative residual below 10^{-6} before the limit of 500 iterations is attained. When $\tau = 2$, the variant with $p = 0$ is within a factor of 4 of the best on about 90% of the problems.

Figures 7.2 and 7.3 report corresponding results on problems generated at iterations 5 and 10 of the interior-point method. In all cases, the final value of the shift α_k from Algorithm 3.3 is zero, i.e., the incomplete factorization succeeds at the first attempt. This is in part due to the fact that the input matrices are already SQD, but it also means that Algorithm 3.2 never broke down in our tests. At iteration 5, the preconditioned variant with $p = 0$ still fails to solve about 21% of the 2×2 problems for both orderings, while $p = 10$ and 20 solve all problems. The global trend across all tests is that the efficiency improves as p increases, though the fact that the profiles for $p = 10$ and $p = 20$ are often nearly superposed suggests that there may not be virtue in increasing p further. We also note that at iteration 10, the benefit of the preconditioners on 2×2 systems is less pronounced in terms of efficiency than for 3×3 systems. Overall, the unpreconditioned variant is less robust on 3×3 systems than on 2×2 systems.

On all sets of plots, both orderings appear to have the same qualitative behavior. In order to compare them, we selected the variant with $p = 20$ as it exhibits good performance and robustness with both orderings. As Figure 7.4 illustrates, SYMAMD consistently outperforms SYMRCM regardless of problem size or iteration number. In all cases, SYMAMD combined to $p = 20$ performs best on about 90% of the problems.

Finally, we compare the variants in terms of CPU time as measured by the MATLAB `etime` function. Our measurements include the time to compute the preconditioner and the time to perform the MINRES iterations, but exclude the time to apply the reordering, as the latter is part of all experiments. In our MATLAB implementation, the triangular solves are performed implicitly inside the `opLLDL` operator using the backslash command. Figure 7.5 reports our results for the systems generated at iteration 0 of the interior-point method, and Figures 7.6 and 7.7 correspond to iterations 5 and 10, respectively. The profiles indicate that the benefit of the limited-memory preconditioner in terms of time is most apparent on the 3×3 systems and that this benefit increases as the interior-point method progresses. At iteration 0, when all systems are reasonably conditioned, the preconditioned variants cause an overall increase in solution time on the 2×2 problems, while the variants with $p = 10$ and $p = 20$ are already effective on the 3×3 systems. At iteration 5, all preconditioned variants globally improve the solution time. At iteration 10, the variant with $p = 0$ has a slightly adverse impact on time on the 2×2 systems, while all preconditioned variants improve both the

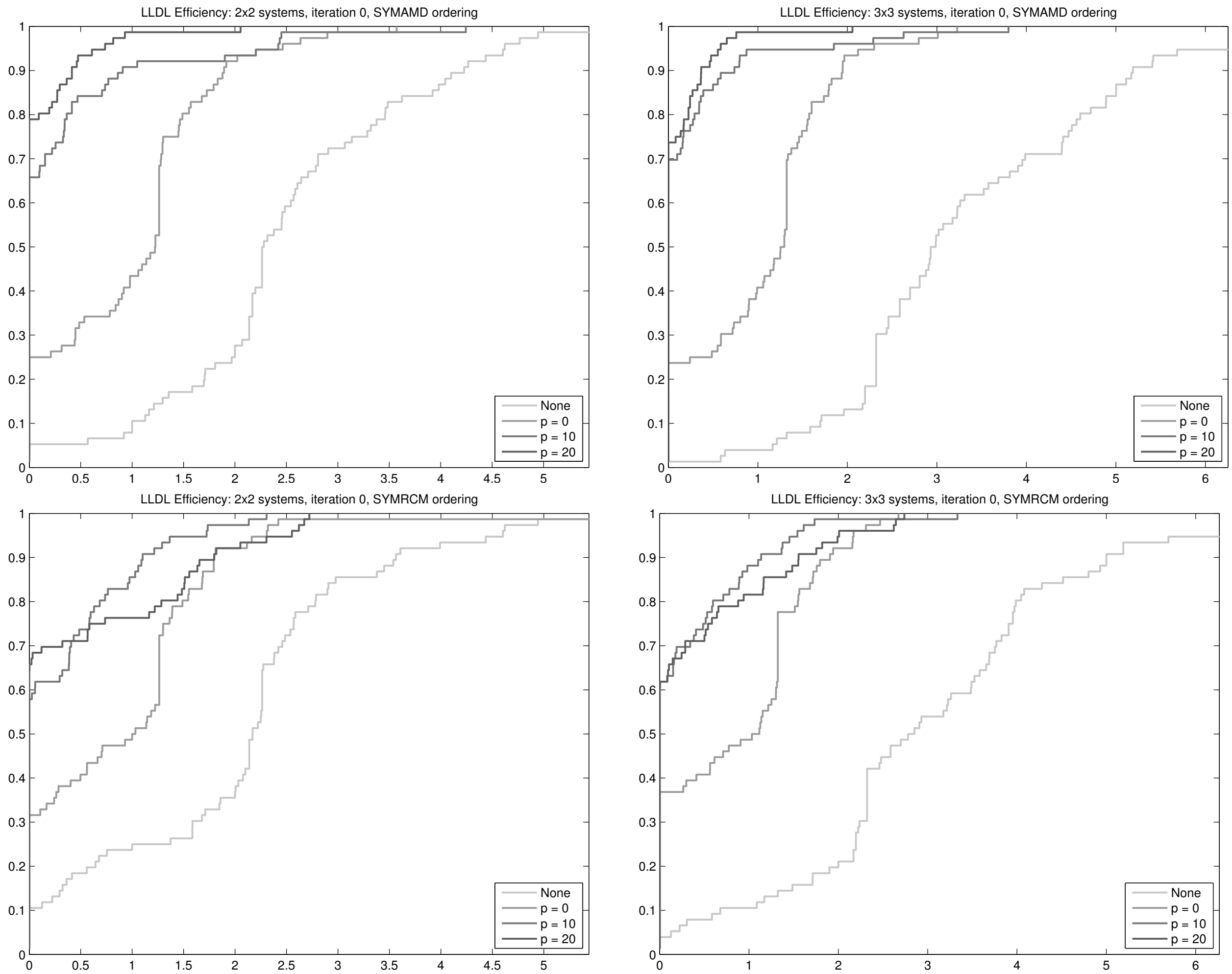

Figure 7.1: Efficiency profiles for 2×2 and 3×3 systems (6.4) at iteration 0 of the interior-point method. The variants compared are MINRES without preconditioner, and with limited-memory preconditioners corresponding to $p = 0$, 10 and 20. The top row corresponds to SYMAMD ordering and the bottom row to SYMRCM ordering.

efficiency and the time on the 3×3 systems. We expect that this is because the conditioning of the 2×2 systems continues to deteriorate, while that of the 3×3 systems stabilizes around a moderate value (Greif et al., 2012).

We define the growth factor as

$$g := \frac{\max\{|\tilde{\ell}_{ij}| \mid i, j = 1, \ldots, n\}}{\max\{|a_{ij}| \mid i, j = 1, \ldots, n\}}, \qquad \text{where} \qquad \tilde{\mathbf{L}} := \mathbf{L}|\mathbf{D}|^{\frac{1}{2}}.$$

Note that this definition of the growth factor coincides with that used in the Cholesky factorization in the case of a positive-definite $\mathbf{A}$. In all of our tests, the growth factor remained under 210 for 2×2 systems and under $1,000$ for 3×3 systems, except in the following cases using SYMAMD ordering. The 2×2 formulation of problem HUES-MOD generates a growth factor of $1.3 \cdot 10^5$ at iteration 5 and $5.9 \cdot 10^4$ at iteration 10, both with $p = 0$. The 3×3 formulation of problem KSIP generates a growth factor of $1.9 \cdot 10^5$ at iterations 5 and 10 with $p = 0$. Increasing the memory parameter brings the growth factor back within acceptable bounds. Overall the SYMRCM ordering appears to keep the growth factor consistently modest independently of the memory parameter.

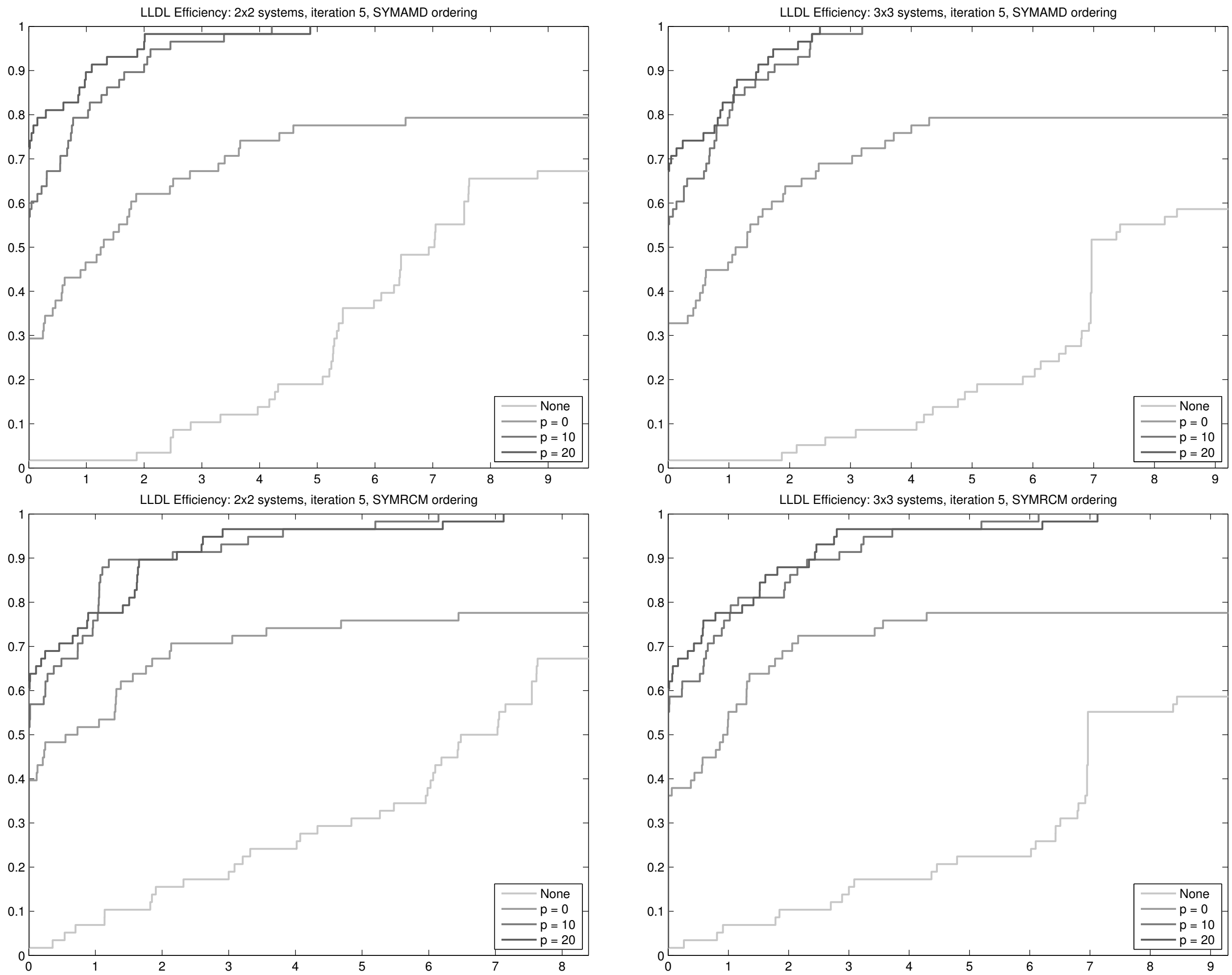

Figure 7.2: Efficiency profiles for 2×2 and 3×3 systems (6.4) at iteration 5 of the interior-point method. The variants compared are MINRES without preconditioner, and with limited-memory preconditioners corresponding to $p = 0$, 10 and 20. The top row corresponds to SYMAMD ordering and the bottom row to SYMRCM ordering.

8 Potential Extensions

Future versions of LLDL could feature a number of enhancements compared to the simple implementation presented here. Firstly, a drop tolerance could be specified by the user in addition to a memory factor so as to encourage further sparsity. It remains to be seen whether Theorem 3.2 generalizes to the case of a drop tolerance. Secondly, for performance reasons, the ordering could be performed implicitly in the course of the factorization and recorded in the permutation vector rather than performed explicitly before attempting a factorization. Finally, Algorithm 3.3 can be applied to a general symmetric indefinite matrix, e.g., one with a zero bottom block. This would result in preconditioning a general symmetric system with the incomplete LDL^T factorization of a related SQD operator. A conceptual disadvantage of this approach is that, by contrast with the SQD case, the incomplete factors do not necessarily converge to exact factors as the memory parameter increases. However, preconditioning with SQD operators has been suggested in the past by Axelsson and Neytcheva (2003). Theorem 3.2 generalizes to the symmetric indefinite case because for all sufficiently large $\alpha > 0$, $\hat{\mathbf{A}} + \alpha\hat{\mathbf{D}}$ is SQD and diagonally dominant. Assuming $\mathbf{F} = \mathbf{0}$ in (1.1), Algorithm 3.3 then computes an incomplete factorization of

$$\begin{bmatrix} \mathbf{E} + \alpha\mathbf{I} & \mathbf{C}^\mathsf{T} \\ \mathbf{C} & -\alpha\mathbf{I} \end{bmatrix}.$$

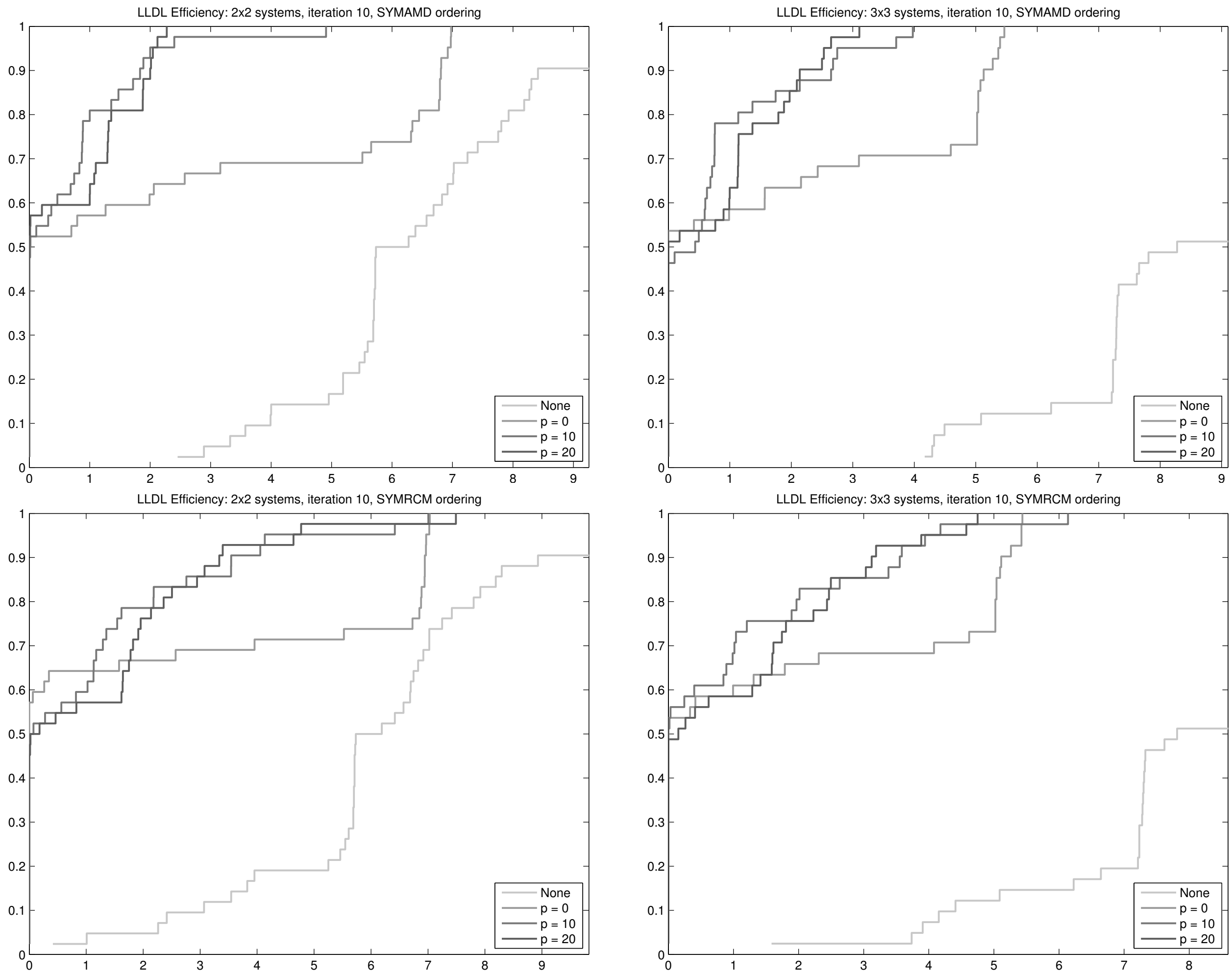

Figure 7.3: Efficiency profiles for 2×2 and 3×3 systems (6.4) at iteration 10 of the interior-point method. The variants compared are MINRES without preconditioner, and with limited-memory preconditioners corresponding to $p = 0$, 10 and 20. The top row corresponds to SYMAMD ordering and the bottom row to SYMRCM ordering.

As the experiments of §7 suggest, the factorization is likely to succeed for moderate values of α.

The following simple experiment suggests that even in the general symmetric indefinite case, the growth factor does not necessarily take large values. We computed the growth factor associated to incomplete factors of a Helmholtz-type operator. As the eigenvalues of the discrete two-dimensional Laplacian lie between 0 and 8, we use shifts ranging from 1 to 7 so the shifted operator is indefinite. The MATLAB excerpt of Listing 2 produces the results of Table 8.2. In the table, λ is the shift applied to the Laplacian, g is the growth factor, and α is the shift resulting from Algorithm 3.3. The incomplete factorization always succeeds with the smallest

```
K = delsq(numgrid('S', 100+2));      % 2D Laplacian of size 10000.
growth = []; shifts = [];
for fact = [1:7]
  LLDL = opLLDL(K - fact * speye(size(K)), 10);          % p = 10.
  growth = [growth LLDL.growth]; shifts = [shifts LLDL.shift];
end
```

Listing 2: Incomplete factors of a Helmholtz-type operator

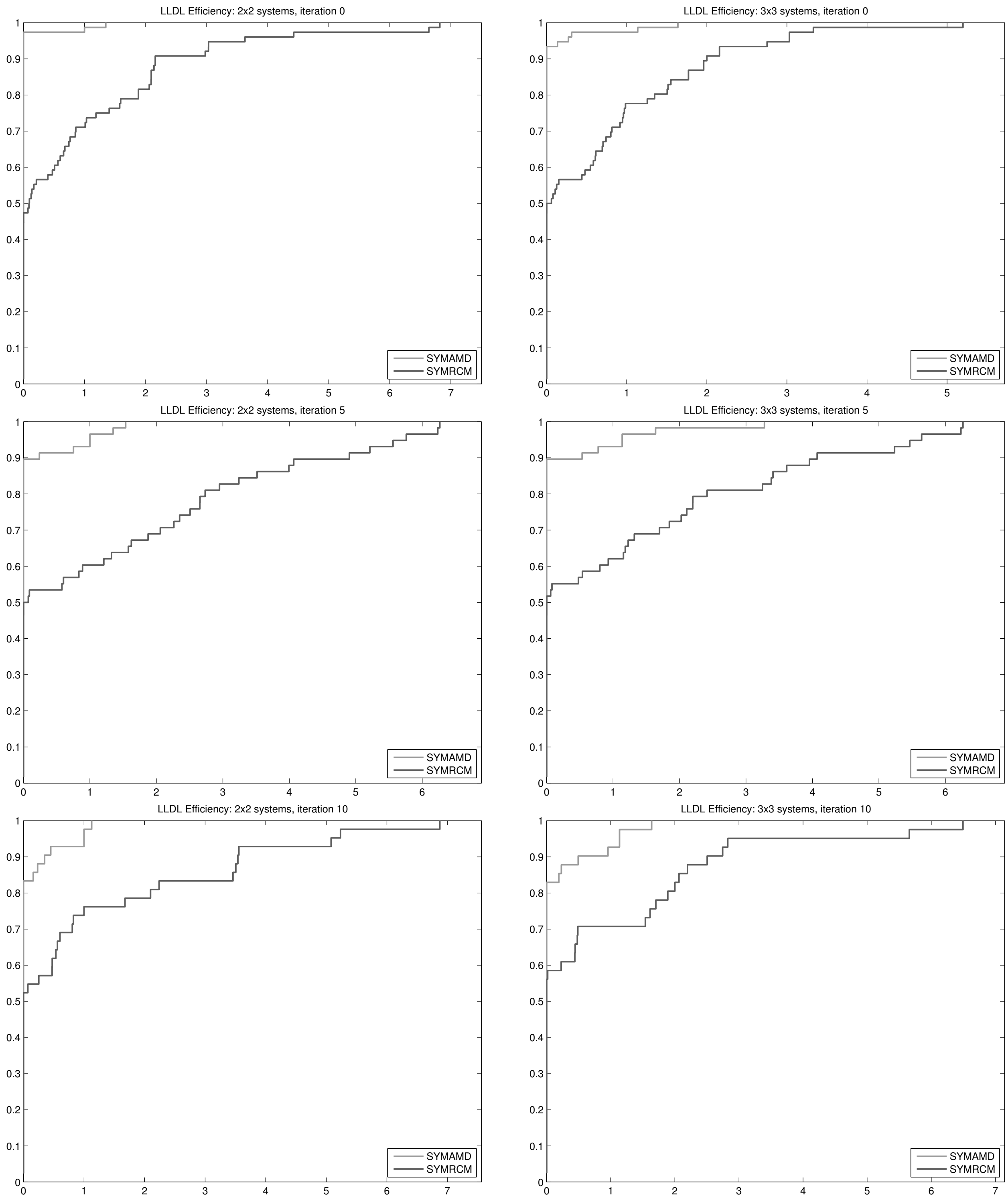

Figure 7.4: Efficiency profiles comparing the SYMAMD and SYMRCM orderings using a limited-memory preconditioner with $p = 20$. The left column corresponds to 2×2 systems and the right column to 3×3 systems.

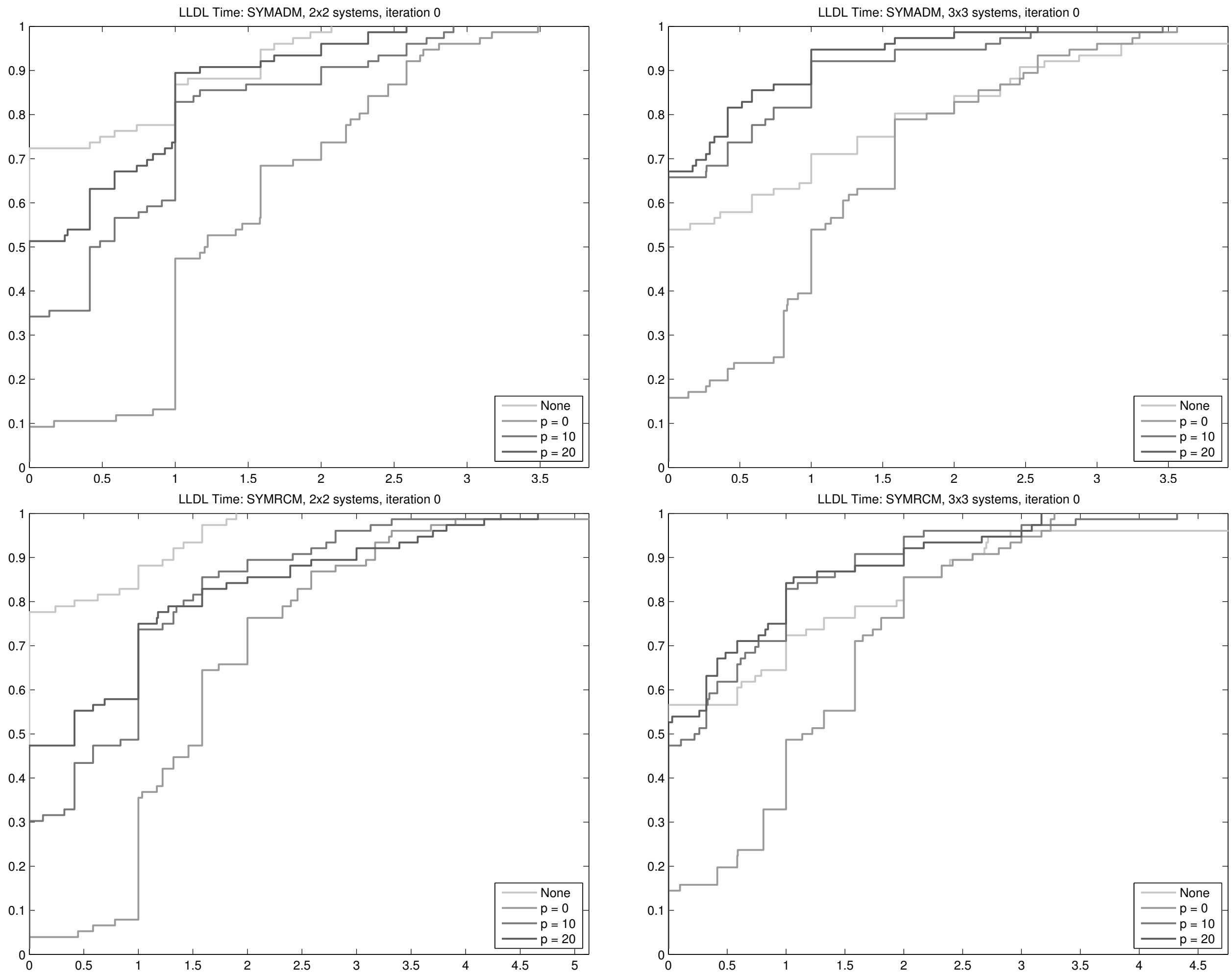

Figure 7.5: Time profiles for 2×2 and 3×3 systems (6.4) at iteration 0 of the interior-point method. The variants compared are MINRES without preconditioner, and with limited-memory preconditioners corresponding to $p = 0$, 10 and 20. The top row corresponds to SYMAMD ordering and the bottom row to SYMRCM ordering.

shift, i.e., $\alpha = 10^{-3}$. The growth factor is at its largest when λ is between 3 and 5, i.e., when the shifted Laplacian is the most indefinite. Nevertheless g remains modest. Results with $p = 0$ and $p = 20$ are similar.

Table 8.2: Growth factor observed in incomplete factors of a shifted two-dimensional Laplacian with $p = 10$.

λ	1	2	3	4	5	6	7
g	3.0e+01	5.2e+01	4.4e+02	3.7e+02	4.4e+02	5.2e+01	3.0e+01
α	1.0e−03	1.0e−03	1.0e−03	1.0e−03	1.0e−03	1.0e−03	1.0e−03

The incomplete Bunch-Kaufman factorization of Greif and Liu (2013) is a sensible choice when the input matrix is symmetric indefinite but not SQD. The tests above and Axelsson and Neytcheva (2003) suggest that LLDL may also have certain advantages in this general setting. A proper comparison of the two implementations is beyond the scope of the present paper but is in progress jointly with the authors of iLDL.

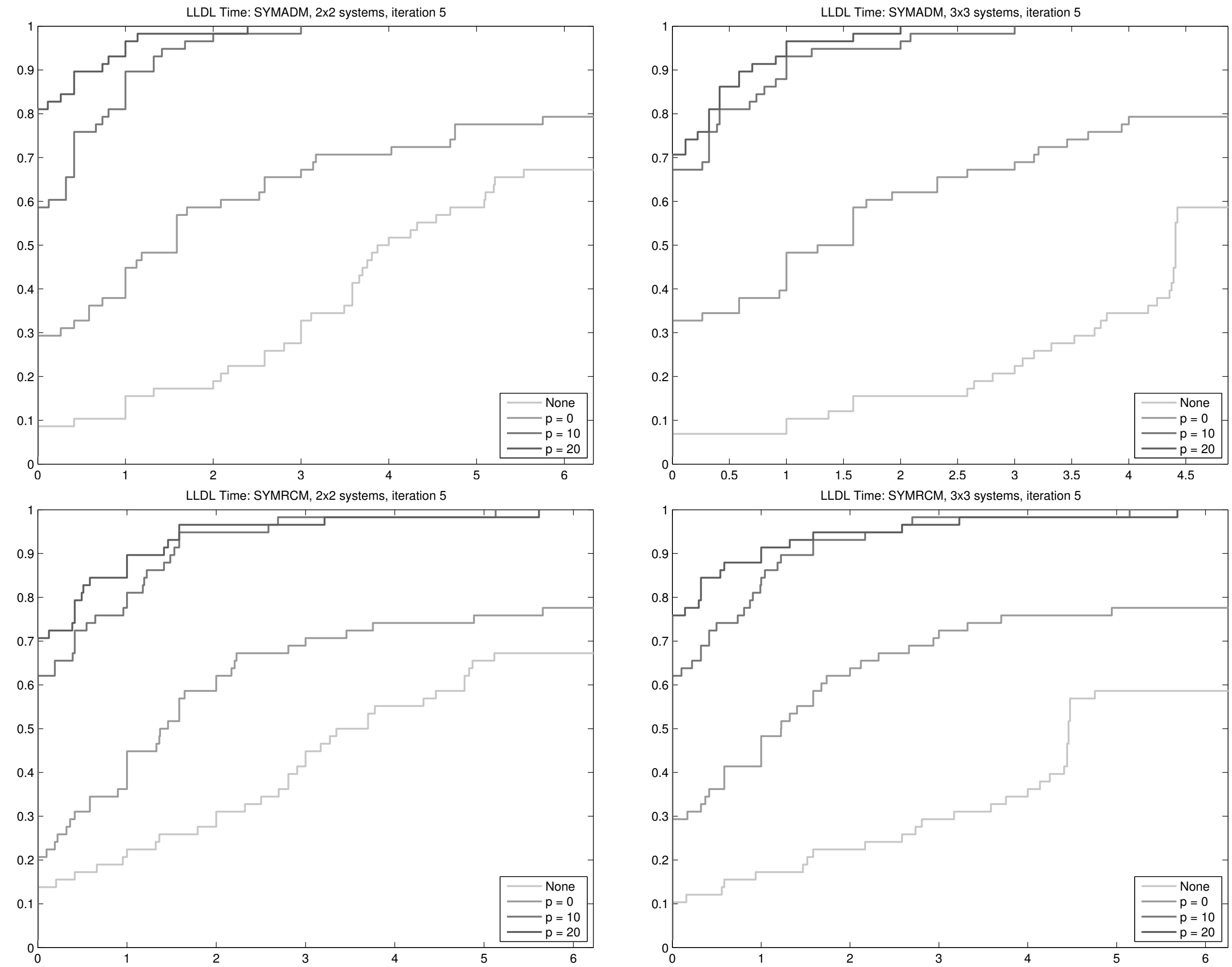

Figure 7.6: Time profiles for 2×2 and 3×3 systems (6.4) at iteration 5 of the interior-point method. The variants compared are MINRES without preconditioner, and with limited-memory preconditioners corresponding to $p = 0$, 10 and 20. The top row corresponds to SYMAMD ordering and the bottom row to SYMRCM ordering.

9 Concluding Remarks

The limited-memory Cholesky factorization of Lin and Moré (1999) generalizes elegantly to SQD systems. Preliminary experiments suggest that limited-memory LDL^T preconditioners are effective on systems arising from a regularized interior-point method, especially if using a SYMAMD ordering of the 3×3 block system analyzed by Greif et al. (2012). In our framework, the preconditioned system is still symmetric and indefinite, but not SQD, and is therefore a good candidate for MINRES.

It is possible to ensure that the preconditioned system is SQD, e.g., by choosing $\mathbf{L}$ block diagonal. This may be achieved by selecting $\mathbf{L}_{11}$ and $\mathbf{L}_{22}$ as incomplete Choleksy factors of $\mathbf{M}$ and $\mathbf{N}$, respectively, and setting $\mathbf{L} = \mathrm{blkdiag}(\mathbf{L}_{11}, \mathbf{L}_{22})$ and $|\mathbf{D}| = \mathbf{I}$, or otherwise discarding elements of $\mathbf{L}$ that lie outside of the diagonal blocks.

SQD linear systems arise in other applications, such as the discretization of the stabilized Stokes equations (Elman et al., 2005), and evaluating the quality of limited-memory preconditioners in such contexts would be worthwhile. Arioli and Orban (2013) describe several families of iterative methods that exploit the SQD structure and are based on Schur-complement reductions of (1.1). As it turns out, it is legitimate to apply the conjugate gradient method to an SQD system with an appropriate initial guess. As Arioli and Orban

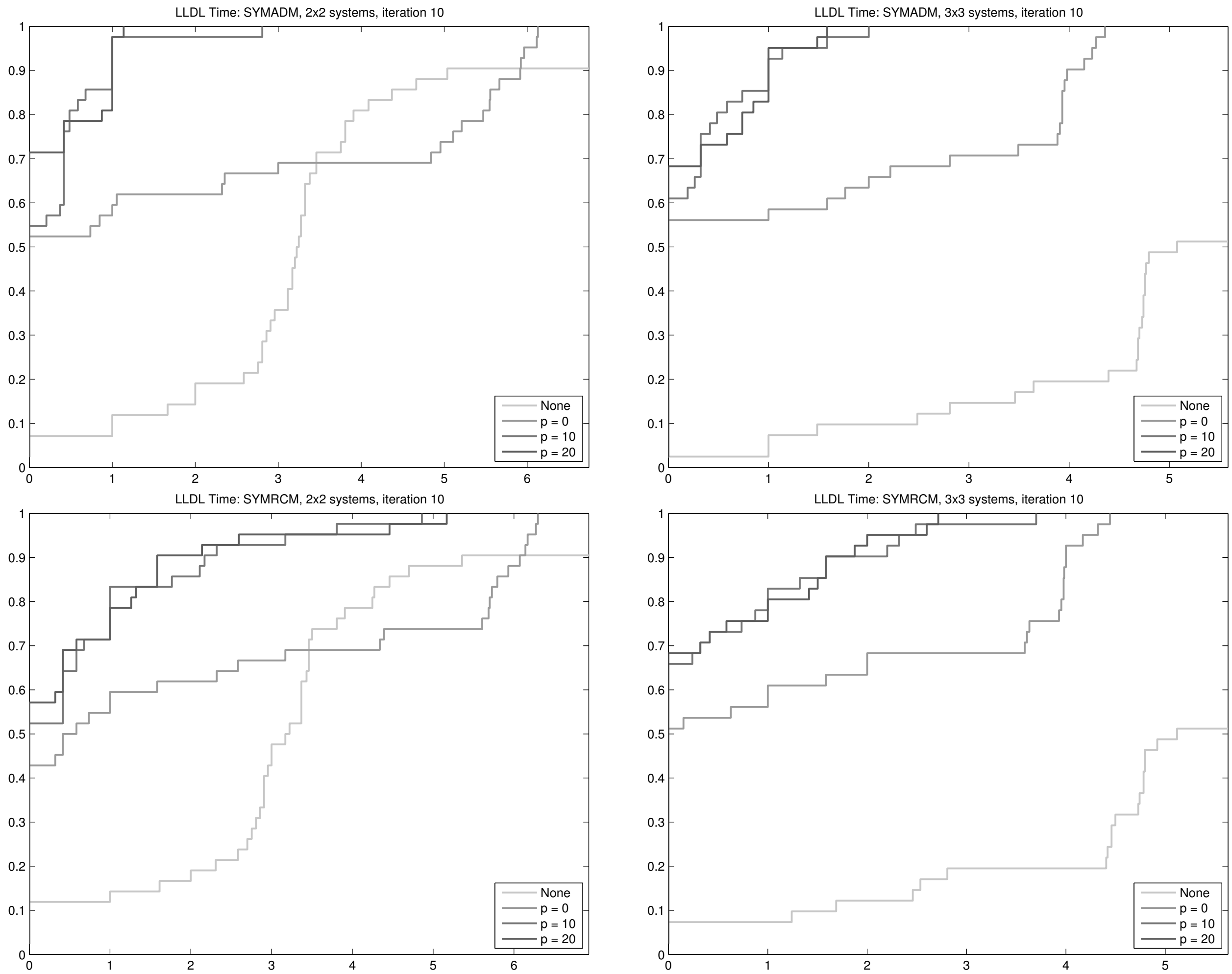

Figure 7.7: Time profiles for 2×2 and 3×3 systems (6.4) at iteration 10 of the interior-point method. The variants compared are MINRES without preconditioner, and with limited-memory preconditioners corresponding to $p = 0$, 10 and 20. The top row corresponds to SYMAMD ordering and the bottom row to SYMRCM ordering.

establish, Schur-complement-based methods for SQD systems provably perform half of the work as both the conjugate gradient method and MINRES provided both $\mathbf{M}$ and $\mathbf{N}$ can be inverted. When working with the full 3×3 systems, the limited-memory preconditioners of the present paper appear to be viable options.

References

P. Amestoy, T. Davis, and I. Duff. An approximate minimum degree ordering algorithm. *SIAM Journal on Matrix Analysis and Applications*, 17(4):886–905, 1996. DOI: 10.1137/S0895479894278952.

M. Arioli and D. Orban. Iterative methods for symmetric quasi-definite linear systems—Part I: Theory. *Les Cahiers du GERAD* G–2013–32, GERAD, Montréal, QC, Canada, 2013.

O. Axelsson. *Iterative Solution Methods*. Cambridge University Press, Cambridge, UK, 1994.

O. Axelsson and M. Neytcheva. Preconditioning methods for linear systems arising in constrained optimization problems. *Numerical Linear Algebra with Applications*, 10:3–31, 2003. DOI: 10.1002/nla.310.

J. R. Bunch and L. Kaufman. Some stable methods for calculating inertia and solving symmetric linear systems. *Mathematics of Computation*, 31(137):163–179, 1977.

J. J. Buoni. A stable method for the incomplete factorization of H-matrices. *Linear Algebra and its Applications*, 129: 143–154, 1990. DOI: 10.1016/0024-3795(90)90302-S.

W.M. Chan and A. George. A linear time implementation of the reverse Cuthill-McKee algorithm. *BIT Numerical Mathematics*, 20(1):8–14, 1980. DOI: 10.1007/BF01933580.

E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 1969 24th national conference*, ACM '69, pages 157–172, New York, NY, USA, 1969. ACM. DOI: 10.1145/800195.805928.

E. Dolan and J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming, Series B*, 91:201–213, 2002. DOI: 10.1007/s101070100263.

H. Elman, D. Silvester, and A. Wathen. *Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics*. Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, UK, 2005.

A. Forsgren, Ph. E. Gill, and M. H. Wright. Interior methods for nonlinear optimization. *SIAM Review*, 44(4):525–597, 2002. DOI: 10.1137/S0036144502414942.

M. P. Friedlander and D. Orban. A primal-dual regularized interior-point method for convex quadratic programs. *Mathematical Programming Computation*, 4(1):71–107, 2012. DOI: 10.1007/s12532-012-0035-2.

M. P. Friedlander and E. van Den Berg. Spot—a linear-operator toolbox, November 2013. URL `www.cs.ubc.ca/labs/scl/spot`.

A. George and KH. Ikramov. On the condition of symmetric quasi-definite matrices. *SIAM Journal on Matrix Analysis and Applications*, 21(3):970–977, 2000. DOI: 10.1137/S0895479898333247.

P. E. Gill, M. A. Saunders, and J. R. Shinnerl. On the stability of Cholesky factorization for symmetric quasidefinite systems. *SIAM Journal on Optimization*, 17(1):35–46, 1996. DOI: 10.1137/S0895479893252623.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Series in the Mathematical Sciences. Johns Hopkins, Baltimore, third edition, 1996.

N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr and SifDec, a Constrained and Unconstrained Testing Environment, revisited. *ACM Transactions on Mathematical Software*, 29(4):373–394, 2003.

N. I. M. Gould, D. Orban, and Ph. L. Toint. Numerical methods for large-scale nonlinear optimization. *Acta Numerica*, 14:299–361, 2005. DOI: 10.1017/S0962492904000248.

C. Greif and P. Liu. sym-ildl: Incomplete LDL^T factorizations of indefinite symmetric matrices, December 2013. URL `www.cs.ubc.ca/~inutard/html`.

C. Greif, E. Moulding, and D. Orban. Bounds on the eigenvalues of block matrices arising from interior-point methods. *Les Cahiers du GERAD* G–2012–42, GERAD, Montréal, Québec, Canada, 2012. To appear in SIAM Journal on Optimization.

I. Gustafsson. A class of first order factorization methods. *BIT*, 18(2):142–156, 1978. DOI: 10.1007/BF01931691.

N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. Number OT80 in Other Titles in Applied Mathematics. SIAM, Philadelphia, PA, second edition, 2002. DOI: 10.1137/1.9780898718027.

HSL. *A collection of Fortran codes for large-scale scientific computation*. `www.hsl.rl.ac.uk`, 2013.

M. T. Jones and P. E. Plassmann. An improved incomplete Cholesky factorization. *ACM Transactions on Mathematical Software*, 21(1):5–17, 1995. DOI: 10.1145/200979.200981.

I. E. Kaporin. High quality preconditioning of a general symmetric positive definite matrix based on its $U^TU + U^TR + R^TU$ decomposition. *Numerical Linear Algebra with Applications*, 5(6):483–509, 1998. DOI: 10.1002/(SICI)1099-1506(199811/12)5:6<483::AID-NLA156>3.0.CO;2-7.

N. Li, Y. Saad, and E. Chow. Crout versions of ILU for general sparse matrices. *SIAM Journal on Scientific Computing*, 25(2):716–728, 2003. DOI: 10.1137/S1064827502405094.

Na Li and Yousef Saad. Crout versions of ILU factorization with pivoting for sparse symmetric matrices. *Electronic Transactions on Numerical Analysis*, 20:75–85, 2005.

C.-J. Lin and J. J. Moré. Incomplete Cholesky factorizations with limited memory. *SIAM Journal on Scientific Computing*, 21(1):24–45, 1999. DOI: 10.1137/S1064827597327334.

T. A. Manteuffel. Shifted incomplete Cholesky factorization. In I. S. Duff and G. W. Stewart, editors, *Sparse Matrix Proceedings*, pages 41–61, Philadelphia, PA, 1979. SIAM.

T. A. Manteuffel. An incomplete factorization technique for positive definite linear systems. *Mathematics of Computation*, 34(150):473–497, 1980.

S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4): 575–601, 1992. DOI: 10.1137/0802028.

J. A. Meijerink and H. A. van der Vorst. An iterative solution method for linear equations systems of which the coefficient matrix is a symmetric M-matrix. *Mathematics of Computation*, 31(137):148–162, 1977.

A. Messaoudi. On the stability of the incomplete LU-factorizations and characterizations of H-matrices. *Numerische Mathematik*, 69(3):321–331, 1995. DOI: 10.1007/s002110050095.

N. Munksgaard. Solving sparse symmetric sets of linear equations by preconditioned conjugate gradients. *ACM Transactions on Mathematical Software*, 6(2):206–219, 1980. DOI: 10.1145/355887.355893.

J. M. Ortega. *Introduction to Parallel and Vector Solution of Linear Systems*. Fontiers of Computer Science. Springer Science+Business Media, New York, 1988. Originally published by Plenum Press.

C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975. DOI: 10.1137/0712047.

Y. Saad. ILUT: A dual threshdol incomplete LU factorization. *Numerical Linear Algebra with Applications*, 1(4): 387–402, 1994. DOI: 10.1002/nla.1680010405.

J. Scott and M. Tůma. HSL_MI28: an efficient and robust limited-memory incomplete Cholesky factorization code. Preprint RAL-P-2013-004, Rutherford Appleton Laboratory, Didcot, UK, 2013.

M. Tismenetsky. A new preconditioning technique for solving large sparse linear systems. *Linear Algebra and its Applications*, 154–156:331–353, 1991. DOI: 10.1016/0024-3795(91)90383-8.

M. Tůma. A note on the LDL^T decomposition of matrices from saddle-point problems. *SIAM Journal on Matrix Analysis and Applications*, 23(4):903–915, 2002. DOI: 10.1137/S0895479897321088.

R. J. Vanderbei. Symmetric quasi-definite matrices. *SIAM Journal on Optimization*, 5(1):100–113, 1995. DOI: 10.1137/0805005.