Computational Optimization and Applications (2022) 83:727–757
https://doi.org/10.1007/s10589-022-00424-5

General-purpose preconditioning for regularized interior point methods

Jacek Gondzio[1] · **Spyridon Pougkakiotis**[2] · **John W. Pearson**[1]

Received: 14 July 2021 / Accepted: 13 October 2022 / Published online: 14 November 2022

Abstract
In this paper we present general-purpose preconditioners for regularized augmented systems, and their corresponding normal equations, arising from optimization problems. We discuss positive definite preconditioners, suitable for CG and MINRES. We consider "sparsifications" which avoid situations in which eigenvalues of the preconditioned matrix may become complex. Special attention is given to systems arising from the application of regularized interior point methods to linear or nonlinear convex programming problems.

Keywords Preconditioning · Krylov subspace methods · Interior point methods · Regularization · Saddle point systems · Convex optimization

1 Introduction

In this paper, we are concerned with applying Krylov-subspace methods for the efficient solution of systems of the following form:

$$\underbrace{\begin{bmatrix} -(Q+\rho I_n) & A^\top \\ A & \delta I_m \end{bmatrix}}_{K} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix}, \tag{1}$$

✉ Spyridon Pougkakiotis
spyridon.pougkakiotis@yale.edu

Jacek Gondzio
j.gondzio@ed.ac.uk

John W. Pearson
j.pearson@ed.ac.uk

[1] University of Edinburgh, Edinburgh, Scotland

[2] Yale University, New Haven, USA

where $A \in \mathbb{R}^{m\times n}$ (with $m \leq n$), $Q \geq 0 \in \mathbb{R}^{n\times n}$, I_n is the identity matrix of size n, and $\delta,\ \rho > 0$. Such systems arise in a plethora of applications [6], which go far beyond optimization. However, in this paper we restrict the discussion to the case of regularized systems arising in Interior Point Methods (IPMs) for optimization [1, 3, 19, 34, 40, 43, 46]. Due to the potential large dimensions of the systems, they are often solved by means of iterative techniques, usually from the family of Krylov-subspace methods [21]. To guarantee efficiency of such methods the possibly ill-conditioned system (1) often needs to be appropriately preconditioned and, indeed, there exists a rich literature which addresses the issue (see the discussions in [4, 6, 8, 9, 14–16], and the references therein).

Multiple preconditioning approaches have been developed in the literature, used to accelerate the associated iterative methods. These can be divided into positive definite (e.g. [6, 10, 20, 30, 32, 33, 45]) and indefinite ones (e.g. [15, 25, 26, 30]). The latter are often employed within long-recurrence non-symmetric solvers (such as the Generalized Minimal RESidual method [42]), while the former can be used within short-recurrence methods (such as the MINimal RESidual method [35]). A comprehensive study of saddle point systems and their associated "optimal" preconditioners can be found in [6]. Indefinite preconditioners are significantly more difficult to analyze and a simple spectral analysis is not sufficient to deduce their effectiveness (see [22]). On the other hand, positive definite preconditioners are often easier to analyze, and the eigenvalues of the preconditioned matrices allow one to theoretically compare different preconditioning approaches.

In this paper, we are focused on systems arising from the application of regularized interior point methods for the solution of an arbitrary convex programming problem. In this case, Q represents the Hessian of the primal barrier problem's objective function (or the Hessian of the Lagrangian in the nonlinear programming case), A represents the constraint matrix (or the Jacobian of the constraints in the nonlinear programming case), while ρ and δ are the primal and dual regularization parameters, respectively. We note that the IPM may contribute a term to the (1, 1) or the (2, 2) block of (1), depending on the form of the constraints and non-negativity variables. Here we assume that the term is added in the (1, 1) block. For example, the matrix Q may be written as $Q := H + \Xi^{-1}$, where H is the Hessian of the Lagrangian and $\Xi := XZ^{-1}$ is a diagonal IPM scaling matrix (assuming $x,\ z$ are the primal and dual non-negative variables, respectively, while $X,\ Z$ denote the diagonal matrices with diagonal entries taken from vectors $x,\ z$, respectively) which originates from the use of the logarithmic barrier.

We present positive definite preconditioning approaches that can be used within MINRES [35] or the Conjugate Gradient method [23], and we provide some basic spectral analysis results for the associated preconditioned systems. More specifically, we consider preconditioners which are derived by "sparsifications" of system (1), that is, by dropping specific entries from sparse matrices Q and A, thus making them more sparse and hence easier to factorize. Various such approaches have been proposed to date and include: preconditioners which exploit an early guess of a basic–nonbasic partition of variables to drop columns from A [33], constraint preconditioners [9, 14–16], inexact constraint preconditioners [8] which drop specific entries in the matrices Q and A, and of course a plethora of preconditioners which

involve various levels of incomplete Cholesky factorizations of the matrix in (1), see for example [10]. The literature on preconditioners is growing rapidly and we refer the interested reader to [5, 6, 12, 37, 48] and the references therein, for a detailed discussion.

We consider dropping off-diagonal entries of Q, but restrict the elimination of entries in A only to the removal of complete columns. Additionally, we consider sparsifying parts of rows of the Schur complement corresponding to system (1). Such a strategy guarantees that we avoid situations in which eigenvalues of the preconditioned matrix may become complex (such as those employed in [8]), which as a consequence would have required us to employ non-symmetric Krylov methods.

In order to construct the preconditioners, by following [7], we take advantage of the properties of the logarithmic barrier, that allow us to know in advance which columns of the problem matrix are important and which are less influential. In particular, assuming the aforementioned representation of Q as $Q = H + \Xi^{-1}$, the logarithmic barrier indicates which variables of the problem are likely to be inactive in the solution. More precisely, the variables are naturally split into "basic"–$\mathcal{B}$ (not in the simplex sense), "non-basic"–$\mathcal{N}$, and "undecided"–$\mathcal{U}$. Hence, as IPMs progress towards optimality, we expect the following partition of the diagonal barrier matrix Ξ^{-1}:

$$\forall j \in \mathcal{N} : \left(\Xi^{(j,j)}\right)^{-1} = \boldsymbol{\Theta}\left(\mu^{-1}\right), \quad \forall j \in \mathcal{B} : \left(\Xi^{(j,j)}\right)^{-1} = \boldsymbol{\Theta}(\mu),$$
$$\forall j \in \mathcal{U} : \left(\Xi^{(j,j)}\right)^{-1} = \boldsymbol{\Theta}(1),$$

where μ is the barrier parameter (and is such that $\mu \to 0$), $\mathcal{N}$, $\mathcal{B}$, and $\mathcal{U}$ are mutually disjoint, and $\mathcal{N} \cup \mathcal{B} \cup \mathcal{U} = \{1, \ldots, n\}$, while $\boldsymbol{\Theta}(\cdot)$ denotes that two positive quantities are of the same order of magnitude (see the notation section at the end of the introduction). Given the large magnitude of the diagonal elements of Q for any $j \in \mathcal{N}$ (assuming μ is close to zero), we expect that the corresponding columns of A (i.e. $A^{(:,\mathcal{N})}$) will not contribute important information, and thus can be set to zero when constructing a preconditioner for (1). In [7], the Hessian was approximated by its diagonal. In this paper, we extend this work by allowing the utilization of non-diagonal Hessian information. More specifically, we showcase how to analyze, construct, and apply the inverse of preconditioners in which we only drop non-diagonal elements of Q corresponding to diagonal elements in $\mathcal{N}$. We should note at this point that such a splitting of Q occurs in other second-order optimization methods as well, such as those based on augmented Lagrangian strategies (e.g. see [27]).

Furthermore, we discuss some approaches for dealing with problems for which the matrix A may contain a subset of dense columns or rows. Any dense column or row in A can pose great difficulties when trying to factorize the associated saddle-point matrix (or a preconditioner approximating it). Thus, it is desirable to alleviate the dangers of such columns or rows, by appropriate "sparsifications" of the preconditioner, allowing us to reduce the memory requirements of applying its inverse.

All such "sparsifications" are captured in a general result presented in Sect. 2 which provides the spectral analysis of an appropriate preconditioned normal equations' matrix. The main theorem sheds light on consequences of sparsifying

rows of the normal equations corresponding to (1), or dropping columns of A, and demonstrates that the former might produce a larger number of non-unit eigenvalues. In Sect. 3, these normal equation approximations are utilized to construct positive definite block-diagonal preconditioners for the saddle point system in (1), and the spectral properties of the resulting preconditioned matrices are also discussed. Additionally, an alternative saddle-point preconditioner based on an $LDL^\top$ decomposition is presented.

All of the preconditioning approaches discussed are compared numerically on saddle-point systems arising from the application of a regularized IPM for the solution of real-life linear and convex quadratic programming problems. In particular, we present some numerical results on certain test problems taken from the Netlib (see [31]) and the Maros–Mészáros (see [29]) collections. Then, we test the preconditioners on examples of Partial Differential Equation (PDE) optimal control problems. All preconditioning approaches have been implemented within an Interior Point-Proximal Method of Multipliers (IP-PMM) framework, which is a polynomially convergent primal-dual regularized IPM, based on the developments in [40, 41]. A robust implementation is provided.

It is worth stressing that the proposed preconditioners are *general* and do not assume the knowledge of special structures which might be present in the matrices Q and A (such as block-diagonal, block-angular, network, PDE-induced, and so on). Therefore they may be applied within general-purpose IPM solvers for linear and convex quadratic programming problems.

Notation. Throughout this paper we use lowercase Roman and Greek letters to indicate vectors and scalars. Capitalized Roman fonts are used to indicate matrices. Superscripts are used to denote the components of a vector/matrix. Sets of indices are denoted by caligraphic capital fonts. For example, given $M \in \mathbb{R}^{m\times n}$, $v \in \mathbb{R}^n$, $\mathcal{R} \subseteq \{1,\dots,m\}$, and $\mathcal{C} \subseteq \{1,\dots,n\}$, we set $v^{\mathcal{C}} := (v^i)_{i\in\mathcal{C}}$ and $M^{(\mathcal{R},\mathcal{C})} := (m^{(i,j)})_{i\in\mathcal{R},j\in\mathcal{C}}$, where v^i is the i-th entry of v and $m^{(i,j)}$ the (i,j)-th entry of M. Additionally, the full set of indices is denoted by a colon. In particular, $M^{(:,\mathcal{C})}$ denotes all columns of M with indices in $\mathcal{C}$. Given a matrix M, we denote the diagonal matrix with the same diagonal elements as M by $\mathrm{Diag}(M)$. We use $\lambda_{\min}(B)$ ($\lambda_{\max}(B)$, respectively) to denote the minimum (maximum) eigenvalue of an arbitrary square matrix B with real eigenvalues. Similarly, $\sigma_{\max}(B)$ denotes the maximum singular value of an arbitrary rectangular matrix B. We use $0_{m,n}$ to denote a matrix of size $m\times n$ with entries equal to 0. Furthermore, we use I_n to indicate the identity matrix of size n. For any finite set $\mathcal{A}$, we denote by $|\mathcal{A}|$ its cardinality. Finally, given two positive functions $T, f : \mathbb{R}_+ \mapsto \mathbb{R}_+$, we write $T(x) = \boldsymbol{\Theta}(f(x))$ if these functions are of the same order of magnitude, that is, there exist constants $c_1,\ c_2 > 0$ and some $x_0 \geq 0$ such that $c_1 f(x) \leq T(x) \leq c_2 f(x)$, for all $x \geq x_0$.

Structure of the article. The rest of this paper is organized as follows. In Sect. 2 we present some preconditioners suitable for the normal equations. Then, in Sect. 3, we adapt these preconditioners to regularized saddle point systems. Subsequently, in Sect. 4 we focus on saddle point systems arising from the application of regularized IPMs to convex programming problems, and present some numerical results. Finally, in Sect. 5, we deliver our conclusions.

2 Regularized normal equations

We begin by defining the regularized normal equations matrix (or Schur complement) $M := AGA^\top + \delta I_m \in \mathbb{R}^{m\times m}$, corresponding to (1), where $G := (Q+\rho I_n)^{-1} \succ 0 \in \mathbb{R}^{n\times n}$. In this section, we derive and analyze preconditioning approaches for M. As we have already mentioned in the introduction, we achieve a simplification of the preconditioner by setting to zero certain columns of the matrix A (and consequently sparsifying the corresponding rows and columns of Q), as well as by sparsifying certain rows of the matrix M.

More specifically, we define two integers, k_c and k_r, such that $0 \le k_c \le n$ and $0 \le k_r \le m$. The former counts the number of columns of A (and corresponding columns and rows of Q) that are set to zero (that are sparsified, respectively), while the latter counts the number of rows of the matrix M that are sparsified. At this point, we assume that we have been given these columns or rows, but we later specify how these can be chosen (see Remark 1 and Sect. 4). In order to highlight these given columns and rows, we assume that we are given two permutation matrices; a column permutation $\mathscr{P}_c \in \mathbb{R}^{n\times n}$, and a row permutation $\mathscr{P}_r \in \mathbb{R}^{m\times m}$. Applied to the constraint matrix A in (1), these permutations bring all the columns and rows which will need to be treated specially to the leading positions of columns and rows of $\mathscr{P}_r A \mathscr{P}_c$, respectively.

Given the previous permutation matrices, let us firstly define an approximation to the matrix Q. In particular, we approximate Q by the following block-diagonal and positive semi-definite matrix:

$$\widehat{Q} := \mathscr{P}_c \begin{bmatrix} \widehat{Q}_1 & 0_{k_c,(n-k_c)} \\ 0_{(n-k_c),k_c} & \widehat{Q}_2 \end{bmatrix} \mathscr{P}_c^\top, \quad \text{assuming} \quad Q \equiv \mathscr{P}_c \begin{bmatrix} Q_1 & Q_3^\top \\ Q_3 & Q_2 \end{bmatrix} \mathscr{P}_c^\top, \tag{2}$$

where $\widehat{Q}_1 = Q_1$ or $\widehat{Q}_1 = \mathrm{Diag}(Q_1)$, and $\widehat{Q}_2 = Q_2$ or $\widehat{Q}_2 = \mathrm{Diag}(Q_2)$ (both cases are treated concurrently). The column permutation $\mathscr{P}_c$ reorders symmetrically both rows and columns of the matrix Q in (1), and places the k_c columns and rows which will be sparsified at the leading (1, 1) block of the permuted version of the matrix Q. Given this approximation of Q, let us define an approximate normal equations' matrix that will be of interest when analyzing the spectral properties of the preconditioned matrices derived in this paper:

$$\widehat{M} := A\widehat{G}A^\top + \delta I_m, \quad \widehat{G} := \left(\widehat{Q}+\rho I_n\right)^{-1} \equiv \mathscr{P}_c \begin{bmatrix} \left(\widehat{Q}_1+\rho I_{k_c}\right)^{-1} & 0_{k_c,(n-k_c)} \\ 0_{(n-k_c),k_c} & \left(\widehat{Q}_2+\rho I_{n-k_c}\right)^{-1} \end{bmatrix} \mathscr{P}_c^\top. \tag{3}$$

In what follows, we derive a preconditioner for the approximate normal equations' matrix $\widehat{M}$. We should note that system (1) is solved using the normal equations only if Q is diagonal (due to obvious numerical considerations), in which case $Q \equiv \widehat{Q}$, and thus $M \equiv \widehat{M}$. If this is not the case, we would like to derive a preconditioner for the approximate normal equations' matrix $\widehat{M}$. Later on, and in particular in Sect. 3, this is utilized to derive and analyze a preconditioner for the matrix K, defined in (1).

We proceed by introducing some notation, for convenience of exposition. Given the definitions in (2), (3), we can write:

$$B := \mathscr{P}_r A \widehat{G}^{\frac{1}{2}} \mathscr{P}_c = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix},$$

where $\mathscr{P}_r$ is a given row-permutation matrix, $B_{11} \in \mathbb{R}^{k_r \times k_c}$, $B_{12} \in \mathbb{R}^{k_r \times (n-k_c)}$, $B_{21} \in \mathbb{R}^{(m-k_r)\times k_c}$, and $B_{22} \in \mathbb{R}^{(m-k_r)\times(n-k_c)}$. Notice that the aim of the row-permutation matrix $\mathscr{P}_r$, is to bring on top all rows of the matrix $\widehat{M}$ that we are planning to sparsify in order to construct the preconditioner. Let us further introduce the following notation:

$$\mathscr{P}_r \widehat{M} \mathscr{P}_r^\top \equiv \begin{bmatrix} \widehat{M}_{11} & \widehat{M}_{21}^\top \\ \widehat{M}_{21} & \widehat{M}_{22} \end{bmatrix},$$

where $\widehat{M}_{11}$, $\widehat{M}_{21}$, and $\widehat{M}_{22}$ are defined as:

$$\begin{aligned} \widehat{M}_{11} &:= B_{11}B_{11}^\top + B_{12}B_{12}^\top + \delta I_{k_r} \in \mathbb{R}^{k_r \times k_r}, \\ \widehat{M}_{21} &:= B_{21}B_{11}^\top + B_{22}B_{12}^\top \in \mathbb{R}^{(m-k_r)\times k_r}, \\ \widehat{M}_{22} &:= B_{21}B_{21}^\top + B_{22}B_{22}^\top + \delta I_{m-k_r} \in \mathbb{R}^{(m-k_r)\times(m-k_r)}. \end{aligned}$$

In what follows, we present two preconditioning strategies for the matrix $\widehat{M}$. Both approaches exploit the sparsification of the matrix $\widehat{M}$. The first approach relies on a Cholesky decomposition of a sparsified matrix, while the second approach is based on an $LDL^\top$ decomposition of a sparsified augmented system matrix, which is used to implicitly derive a preconditioner for $\widehat{M}$.

2.1 A Cholesky-based preconditioner

Our first proposal is to consider preconditioning $\mathscr{P}_r \widehat{M} \mathscr{P}_r^\top$ with the following matrix:

$$P_{NE,(k_c,k_r)} := \begin{bmatrix} \widehat{M}_{11} & 0_{k_r,(m-k_r)} \\ 0_{(m-k_r),k_r} & \widetilde{M}_{22} \end{bmatrix}, \qquad \widetilde{M}_{22} := \widehat{M}_{22} - B_{21}B_{21}^\top. \tag{4}$$

The notation $P_{NE,(k_c,k_r)}$ signifies that this is a preconditioner for the normal equations, in which we drop k_c columns from the matrix A and sparsify k_r rows of the matrix $\widehat{M}$. Notice that if $k_c = 0$ (that is, we only sparsify certain rows of $\widehat{M}$ to construct the preconditioner), we can write $B \equiv \mathscr{P}_r A \widehat{G}^{\frac{1}{2}} = \begin{bmatrix} B_{12} \\ B_{22} \end{bmatrix}$, while B_{11}, B_{21} are zero-dimensional, and hence absent. In this case, we have

$$P_{NE,(0,k_r)} := \begin{bmatrix} \widehat{M}_{11} & 0_{k_r,(m-k_r)} \\ 0_{(m-k_r),k_r} & \widehat{M}_{22} \end{bmatrix}.$$

On the other hand, if $k_r = 0$ (that is, we only drop k_c columns from A to construct the preconditioner), we have $B \equiv \begin{bmatrix} B_{21} & B_{22} \end{bmatrix}$, and $B_{11},\ B_{12}$ are absent. Then, we obtain

$$P_{NE,(k_c,0)} = \widetilde{M}_{22}.$$

Notice that the latter is obtained since $\widehat{Q}$ is block-separable (with respect to the permutation $\mathscr{P}_c$), and thus dropping the respective k_c columns of A results in dropping $B_{21}B_{21}^\top$. For simplicity of notation, for the rest of this subsection we set $P_{NE,(k_c,k_r)} \equiv P_{NE}$.

In the following theorem, we analyze the spectrum of the preconditioned matrix $P_{NE}^{-1}\mathscr{P}_r\widehat{M}\mathscr{P}_r^\top$, with respect to the spectrum of the associated matrices.

Theorem 1 *The preconditioned matrix* $P_{NE}^{-1}\mathscr{P}_r\widehat{M}\mathscr{P}_r^\top$ *has at least* $\max\{m-(2k_r+k_c),0\}$ *eigenvalues at* 1. *If* $k_c > 0$ *and* $k_r > 0$, *all remaining eigenvalues lie in the following interval*:

$$I_{k_c,k_r} := \left[\frac{\delta}{\delta + \max\left\{\lambda_{\max}(B_{11}B_{11}^\top + B_{12}B_{12}^\top), \lambda_{\max}(B_{22}B_{22}^\top)\right\}}, 2 + \frac{\lambda_{\max}(B_{21}B_{21}^\top)}{\delta + \lambda_{\min}(B_{22}B_{22}^\top)}\right].$$

If $k_c > 0$ *and* $k_r = 0$, *the previous interval reduces to*

$$I_{k_c} := \left[1, 1 + \frac{\lambda_{\max}(B_{21}B_{21}^\top)}{\delta + \lambda_{\min}(B_{22}B_{22}^\top)}\right],$$

while if $k_r > 0$ *and* $k_c = 0$, *we obtain*

$$I_{k_r} := \left[\frac{\delta}{\delta + \max\left\{\lambda_{\max}(B_{12}B_{12}^\top), \lambda_{\max}(B_{22}B_{22}^\top)\right\}}, 2\right].$$

Proof Given an arbitrary eigenvalue λ (which must be positive since $P_{NE} \succ 0$ and $\widehat{M} \succ 0$) corresponding to a unit eigenvector v, let us write the generalized eigenproblem as:

$$\begin{bmatrix} \widehat{M}_{11} & \widehat{M}_{21}^\top \\ \widehat{M}_{21} & \widehat{M}_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \lambda \begin{bmatrix} \widehat{M}_{11}v_1 \\ \widetilde{M}_{22}v_2 \end{bmatrix}. \tag{5}$$

We separate the analysis into two cases.

Case 1: Let $v_2 \in \text{Null}(\widehat{M}_{21}^\top)$. Firstly, we notice that:

$$\dim\left(\text{Null}\left(\widehat{M}_{21}^\top\right)\right) = (m - k_r) - \text{rank}\left(\widehat{M}_{21}^\top\right) \geq \max\{m - 2k_r, 0\}.$$

Two sub-cases arise here. For the first sub-case, we notice that if $v_1 \neq 0$, then from positive definiteness of $\widehat{M}_{11}$, combined with the first block equation of (5), we obtain that $\lambda = 1$. In turn, we claim that this implies that $v_2 \in \text{Null}(B_{21}B_{21}^\top)$ and $v_1 \in \text{Null}(\widehat{M}_{21})$. To see this, assume that $v_2 \notin \text{Null}(B_{21}B_{21}^\top)$. Then from the second block equation of (5) we obtain:

$$\widehat{M}_{21}v_1 + \widehat{M}_{22}v_2 = \widetilde{M}_{22}v_2 \;\;\Rightarrow\;\; \widehat{M}_{21}v_1 = -B_{21}B_{21}^\top v_2,$$

where we used the definition of $\widetilde{M}_{22}$. If $v_2 \notin \text{Null}(B_{21}B_{21}^\top)$, this implies that $v_2^\top B_{21}B_{21}^\top v_2 > 0$. The previous equation then yields that

$$v_2^\top \widehat{M}_{21}v_1 = -v_2^\top B_{21}B_{21}^\top v_2 \;\;\Rightarrow\;\; 0 = -v_2^\top B_{21}B_{21}^\top v_2 < 0,$$

which follows from the base assumption (i.e. $v_2 \in \text{Null}(\widehat{M}_{21}^\top)$), and results in a contradiction. Hence, $v_2 \in \text{Null}(B_{21}B_{21}^\top)$. On the other hand, if $v_1 \notin \text{Null}(\widehat{M}_{21})$ then the second block equation yields directly a contradiction, since we have shown that $v_2 \in \text{Null}(B_{21}B_{21}^\top)$.

Next we consider the second sub-case, i.e. $v_1 = 0$. Combined with our base assumption, the first block equation of (5) becomes redundant. From the second block equation of the eigenproblem, and using $v_1 = 0$, we obtain:

$$\begin{aligned} v_2^\top \widehat{M}_{22}v_2 &= \lambda v_2^\top \widetilde{M}_{22}v_2 \\ \Rightarrow\;\; v_2^\top \left(\widetilde{M}_{22} + B_{21}B_{21}^\top\right)v_2 &= \lambda v_2^\top \widetilde{M}_{22}v_2. \end{aligned}$$

Hence we have that:

$$\lambda = 1 + \frac{v_2^\top (B_{21}B_{21}^\top)v_2}{v_2^\top \widetilde{M}_{22}v_2} \leq 1 + \frac{\lambda_{\max}(B_{21}B_{21}^\top)}{\delta + \lambda_{\min}(B_{22}B_{22}^\top)}.$$

All eigenvalues in this case can be bounded by the previous inequality and there will be at most $\text{rank}(B_{21}B_{21}^\top)$ non-unit eigenvalues. On the other hand, if $v_2 \in \text{Null}(B_{21}B_{21}^\top)$, then trivially $\lambda = 1$. This concludes the first case. Notice that this case would occur necessarily if $k_r = 0$, and thus, we obtain the interval I_{k_c}.

Case 2: In this case, we assume that $v_2 \notin \text{Null}(\widehat{M}_{21}^\top)$. In what follows we assume $\lambda \neq 1$ (noting that $\lambda = 1$ would only occur if $v_1 \in \text{Null}(\widehat{M}_{21})$ and $v_2 \in \text{Null}(B_{21}B_{21}^\top)$), and there are at most $2k_r$ such eigenvalues. Given the previous assumption, and using the first block equation in (5), we obtain:

$$v_1 = \frac{1}{\lambda - 1}\widehat{M}_{11}^{-1}\widehat{M}_{21}^\top v_2.$$

Substituting the previous into the second block equation of (5) yields the following generalized eigenproblem:

$$\left(\widehat{M}_{21}\widehat{M}_{11}^{-1}\widehat{M}_{21}^\top + (\lambda - 1)B_{21}B_{21}^\top\right)v_2 = (\lambda - 1)^2\widetilde{M}_{22}v_2, \tag{6}$$

where we used the definitions of $\widetilde{M}_{22}$ and $\widehat{M}_{22}$. Premultiplying (6) by $v_2^\top$ and rearranging yields the following quadratic algebraic equation that λ must satisfy in this case:

$$\lambda^2 + \beta\lambda + \gamma = 0, \tag{7}$$

where

$$\beta := -2 - \frac{v_2^\top B_{21} B_{21}^\top v_2}{v_2^\top \widetilde{M}_{22} v_2}$$

and

$$\gamma := 1 - \frac{v_2^\top \left(\widehat{M}_{21} \widehat{M}_{11}^{-1} \widehat{M}_{21}^\top - B_{21} B_{21}^\top \right) v_2}{v_2^\top \widetilde{M}_{22} v_2}.$$

Let us notice that the smallest eigenvalue is at least as large as $\frac{\delta}{\delta + \max\left\{\lambda_{\max}(B_{11}B_{11}^\top + B_{12}B_{12}^\top), \lambda_{\max}(B_{22}B_{22}^\top)\right\}}$. This follows from positive definiteness of P_{NE} and $\widehat{M}$, and the bound can be deduced by noticing that

$$\lambda_{\min}\left(P_{NE}^{-1} \mathscr{P}_r \widehat{M} \mathscr{P}_r^\top \right) \geq \frac{\lambda_{\min}(\widehat{M})}{\lambda_{\max}(P_{NE})} \geq \frac{\delta}{\delta + \max\left\{\lambda_{\max}(B_{11}B_{11}^\top + B_{12}B_{12}^\top), \lambda_{\max}(B_{22}B_{22}^\top)\right\}}.$$

Still we need to find an upper bound for the largest eigenvalue. To that end, notice that:

$$\gamma = \frac{v_2^\top \left(\widehat{M}_{22} - \widehat{M}_{21} \widehat{M}_{11}^{-1} \widehat{M}_{21}^\top \right) v_2}{v_2^\top \widetilde{M}_{22} v_2},$$

which follows from the definition of $\widetilde{M}_{22}$. Positive definiteness of $\widehat{M}$ then implies that $\gamma > 0$. From the last relation we also have that:

$$0 < \gamma \leq 1 + \frac{v_2^\top B_{21} B_{21}^\top v_2}{v_2^\top \widetilde{M}_{22} v_2} = \frac{v_2^\top \widehat{M}_{22} v_2}{v_2^\top \widetilde{M}_{22} v_2} \leq 1 + \frac{\lambda_{\max}(B_{21}B_{21}^\top)}{\delta + \lambda_{\min}(B_{22}B_{22}^\top)} =: \gamma_u.$$

Furthermore, $\beta_l := -\left(2 + \frac{\lambda_{\max}(B_{21}B_{21}^\top)}{\delta + \lambda_{\min}(B_{22}B_{22}^\top)}\right) \leq \beta \leq -2$. From the previous inequality, one can also observe that $\gamma < -\beta - 1$.

Returning to (7), we first consider the following solution:

$$\lambda_- = \frac{1}{2}\left(-\beta - \sqrt{\beta^2 - 4\gamma} \right).$$

It is easy to see that $\beta^2 - 4\gamma$ is always larger than 0. Next, we notice that the relation for λ_- is increasing with respect to γ. We omit finding a lower bound for λ_- since this was established earlier. For the upper bound, we use the fact that $\gamma < -\beta - 1$, to obtain:

$$\lambda_- < \frac{1}{2}\left(-\beta - \sqrt{\beta^2 + 4(\beta + 1)} \right) = \frac{1}{2}(|\beta| - |\beta + 2|) = 1,$$

since $\beta \leq -2$ (also, in the beginning of this case, we have treated $\lambda_- = 1$ separately).

Finally, we consider the other solution of (7), which reads:

$$\lambda_+ = \frac{1}{2}\Big(-\beta + \sqrt{\beta^2 - 4\gamma}\Big).$$

Firstly, we can easily notice that $\lambda_+ > 1$. Subsequently, upon noticing that λ_+ is decreasing with respect to γ, we can obtain the following obvious bound:

$$\lambda_+ \leq |\beta| \leq -\beta_l.$$

Now, let us observe that dropping $\widehat{M}_{21}$ and $\widehat{M}_{21}^\top$ yields at most $k_r + \text{rank}(\widehat{M}_{21}^\top) \leq 2k_r$ eigenvalue outliers. Similarly, dropping $B_{21}B_{21}^\top$ from the (2,2) block of M yields at most $\text{rank}\big(B_{21}\big) \leq k_c$ eigenvalue outliers. Hence, there will be at least $\max\big\{m - (2k_r + k_c), 0\big\}$ eigenvalues of the preconditioned matrix at 1.

Finally, the case where $k_c = 0$ and $k_r > 0$ follows by a direct generalization of [13, Theorem 4.1], and completes the proof. □

Remark 1 Now that we have presented the spectral properties of the preconditioned system, let us discuss the use of such a preconditioning strategy. In practice, we solve system (1) using the normal equations only when the matrix Q is diagonal. As already mentioned, in this case $\widehat{M} = M$, and thus P_{NE} is a preconditioner for the normal equations' matrix. In Sect. 3, we discuss how P_{NE} is utilized to construct preconditioners for the saddle point matrix in (1), even if Q is not diagonal (in which case $\widehat{M}$ is an approximation of the normal equations' matrix).

Let us now discuss how to choose which columns of A (or rows of $\widehat{M}$, respectively) to drop (sparsify, respectively).

- Firstly, it often happens in optimization, and especially when solving systems arising from the application of an interior point method (as already mentioned in the introduction), to have certain diagonal elements of G that are very small. In view of this property, and given the bound presented in Theorem 1, we can observe that dropping all columns corresponding to small diagonal elements in G results in manageable and not too sizeable outliers. Such a preconditioner was proposed in [7] for the case where $\widehat{Q} = \text{Diag}(Q)$, and arises as a special case of P_{NE} in (4), by choosing $k_r = 0$ and a suitable permutation matrix $\mathscr{P}_c$, traversing first the k_c indices corresponding to the smallest diagonal elements of G.
- Secondly, it is common in many application areas to have a small number of columns or rows of A that are dense. Such columns (or rows) could pose significant difficulties as they produce dense factors when one tries to factorize the normal equations (e.g. using a Cholesky decomposition). This is especially the case for dense columns. A single dense column of A with p non-zero entries induces a dense window of size $p \times p$ in the normal equations (we refer the reader to the discussion in [2, Sect. 4]). The use of a preconditioner like the one defined in (4) serves the purpose of dropping (sparsifying, respectively) such columns of A (rows of $\widehat{M}$, respectively), thus making the Cholesky factors of P_{NE} signifi-

cantly more sparse. For example, we may find two permutation matrices $\mathscr{P}_r$, $\mathscr{P}_c$ which sort the rows and columns, respectively, of A in descending order of their number of non-zeros, and write $\widehat{A} = \mathscr{P}_r A \mathscr{P}_c$. Then, the resulting normal equations read as $\mathscr{P}_r^\top BB^\top \mathscr{P}_r + \delta I_m$. As long as the number of dropped columns or rows is low (which is observed in several applications), the number of outliers produced by this dropping strategy is manageable. While some of these outliers will be dangerously close to zero (given that the regularization parameter $\delta > 0$ is small), they can be dealt with efficiently. We should note, however, that if $\widehat{Q}$ is non-diagonal, without a sparse representation of its inverse, this strategy would be unattractive to employ, and thus in this work we consider it only when $\widehat{Q}$ is diagonal. Indeed, as we discuss in Sect. 2.2, in this case we never explicitly form the normal equations. Instead, we appropriately utilize an $LDL^\top$ decomposition, and the fill-in produced by few dense rows or columns of A can be prevented.

Remark 2 As we discuss later, a case of interest would be to only drop certain k_c columns of A, which results in introducing at most k_c eigenvalue outliers in the preconditioned matrix. Similarly, only sparsifying certain k_r rows of $\widehat{M}$, introduces at most $2k_r$ non-unit eigenvalues. Notice that dropping columns is expected to be more useful in general, resulting in fewer outliers and possibly in greater gains (either in terms of processing time or memory requirements). Indeed, notice that, on the one hand, dropping dense columns of A can result in a significant reduction of the fill-in within a factorization of the normal equations, while, on the other hand, dropping any column of A corresponding to a small diagonal element of G yields a not too sizeable outlier. However, in certain special applications one has to resort to sparsifying "problematic" rows. Indeed, we refer the reader to [13, Sect. 4], where such a row-sparsifying strategy was key to the efficient solution of fMRI classification problems.

2.2 An $LDL^\top$-based preconditioner

Next, we present an alternative to the preconditioner in (4). This approach offers a possibility for dealing with a small set of dense columns or rows of the matrix A, while remaining efficient when the approximate Hessian $\widehat{Q}$, given in (2), is non-diagonal. More specifically, let us divide the columns of the matrix A into two mutually exclusive sets $\mathcal{B}$ and $\mathcal{N}$, based solely on the magnitude of the respective diagonal elements of the matrix G, and not on the density of the columns of A. Then, using the column-dropping strategy presented in the previous section (assuming that the variables corresponding to $\mathcal{N}$ are such that $Q^{(j,j)} \geq Q^{(i,i)}$, for all $j \in \mathcal{N}$ and all $i \in \mathcal{B}$), we propose approximating the matrix $\widehat{M}$ by the following preconditioner:

$$P_{NE,(|\mathcal{N}|,0)} = A^{(:,\mathcal{B})} \widehat{G}^{(\mathcal{B},\mathcal{B})} \left(A^{(:,\mathcal{B})}\right)^\top + \delta I_m, \tag{8}$$

which was proposed in [7], for the special case where $\widehat{G}$ was diagonal. Notice that the block-separable structure of $\widehat{Q}$, given in (2), implies that $(\widehat{G}^{(\mathcal{B},\mathcal{B})})^{-1} = \widehat{Q}^{(\mathcal{B},\mathcal{B})} + \rho I_{|\mathcal{B}|}$. Given our previous discussion, we would like to avoid applying the inverse of this preconditioner by means of a Cholesky decomposition,

as a single dense column of A in $\mathcal{B}$ could result in dense Cholesky factors, while the potential non-diagonal nature of $\widehat{Q}^{(\mathcal{B},\mathcal{B})}$ might prevent us from even efficiently forming it. Instead, we form an appropriate saddle point system to compute the action of the approximated normal equations. More specifically, given an arbitrary vector $y \in \mathbb{R}^m$, instead of computing $P^{-1}_{NE,(|\mathcal{N}|,0)}y$ using a Cholesky decomposition, we can compute

$$\underbrace{\begin{bmatrix} -\left(Q^{(\mathcal{B},\mathcal{B})} + \rho I_{|\mathcal{B}|}\right) & \left(A^{(:,\mathcal{B})}\right)^\top \\ A^{(:,\mathcal{B})} & \delta I_m \end{bmatrix}}_{\widetilde{P}_{NE}} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 0_{|\mathcal{B}|} \\ y \end{bmatrix}, \tag{9}$$

by means of an $LDL^\top$ decomposition of the previous saddle point matrix. Then, we notice that returning w_2 is equivalent to computing $P^{-1}_{NE,(|\mathcal{N}|,0)}y$.

Following the discussion in [2, Sect. 4], we know that using an $LDL^\top$ structure to factorize the matrix in (9) can result in significant memory savings compared to the Cholesky decomposition of $P_{NE,(|\mathcal{N}|,0)}$. Notice that in view of the regularized nature of the systems under consideration (indeed, we have assumed that G is positive definite), we can use the result in [46], stating that matrices like the one in (9) are *quasi-definite*; any symmetric permutation of such matrices admits an $LDL^\top$ decomposition.

While this approach might seem expensive, it can provide significant time and memory savings, especially in cases where $A^{(:,\mathcal{B})}$ contains dense columns. In the previous section we discussed a strategy for alleviating this issue, noting however that such a strategy can only be used to deal with a small number of dense columns. On the contrary, if we have a sizeable subset of the columns of $A^{(:,\mathcal{B})}$ that are dense, we could delay their pivot order within the $LDL^\top$, thus significantly reducing the overall fill-in of the decomposition factors, without introducing any eigenvalue outliers in the preconditioned system. Of course, finding the optimal permutation for the $LDL^\top$ decomposition is an NP-hard problem, however, several permutation heuristics have been developed which are tailored to such symmetric decompositions. Moreover, in the $LDL^\top$ factorization, the pivots are computed dynamically to ensure both stability and sparsity. In view of the previous, the preconditioner based on solving (9) is expected to be more stable than its counterpart based on the Cholesky decomposition. Finally, difficulties arising from dense rows or in general "problematic" rows can also be alleviated using a heuristic proposed in [28].

On the other hand, by using this approach we avoid explicitly forming the preconditioner $P_{NE,(|\mathcal{N}|,0)}$. This is especially important in cases where $\widehat{Q}$ is non-diagonal, and thus forming P_{NE} can be extremely expensive. Hence, the approach presented in this subsection allows us to utilize non-diagonal information in a practical way when constructing an approximation for the matrix Q.

3 Regularized saddle point matrices

Let us now consider the regularized saddle point system in (1). In what follows, we discuss two families of preconditioning strategies, noting their advantages and disadvantages. All presented preconditioners are positive definite in order to be usable within the MINRES method, which is a short-recurrence iterative solver, suitable for

solving symmetric indefinite or quasi-definite systems. This allows us to avoid non-symmetric long-recurrence solvers like the GMRES method.

3.1 Block-diagonal preconditioners

The most common approach is to employ a block-diagonal preconditioner (see [6, 7, 32, 45]). To construct such a preconditioner we need approximations for the (1, 1) block $F := Q + \rho I_n$ of the coefficient matrix in (1), and for its associated Schur complement $M = A\big(Q + \rho I_n\big)^{-1}A^\top + \delta I_m$.

In this section we assume that Q is approximated as shown in (2), and thus can potentially contain non-diagonal blocks. Concerning the approximation of the Schur complement matrix M, we can employ the preconditioner $P_{NE,(k_c,k_r)}$ given in (4). Then, we can define the following preconditioner for the coefficient matrix K in (1):

$$P_{AS,(k_c,k_r)} := \begin{bmatrix} \widehat{Q} + \rho I_n & 0_{n,m} \\ 0_{m,n} & P_{NE,(k_c,k_r)} \end{bmatrix} \equiv \begin{bmatrix} \widehat{F} & 0_{n,m} \\ 0_{m,n} & P_{NE,(k_c,k_r)} \end{bmatrix}. \tag{10}$$

For the rest of this subsection, let $P_{AS,(k_c,k_r)} \equiv P_{AS}$ and $P_{NE,(k_c,k_r)} \equiv P_{NE}$.

In order to analyze the spectrum of the preconditioned matrix $P_{AS}^{-1}K$, we introduce some notation for simplicity of exposition. We work with positive definite similarity transformations of the associated matrices, defined as

$$\widetilde{F} = \widehat{F}^{-1/2} F \widehat{F}^{-1/2}, \quad \widetilde{M}_{NE} = P_{NE}^{-1/2} \widehat{M} P_{NE}^{-1/2}, \tag{11}$$

where $\widehat{M} := A\widehat{F}^{-1}A^\top + \delta I_m$. Then, we set

$$\begin{aligned} \alpha_{NE} &= \lambda_{\min}\Big(\widetilde{M}_{NE}\Big), \quad \beta_{NE} = \lambda_{\max}\Big(\widetilde{M}_{NE}\Big), \quad \kappa_{NE} = \frac{\beta_{NE}}{\alpha_{NE}}, \\ \alpha_F &= \lambda_{\min}\Big(\widetilde{F}\Big), \quad \beta_F = \lambda_{\max}\Big(\widetilde{F}\Big), \quad \kappa_F = \frac{\beta_F}{\alpha_F}. \end{aligned}$$

From the definition of $\widehat{Q}$ given in (2), we can observe that $\alpha_F \le 1 \le \beta_F$ as

$$\frac{1}{n}\sum_{i=1}^{n} \lambda_i\Big(\widehat{F}^{-1}F\Big) = \frac{1}{n}\mathrm{Tr}\Big(\widehat{F}^{-1}F\Big) = 1.$$

On the other hand, notice that α_{NE} and β_{NE} can be bounded directly using Theorem 1. Indeed, from (11), we observe that we need to bound the spectrum of an approximate preconditioned Schur complement matrix, since M has been substituted by $\widehat{M}$. This is exactly what the analysis in Sect. 2.1 does. Below we provide a theorem analyzing the spectral properties of the matrix $P_{AS}^{-1}K$.

Theorem 2 *The eigenvalues of $P_{AS}^{-1}K$ lie in the union of the following intervals*:

$$I_- = \left[-\beta_F - \sqrt{\beta_{NE}}, -\alpha_F\right]; \quad I_+ = \left[\frac{1}{2}\left(-\beta_F + \sqrt{\beta_F^2 + 4\alpha_{NE}}\right), 1 + \sqrt{\beta_{NE} - 1}\right].$$

Proof The proof, which follows by trivially extending [7, Theorem 3], is summarized in the Appendix for completeness. □

The authors of [7] make use of the approximation $\widehat{Q} = \text{Diag}(Q)$, and $P_{NE} = P_{NE,(|\mathcal{N}|,0)}$ (where the latter is defined in (8), with $\widehat{G} = (\widehat{Q} + \rho I_n)^{-1}$). Approximating Q by its diagonal allows one to form the normal equations' preconditioner (i.e. $P_{NE,(|\mathcal{N}|,0)}$), thus enabling the efficient use of a Cholesky factorization. However, there might exist problems for which a better approximation of the matrix Q is required. In this case, one could consider a sparsified version of Q like the one given in (2). While this could lead to reasonable approximations, the problem of fill-in introduced by $(\widehat{Q} + \rho I_n)^{-1}$ in the Schur complement approximation, $P_{NE,(|\mathcal{N}|,0)}$, would (in general) remain.

In order to address the previous issue, we make use of the $LDL^\top$-based preconditioner defined in Sect. 2.2. As in Sect. 2.2, we divide the columns of A using the sets $\mathcal{B}$ and $\mathcal{N}$, where $\mathcal{N}$ contains all indices corresponding to the largest diagonal elements of Q. Assume that Q is approximated by $\widehat{Q}$ as given in (2), where the permutation matrix $\mathscr{P}_c$ traverses first the column indices in $\mathcal{N}$. Then, for example, a reasonable approximation would be the following

$$\mathscr{P}_c^\top \widehat{Q} \mathscr{P}_c = \begin{bmatrix} \text{Diag}\left(Q^{(\mathcal{N},\mathcal{N})}\right) & 0_{|\mathcal{N}|,|\mathcal{B}|} \\ 0_{|\mathcal{B}|,|\mathcal{N}|} & Q^{(\mathcal{B},\mathcal{B})} \end{bmatrix}. \tag{12}$$

The effect of this approximation in the context of regularized IPMs has been analyzed in detail in [39]. We notice that $(Q^{(\mathcal{B},\mathcal{B})} + \rho I_{|\mathcal{B}|})^{-1}$ does not introduce significant fill-in in the (2, 2) block of the preconditioner in (10), as we implicitly invert this block using the methodology presented in Sect. 2.2.

Remark 3 Notice that further approximations can be employed here. In particular, we could define a banded approximation of Q and then employ the approximation proposed earlier. The implicit inversion of the Schur complement, outlined in Sect. 2.2, gives us complete freedom on how to approximate Q, and hence we no longer rely on diagonal approximations. We return to this point in the numerical experiments.

3.2 Factorization-based preconditioners

Finally, given the regularized nature of the systems under consideration, we can construct factorization-based preconditioners for MINRES. In particular, we can compute $K = LDL^\top$ (with K in (1)), where D is a diagonal matrix (since K is

quasi-definite [46]) with n negative and m positive elements on its diagonal. Then, by defining $P_K := L|D|^{\frac{1}{2}}$, the preconditioned saddle point matrix reads:

$$P_K^{-1} K P_K^{-\top} = |D|^{-1} D,$$

and hence contains only two distinct eigenvalues: -1 and 1 [20, 34]. As before, let us assume that we have available a splitting of the columns of A such that $A\mathscr{P}_c = [A^{\mathcal{B}} \; A^{\mathcal{N}}]$, where $\mathcal{B}$ contains indices corresponding to the smallest diagonal elements of Q. Then, we can precondition K, left and right, by $\widehat{P}_K := \widehat{L}|\widehat{D}|^{\frac{1}{2}}$, where $\widehat{K} = \widehat{L}\widehat{D}\widehat{L}^{\top}$ and:

$$\widehat{K} := \begin{bmatrix} -\widehat{Q} & \widehat{A}^{\top} \\ \widehat{A} & \delta I_m, \end{bmatrix},$$

with $\widehat{A} := [A^{\mathcal{B}} \; 0_{m,|\mathcal{N}|}]\mathscr{P}_c^{\top}$, and $\widehat{Q}$ defined as in (12). Notice that by setting several columns of A to zero, as well as by sparsifying the respective rows and columns of Q, the cost of applying the inverse of $\widehat{K}$ is significantly reduced when compared to that required to apply the inverse of K.

Further limited-memory versions of this preconditioner can be employed, e.g. by using the methodologies presented in [34, 44]. Other approximations of the blocks of $\widehat{K}$, based on the structure of the problem at hand, could also be possible, as already mentioned in the previous subsection.

We should note, however, that this approach is less stable than the approach presented in Sect. 3.1. This is because we are required to use only diagonal pivots during the $LDL^{\top}$ decomposition for this methodology to work (indeed, notice that the presence of a non-diagonal matrix D in the factorization of K would not allow the use of such a preconditioning strategy). If the regularization parameters δ or ρ have very small values, the stability of the factorization could be compromised (since we enforce the use of only diagonal 1×1 pivots), and we would have to heavily rely on stability introduced by means of uniform [43] or weighted regularization [1]. On the other hand, the methodology presented in Sect. 3.1 would not be affected by the occasional use of 2×2 pivots within the $LDL^{\top}$ factorization for the implicit inversion of the approximate Schur complement. Of course the latter is not the case if the "Analyze" phase of the factorization (used to determine the pivot order) is performed separately, however, the subset of columns in $\mathcal{B}$ may change significantly from one iteration to the next, making this strategy less attractive. Nevertheless, this factorization-based approach can be more efficient than the approach presented in Sect. 3.1, when solving certain non-separable convex programming problems. This is because the approach in Sect. 3.1 requires the computation of an $LDL^{\top}$ decomposition of the coefficient matrix $\widetilde{P}_{NE}$ in (9) (with potential 2×2 pivots) as well as a Cholesky decomposition of $\widehat{Q} + \rho I_n$ (or some iterative scheme which could be application-dependent, as in [38]).

4 Regularized IPMs: numerical results

Let us now focus on the case of the regularized saddle point systems (and their respective normal equations) arising from the application of regularized IPMs on convex programming problems. The MATLAB code, which is based on the IP-PMM presented in [40, 41], can be found on GitHub.[1]

In all the presented experiments a 6-digit accurate solution is requested. The reader is referred to [7, Sect. 4] and [40, Sect. 5] for the implementation details of the algorithm (such as termination criteria, the employed predictor–corrector scheme for the solution of the Newton system, as well as the tuning of the algorithmic regularization parameters). The associated iterative methods (i.e. the Preconditioned Conjugate Gradient method (PCG) or MINRES) are adaptively terminated if the following accuracy is reached: $\frac{\min\{10^{-3},\max\{10^{-1}\cdot\mu_k,\texttt{tol}\}\}}{\max\{1,\|\texttt{rhs}\|\}}$, where $\texttt{tol} = 10^{-6}$, μ_k is the barrier parameter at iteration k, and `rhs` is the right hand side of the system being solved. This adaptive stopping criterion is based on the developments in [11]. When PCG is employed we allow at most 100 iterations per linear system solved, while for MINRES up to 200 iterations are allowed. If the maximum number of Krylov iterations is reached, an inexact Newton direction is accepted if it is at least 3-digit accurate. Any Cholesky decomposition is computed via the `chol` function of MATLAB. When an $LDL^\top$ decomposition is employed, we utilize the `ldl` function of MATLAB. In this case, the minimum pivot threshold is adaptively set to $\text{pivot}_{\text{thr}} = 0.1 \cdot \min\{\delta, \rho, 10^{-4}\}$. This is done to ensure that no 2×2 pivots are used during the factorization, ensuring that the factorization remains efficient. However, in the context of the preconditioner in Sect. 2.2, where 2×2 pivots can safely be used (unlike the preconditioner presented in Sect. 3.2, which requires the use of 1×1 pivots), this mechanism is turned off when $\min\{\delta, \rho\} \leq 10^{-8}$, and we set $\text{pivot}_{\text{thr}} = 10^{-6}$ to ensure stability. All the presented experiments were run on a PC with a 2.2GHz Intel Core i7-8750 H processor (hexa-core), 16GB RAM, run under the Windows 10 operating system. The MATLAB version used was 2019a.

4.1 Linear programming

Let us initially focus on Linear Programming (LP) problems of the following form:

$$\min_{x\in\mathbb{R}^n} c^\top x, \quad \text{s.t.} \quad Ax = b,\ x^{\mathcal{I}} \geq 0,\ x^{\mathcal{F}} \text{ free}, \tag{LP}$$

where $A \in \mathbb{R}^{m\times n}$, $\mathcal{I} \cap \mathcal{F} = \emptyset$, and $\mathcal{I} \cup \mathcal{F} = \{1, \ldots, n\}$. Applying regularized IPMs to problems like (LP), one often solves a regularized normal equations system at every iteration. Such systems have a coefficient matrix of the following form:

$$M = AGA^\top + \delta I_m, \qquad G^{(i,i)} = \begin{cases} \frac{1}{\rho} & \text{if } i \in \mathcal{F}, \\ \frac{1}{\rho + z^i/x^i} & \text{if } i \in \mathcal{I}, \end{cases}$$

[1] https://github.com/spougkakiotis/IP-PMM_QP_Solver.

where δ, $\rho > 0$ and $z \in \mathbb{R}^n$ (where $z^{\mathcal{I}} \geq 0$, $z^{\mathcal{F}} = 0$) are the dual slack variables. Notice that the IPM penalty parameter μ is often tuned as $\mu = \frac{(x^{\mathcal{I}})^\top z^{\mathcal{I}}}{n}$ and we expect that $\mu \to 0$. As already mentioned in the introduction, the variables are naturally split into "basic"–$\mathcal{B}$, "non-basic"–$\mathcal{N}$, and "undecided"–$\mathcal{U}$. Hence, as IPMs progress towards optimality, we expect the following partition of the quotient $\frac{x^{\mathcal{I}}}{z^{\mathcal{I}}}$:

$$\begin{aligned}
\forall j \in \mathcal{N} : x^j \to 0, \quad z^j \to \widehat{z}^j > 0 \quad &\Rightarrow \quad \frac{x^j}{z^j} = \frac{x^j z^j}{(z^j)^2} = \boldsymbol{\Theta}(\mu), \\
\forall j \in \mathcal{B} : x^j \to \widehat{x}^j > 0, \quad z^j \to 0 \quad &\Rightarrow \quad \frac{x^j}{z^j} = \frac{(x^j)^2}{x^j z^j} = \boldsymbol{\Theta}(\mu^{-1}), \\
\forall j \in \mathcal{U} : x^j = \boldsymbol{\Theta}(1), \quad z^j = \boldsymbol{\Theta}(1) \quad &\Rightarrow \quad \frac{x^j}{z^j} = \boldsymbol{\Theta}(1),
\end{aligned}$$

where $\mathcal{N}$, $\mathcal{B}$, and $\mathcal{U}$ are mutually disjoint, and $\mathcal{N} \cup \mathcal{B} \cup \mathcal{U} = \mathcal{I}$. For the rest of this section, we assume that $\delta = \boldsymbol{\Theta}(\rho) = \boldsymbol{\Theta}(\mu)$. This assumption is based on the developments in [40, 41], where a polynomially convergent regularized IPM is derived for convex quadratic and linear positive semi-definite programming problems, respectively. Following [7], we could precondition the matrix M using the following matrix:

$$P_{NE,(|\mathcal{N}|,0)} = A^{(:,\mathcal{R})} G^{(\mathcal{R},\mathcal{R})} \left(A^{(:,\mathcal{R})}\right)^\top + \delta I_m, \tag{13}$$

where $\mathcal{R} := \mathcal{F} \cup \mathcal{B} \cup \mathcal{U}$. Then, by [7, Theorem 1] (or by applying Theorem 1), we obtain:

$$\lambda_{\max}\left(P^{-1}_{NE,(|\mathcal{N}|,0)} M\right) \leq 1 + \frac{\max\limits_{j \in \mathcal{N}} \left(G^{(j,j)}\right)}{\delta} \sigma^2_{\max}(A), \qquad \lambda_{\min}\left(P^{-1}_{NE,(|\mathcal{N}|,0)} M\right) \geq 1.$$

The preconditioner in (13) is a special case of the preconditioner defined in Sect. 2. Indeed, as already indicated by its notation, it can be derived by setting $k_r = 0$ and then by dropping all columns belonging to $\mathcal{N}$, i.e. we set $k_c = |\mathcal{N}|$ and we drop the k_c columns of A corresponding to the smallest diagonal elements of G. Notice that in the linear programming case, the matrix $\widehat{M}$ that is analyzed in Sect. 2 coincides with M, since G is diagonal.

From our previous remarks, we notice that

$$\max_{j \in \mathcal{N}} \left(G^{(j,j)}\right) = \boldsymbol{\Theta}(\mu) = \boldsymbol{\Theta}(\delta)$$

implies that the spectrum of the preconditioned matrix remains bounded and is asymptotically independent of μ (assuming that $\delta = \boldsymbol{\Theta}(\mu)$). While this preconditioner performs very well in practice (see [7, Sect. 4]), it can be expensive to compute in certain cases, as its inverse needs to be applied by means of a Cholesky decomposition. To that end, we propose to further approximate this matrix as indicated in Sect. 2. This idea is based on the fact that the PCG method is expected to converge in a small number of iterations, if the preconditioned system matrix can be written as:

$$P^{-1}M = I_m + U + V,$$

where P is the preconditioner, M is the normal equations matrix, U is a low-rank matrix, and V is a matrix with small norm. In our case, dropping the part of the normal equations corresponding to $\mathcal{N}$ contributes the small-norm term (that is, $V = A^{(:,\mathcal{N})}G^{(\mathcal{N},\mathcal{N})}(A^{(:,\mathcal{N})})^\top$, the norm of which is of the order of magnitude of μ), and furthermore dropping a few dense columns (or sparsifying certain rows) contributes the low-rank term (indeed, as already shown in Theorem 1, dropping k_c dense columns of A and sparsifying k_r dense rows of M yields at most $2k_r + k_c$ outliers, and thus $\text{rank}(U) \leq 2k_r + k_c$, where, in this case, k_c does not account for columns corresponding to the index set $\mathcal{N}$).

To construct such a preconditioner, we first need to note that $\mathcal{R}$ will change at every IPM iteration. However, we can heuristically choose which columns to drop (and/or rows to sparsify) based on the sparsity pattern of A. To that end, at the beginning of the optimization procedure, we count the number of non-zeros of each column and row of A, respectively. These can then be used to sort the columns and rows of A in descending order of their number of non-zero entries. These sorted columns and rows can easily be represented by means of two permutation matrices $\mathscr{P}_c$ and $\mathscr{P}_r$. We note that this is a heuristic, and it is not guaranteed to identify the most "problematic" columns or rows (which can be sources of difficulty for IPMs). For a discussion on such heuristics, and alternatives, the reader is referred to [2, Sect. 4], and the references therein.

4.1.1 Numerical results

Initially, we present some results to show the effect of dropping dense columns of A and then of sparsifying dense rows of M using the strategy outlined in Sect. 2. Then, we present a comparison between the preconditioner in (4) (that is, $P_{NE,(k_c,k_r)}$), the preconditioner given in (8) or (13) (denoted as $P_{NE,(|\mathcal{N}|,0)}$), noting that (8) and (13) are equivalent, and the one in (9) (that is, $\widetilde{P}_{NE}$). Notice that in the linear programming case employing $\widetilde{P}_{NE}$ and $P_{NE,(|\mathcal{N}|,0)}$ should yield identical results in exact arithmetic. The difference between these preconditioning strategies is that in the latter case (i.e. $\widetilde{P}_{NE}$) the action of the former preconditioner (i.e. $P_{NE,(|\mathcal{N}|,0)}$) is computed implicitly by means of an $LDL^\top$ factorization of $\widetilde{P}_{NE}$, as indicated in Sect. 2.2.

Dropping dense columns versus factorizing directly. We run IP-PMM on all problems from the Netlib collection that have some dense columns, where dense is defined in this case to be a column with at least 15% non-zero elements. We note that these were the only problems within the Netlib collection having any columns with such a density of non-zero elements. We compare an IP-PMM using Cholesky factorization for the solution of the associated Newton system (Chol.), with an IP-PMM that uses the preconditioner $P_{NE,(k_c,0)}$ presented in Sect. 2 alongside PCG. The latter method is only allowed to drop dense columns (at most 30) to create the preconditioner (and thus $k_c \leq 30$). The results are collected in Table 1, where k_c denotes the number of dense columns that are dropped to create the preconditioner, nnz denotes the number of non-zero elements present in the Cholesky factor, Avg.

Table 1 The effect of dropping dense (> 15%) columns of A (Netlib collection)

Name	k_c	nnz		Time (s)		Avg. Krylov It.	Krylov Last
		Chol.	$P_{NE,(k_c,0)}$	Chol.	$P_{NE,(k_c,0)}$		
BLEND	5	1006	**736**	**0.05**	**0.05**	14.50	14
FIT1P	20	197,676	**26,706**	1.16	**0.32**	18.67	35
FIT2P	16	4,516,500	**1,962,616**	27.17	**16.21**	17.33	33
FORPLAN	8	3810	**2,918**	**0.11**	**0.11**	2.81	3
ISRAEL	30	12,261	**1,744**	**0.14**	0.30	53.50	100
SEBA	14	55,937	**2,238**	0.35	**0.18**	13.83	15

Bold numbers denote the best performance for a given metric

Krylov It. denotes the average number of Krylov iterations performed from the inexact approach, while Krylov Last denotes the number of Krylov iterates performed in the last IPM iteration of the inexact approach. In what follows, when comparing different strategies, we report the best performing metrics achieved in bold.

From Table 1 we can immediately see that certain dense columns present in the constraint matrix A can have a significant impact on the sparsity pattern of the Cholesky factors. This is a well-known fact (see for example the discussion in [2, Sect. 4]). Notice that the Netlib collection contains only small- to medium-scale instances. For such problems, memory is not an issue, and hence direct methods tend to be faster than their iterative alternatives (like PCG). Despite the small size of the presented problems, we can see tremendous memory savings (and even a decrease in CPU time) for problems FIT1P, FIT2P, and SEBA, by eliminating only a small number of dense columns. On the other hand, for problems where we observe an increase in CPU time (e.g. see ISRAEL), the memory savings can be significant, making this acceptable.

Sparsifying dense rows versus factorizing directly. Next, we consider the case where the inexact version of IP-PMM is only allowed to sparsify dense rows, where dense is defined in this case to be a row with at least 25% non-zero elements.

Before moving to the numerical results, let us note some differences between sparsifying rows of M and dropping columns of A. Firstly, as we have shown in Sect. 2, sparsifying k rows can potentially introduce twice as many outliers, while dropping k columns introduces at most k eigenvalue outliers. Furthermore, the potential density induced in the Cholesky factors by a single dense column is usually more significant than that introduced by a single dense row. However, we cannot know in advance how effective the dropping of a column will be. On the other hand, sparsifying dense rows of M introduces a certain separability in the approximate normal equations matrix, allowing us to estimate very well the memory savings.

In Table 2 we compare the direct IP-PMM, to its inexact version, the preconditioner of which is only allowed to sparsify at most 30 dense rows of M. Any problem with at least one dense row from the Netlib collection is considered.

From Table 2 we can observe that the required memory to form the Cholesky factors is consistently decreased but CPU time is often increased by the row-dropping

Table 2 The effect of dropping dense (> 25%) rows of M (Netlib collection)

Name	k_r	nnz		Time (s)		Avg. Krylov It.	Krylov Last
		Chol.	$P_{NE,(0,k_r)}$	Chol.	$P_{NE,(0,k_r)}$		
BEACONFD	17	2903	**1,475**	**0.07**	0.09	13.59	23
BLEND	1	1006	**959**	0.05	**0.04**	2.88	3
D6CUBE	6	55,179	**52,757**	**0.27**	0.44	12.73	18
FIT1D	11	14,726	**4,973**	**0.22**	0.38	23.50	46
FIT2D	12	139,843	**49,513**	**1.51**	2.47	21.30	44
ISRAEL	3	12,261	**11,758**	**0.14**	**0.14**	6.96	8
STANDATA	1	3416	**3,168**	**0.07**	**0.07**	2.94	3
STANDGUB	1	3418	**3,170**	**0.08**	**0.08**	2.98	3
STANDMPS	1	5529	**5,185**	**0.09**	**0.09**	2.91	3
WOOD1P	27	19,088	**13,879**	**0.34**	0.44	14.95	13

Bold numbers denote the best performance for a given metric

strategy. We should note that an increase in CPU time usually relates to the size of the problems under consideration, and CPU time as well as memory advantages can be observed if the problem is sufficiently large with sufficiently many dense rows. In particular, this row sparsifying strategy was successfully used within IP-PMM in [13, Sect. 4] in order to tackle fMRI sparse approximation problems (in which the constraint matrix contains thousands of dense rows). Memory requirements were significantly lowered, allowing this inexact version to outperform its exact counterpart, while being competitive with standard state-of-the-art first-order methods used to solve such problems.

The Cholesky versus the $LDL^\top$ approach. Let us now provide some numerical evidence for the potential benefits and drawbacks of the approach presented in Sect. 2.2 over that presented in Sect. 2.1. To that end, we run three inexact versions of IP-PMM, on some of the most challenging instances within the Netlib collection. The first approach uses the preconditioner given in (4) (denoted as $P_{NE,(k_c,k_r)}$, allowing at most 15 dense columns/rows to be dropped/sparsified), the second uses the preconditioner given in (8) (denoted as $P_{NE,(|\mathcal{N}|,0)}$; notice that this is the same as the former preconditioner, without employing the strategy of dropping/sparsifying dense columns/rows), while the third version uses the preconditioner in (9), denoted as $\widetilde{P}_{NE}$. In all three cases the set $\mathcal{B}$, used to decide which columns are dropped irrespectively of their density, is determined as indicated at the beginning of this section.

From Table 3 we can observe that the $LDL^\top$-based preconditioner can provide substantial (memory and/or CPU time) benefits for certain problems (e.g. see problems FIT1P, FIT2P, QAP12, QAP15). Nevertheless, we should note that this approach is usually slower, albeit more stable (as a pivot re-ordering is computed at every iteration, and the pivots of the $LDL^\top$ factorization are chosen to ensure stability as well as efficiency). We observe that instances AGG, DFL001, PILOT did not benefit from the use of this strategy, neither in terms of efficiency nor memory requirements, despite a comparable number of Krylov iterations. This comes in line with our observations in Sect. 2.2, since none of the aforementioned instances contains any dense rows or columns. Notice that the stability and efficiency of the

Table 3 Cholesky-based versus the $LDL^\top$-based preconditioner (Netlib collection)

Name	max nnz			Time (s)			Krylov Its.		
	$P_{NE,(k_c,k_r)}$	$P_{NE,(\lfloor\mathcal{N}\rfloor,0)}$	$\widetilde{P}_{NE}$	$P_{NE,(k_c,k_r)}$	$P_{NE,(\lfloor\mathcal{N}\rfloor,0)}$	$\widetilde{P}_{NE}$	$P_{NE,(k_c,k_r)}$	$P_{NE,(\lfloor\mathcal{N}\rfloor,0)}$	$\widetilde{P}_{NE}$
AGG	$1.60 \cdot 10^4$	$1.60 \cdot 10^4$	$\mathbf{1.40 \cdot 10^4}$	**0.42**	**0.42**	0.58	2966	2966	**2,268**
DFL001	$1.55 \cdot 10^6$	$1.55 \cdot 10^6$	$\mathbf{1.54 \cdot 10^6}$	**8.40**	**8.40**	31.51	**4,778**	**4,778**	4969
FIT1P	$1.20 \cdot 10^5$	$2.00 \cdot 10^5$	$\mathbf{1.36 \cdot 10^4}$	0.58	1.56	**0.51**	1166	636	**491**
FIT2P	$4.19 \cdot 10^6$	$4.60 \cdot 10^6$	$\mathbf{9.4 \cdot 10^4}$	29.45	43.10	**15.95**	2555	1880	**1,867**
PILOT	$\mathbf{1.99 \cdot 10^5}$	$\mathbf{1.99 \cdot 10^5}$	$4.02 \cdot 10^5$	**3.79**	**3.79**	9.16	3558	3558	**3,530**
QAP12	$2.48 \cdot 10^6$	$2.48 \cdot 10^6$	$\mathbf{1.61 \cdot 10^6}$	**2.09**	**2.09**	5.24	**1,583**	**1,583**	1591
QAP15	$8.83 \cdot 10^6$	$8.83 \cdot 10^6$	$\mathbf{5.28 \cdot 10^6}$	**20.56**	**20.56**	25.42	**1,704**	**1,704**	1708
SEBA	$\mathbf{2.18 \cdot 10^3}$	$5.54 \cdot 10^4$	$7.5 \cdot 10^3$	**0.33**	0.80	0.84	2678	2396	**2,313**

Bold numbers denote the best performance for a given metric

preconditioner $\widetilde{P}_{NE}$ depends heavily on the choice for the threshold for the `ldl` function of MATLAB. Larger values imply better stability, however at the cost of efficiency, since more 2×2 pivots will be chosen during the $LDL^\top$ factorization. The stability of this approach can be guaranteed by using a large-enough pivot threshold. Additionally, there are instances without dense rows or columns (see QAP12, QAP15), in which the $LDL^\top$-based preconditioner (i.e. $\widetilde{P}_{NE}$) provides significant advantages in terms of memory requirements. Finally, we note that for problems AGG, PILOT, QAP12, and QAP15, the two Cholesky-based variants are exactly the same, as no dense columns or rows were present.

There is a long-standing discussion on the comparison between the Cholesky and the $LDL^\top$ decompositions. The former tend to be faster and usually easier to implement, while the latter tend to be slower, more stable, and more general. For more on this subject, the reader is referred to [2, Sect. 4] and the references therein.

4.2 Convex quadratic programming

Next, we consider problems of the following form:

$$\min_{x \in \mathbb{R}^n} \; c^\top x + \frac{1}{2} x^\top H x, \quad \text{s.t.} \quad Ax = b, \; x^{\mathcal{I}} \geq 0, \; x^{\mathcal{F}} \text{ free}, \tag{QP}$$

where $H \in \mathbb{R}^{n \times n}$ is the positive semi-definite Hessian matrix. Let us notice that a similar partitioning of the variables as that presented in Sect. 4.1 also holds in this case. Hence, the index set $\mathcal{N}$ guides us on which columns of A to drop. In the case where H is either diagonal, or can be well-approximated by a diagonal, the discussion of Sect. 4.1, about dropping dense columns of A (or sparsifying dense rows of the approximate Schur complement $\widehat{M}$ given in (3)), also applies here.

In what follows we make use of three different preconditioners. We compare the two block-diagonal preconditioners given in Sect. 3.1. The first is called $P^C_{AS,(k_c,k_r)}$ (where the superscript C stands for Cholesky, which is used to invert the (2, 2) block

Table 4 Comparison of QP preconditioners (Maros–Mészáros collection)

Name	max nnz			Time (s)			Krylov Its.		
	$P^C_{AS,(k_c,k_r)}$	$P^L_{AS,(\lvert\mathcal{N}\rvert,0)}$	$\widehat{P}_K$	$P^C_{AS,(k_c,k_r)}$	$P^L_{AS,(\lvert\mathcal{N}\rvert,0)}$	$\widehat{P}_K$	$P^C_{AS,(k_c,k_r)}$	$P^L_{AS,(\lvert\mathcal{N}\rvert,0)}$	$\widehat{P}_K$
CVXQP2_L	$\mathbf{5.06 \cdot 10^4}$	$2.35 \cdot 10^6$	$1.47 \cdot 10^6$	**9.01**	29.74	33.52	3241	**2,057**	3078
CVXQP2_M	$\mathbf{5.04 \cdot 10^3}$	$1.15 \cdot 10^5$	$6.15 \cdot 10^4$	**1.02**	1.54	1.44	3019	**2,588**	3128
DUAL3	$\mathbf{1.12 \cdot 10^2}$	$6.77 \cdot 10^3$	$6.77 \cdot 10^3$	0.15	**0.11**	0.14	911	**543**	992
GOULDQP3	$\mathbf{4.89 \cdot 10^3}$	$1.05 \cdot 10^4$	$1.05 \cdot 10^4$	0.31	0.37	**0.27**	1236	1039	**814**
MOSARQP2	$\mathbf{1.71 \cdot 10^4}$	$2.18 \cdot 10^4$	$2.25 \cdot 10^4$	**0.12**	0.16	0.18	752	**730**	803
POWELL20	$\mathbf{2.10 \cdot 10^4}$	$6.20 \cdot 10^4$	$6.20 \cdot 10^4$	**1.64**	2.41	5.24	**1,531**	1537	1538
Q25FV47	$\mathbf{2.13 \cdot 10^4}$	$6.68 \cdot 10^4$	$1.14 \cdot 10^5$	**2.12**	3.75	4.61	6213	**5,994**	8117
QETA-MACRO	$\mathbf{1.13 \cdot 10^4}$	$2.62 \cdot 10^4$	$2.46 \cdot 10^4$	**0.71**	1.40	1.60	4901	**4,661**	6378
QISRAEL	$\mathbf{2.17 \cdot 10^2}$	$4.60 \cdot 10^3$	$4.84 \cdot 10^3$	**0.49**	0.61	0.80	4516	**3,367**	5920
QSHIP12L	$\mathbf{1.09 \cdot 10^4}$	$2.06 \cdot 10^5$	$2.10 \cdot 10^5$	**3.01**	4.92	5.69	5664	**5,070**	5803
STCQP1	$7.00 \cdot 10^5$	$\mathbf{1.25 \cdot 10^5}$	$1.26 \cdot 10^5$	7.48	5.49	**5.21**	3410	**2,543**	2691
STCQP2	$\mathbf{5.44 \cdot 10^4}$	$2.13 \cdot 10^5$	$2.13 \cdot 10^5$	4.30	6.40	**3.96**	3074	2704	**1,918**
UBH1	$\mathbf{8.40 \cdot 10^4}$	$2.13 \cdot 10^5$	$2.13 \cdot 10^5$	**1.97**	3.78	4.17	681	**679**	700

Bold numbers denote the best performance for a given metric

of this preconditioner), and employs a diagonal approximation for Q, allowing one to drop dense columns and/or sparsify dense rows as shown in Sect. 2.1, and the second is called $P^L_{AS,(|\mathcal{N}|,0)}$ (where the superscript L stands for $LDL^\top$), and employs a block-diagonal approximation of Q, using the implicit inversion of the Schur complement proposed in Sect. 2.2. The block-diagonal preconditioners are also compared against the factorization-based preconditioner presented in Sect. 3.2, termed as $\widehat{P}_K$.

4.2.1 Numerical results

In the following experiments we employ MINRES to solve the associated Newton systems. Initially, we present the comparison of the three preconditioning strategies over some problems from the Maros–Mészáros collection of convex quadratic programming problems. Then, the two block-diagonal preconditioning approaches are compared over some Partial Differential Equation (PDE) optimization problems.

Maros–Mészáros collection. In Table 4, we report on the runs of the three methods on a diverse set of non-separable instances within the Maros–Mészáros test set.

From Table 4, one can observe that most of the time $P^C_{AS,(k_c,k_r)}$ is rather inexpensive, and naturally requires some additional Krylov iterations. On the other hand, $P^L_{AS,(|\mathcal{N}|,0)}$ delivers faster convergence of the Krylov solver, at the cost of additional memory (since we utilize non-diagonal Hessian information). However, while the same is true for most problems when employing $\widehat{P}_K$, the latter can be prone to numerical inaccuracy (since we do not allow the use of 2×2 pivots in the $LDL^\top$

factorization). Whether the use of non-diagonal Hessian information is beneficial should depend on the problem under consideration. In the above experiments, this proved to be beneficial for only 4 out of the 13 instances tested (that is DUAL3, GOULDQP3, STCQP1, STCQP2). Nevertheless, we can observe that all three approaches are competitive, while $P^C_{AS,(k_c,k_r)}$ and $P^L_{AS,(|\mathcal{N}|,0)}$ are both very stable.

PDE-constrained optimization instances. Next, we compare the preconditioning approaches on some PDE optimization problems. In particular, we consider the L^1/L^2-regularized Poisson control problem, as well as the L^1/L^2-regularized convection–diffusion control problem with control bounds. We should emphasize at this point that while bespoke preconditioners have been created for PDE problems of this form, here we treat the discretized problems as if we hardly know anything about their structure, to demonstrate the generality of the approaches presented in this paper.

We consider problems of the following form:

$$\begin{aligned} \min_{y,u} \quad & J(y(\boldsymbol{x}), u(\boldsymbol{x})) := \frac{1}{2}\|y - \bar{y}\|^2_{L^2(\Omega)} + \frac{\alpha_1}{2}\|u\|_{L^1(\Omega)} + \frac{\alpha_2}{2}\|u\|^2_{L^2(\Omega)}, \\ \text{s.t.} \quad & Dy(\boldsymbol{x}) = u(\boldsymbol{x}) + g(\boldsymbol{x}), \\ & u_a(\boldsymbol{x}) \le u(\boldsymbol{x}) \le u_b(\boldsymbol{x}), \end{aligned} \tag{14}$$

where $(y, u) \in H^1(\Omega) \times L^2(\Omega)$, D denotes some linear differential operator associated with the differential equation, $\boldsymbol{x}$ is a 2-dimensional spatial variable, and $\alpha_1,\ \alpha_2 \ge 0$ denote the regularization parameters of the control variable. We note that other variants for $J(y, u)$ are possible, including measuring the state misfit and/or the control variable in other norms, as well as alternative weightings within the cost functionals. In particular, the methods tested here also work well for L^2-norm problems (e.g. see [36]). We consider problems of the form of (14) to create an extra level of difficulty for our solvers.

The problem is considered on a given compact spatial domain Ω, where $\Omega \subset \mathbb{R}^2$ has boundary $\partial\Omega$, and is equipped with Dirichlet boundary conditions. The algebraic inequality constraints are assumed to hold a.e. on Ω. We further note that u_a and u_b may take the form of constants, or functions in spatial variables, however we restrict our attention to the case where these represent constants.

Problems in the form of (14) are often solved numerically, by means of a discrete approximation. In the following experiments we employ the Q1 finite element discretization implemented in IFISS[2] (see [17, 18]). Applying the latter yields a sequence of nonsmooth convex programming problems, which can be transformed to the smooth form of (QP), by introducing some auxiliary variables to deal with the ℓ_1 terms appearing in the objective (see [38, Sect. 2]). In order to restrict the memory requirements of the approach, we consider an additional approximation of H in the preconditioner $P^L_{AS,(|\mathcal{N}|,0)}$. In the cases under consideration, the resulting Hessian matrix takes the following form:

[2] https://personalpages.manchester.ac.uk/staff/david.silvester/ifiss/default.htm.

Table 5 Comparison of QP preconditioners (Poisson Control: problem size and varying regularization)

n	α_1	max nnz		Time (s)		Krylov (IPM) Its.	
		$P^C_{AS,(k_c,k_r)}$	$\widehat{P}^L_{AS,(\lfloor\mathcal{N}\rfloor,0)}$	$P^C_{AS,(k_c,k_r)}$	$\widehat{P}^L_{AS,(\lfloor\mathcal{N}\rfloor,0)}$	$P^C_{AS,(k_c,k_r)}$	$\widehat{P}^L_{AS,(\lfloor\mathcal{N}\rfloor,0)}$
$2.11 \cdot 10^4$	10^{-2}	$\mathbf{3.88 \cdot 10^5}$	$4.65 \cdot 10^5$	**5.81**	5.91	1353 (13)	**868(13)**
	10^{-4}	$\mathbf{3.88 \cdot 10^5}$	$4.65 \cdot 10^5$	**6.53**	6.58	1586 (14)	**1,015(14)**
	10^{-6}	$\mathbf{3.88 \cdot 10^5}$	$4.65 \cdot 10^5$	6.75	**6.61**	1586 (14)	**1,013(14)**
$8.32 \cdot 10^4$	10^{-2}	$\mathbf{2.12 \cdot 10^6}$	$2.21 \cdot 10^6$	**22.67**	39.60	1327 (14)	**887(14)**
	10^{-4}	$\mathbf{2.12 \cdot 10^6}$	$2.21 \cdot 10^6$	**27.58**	44.14	1759 (15)	**1,054(15)**
	10^{-6}	$\mathbf{2.12 \cdot 10^6}$	$2.21 \cdot 10^6$	**24.83**	40.23	1575 (14)	**934(14)**
$3.30 \cdot 10^5$	10^{-2}	$\mathbf{1.06 \cdot 10^7}$	$1.09 \cdot 10^7$	**27.52**	75.89	246 (8)	**203(8)**
	10^{-4}	$\mathbf{1.06 \cdot 10^7}$	$1.09 \cdot 10^7$	**27.20**	76.73	246 (8)	**204(8)**
	10^{-6}	$\mathbf{1.06 \cdot 10^7}$	$1.09 \cdot 10^7$	**27.38**	77.15	246 (8)	**204(8)**
$1.32 \cdot 10^6$	10^{-2}	$5.51 \cdot 10^7$	$\mathbf{5.38 \cdot 10^7}$	**99.14**	308.79	193 (7)	**158(7)**
	10^{-4}	$5.51 \cdot 10^7$	$\mathbf{5.38 \cdot 10^7}$	**101.78**	318.35	193 (7)	**158(7)**
	10^{-6}	$5.51 \cdot 10^7$	$\mathbf{5.38 \cdot 10^7}$	**99.71**	318.01	193 (7)	**158(7)**

Bold numbers denote the best performance for a given metric

$$H = \begin{bmatrix} J_M & 0_{d,d} & 0_{d,d} \\ 0_{d,d} & \alpha_2 J_M & -\alpha_2 J_M \\ 0_{d,d} & -\alpha_2 J_M & \alpha_2 J_M \end{bmatrix},$$

where J_M is the mass matrix of size d. When non-diagonal Hessian information is utilized within the preconditioner, we approximate each block of H by its diagonal (i.e. $\widetilde{J}_M = \mathrm{Diag}(J_M)$; an approximation which is known to be optimal [47]). The resulting matrix is then further approximated as discussed in Sect. 3. From now on, the $LDL^\top$ preconditioner, which is based on an approximation of $P^L_{AS,(\lfloor\mathcal{N}\rfloor,0)}$, is referred to as $\widehat{P}^L_{AS,(\lfloor\mathcal{N}\rfloor,0)}$, in order to stress the additional level of approximation employed within the Hessian matrix. For these examples, the preconditioning strategy based on $\widehat{P}_K$ (given in Sect. 3.2) behaved significantly worse, and hence was not included in the numerical results. The preconditioner $\widehat{P}^L_{AS,(\lfloor\mathcal{N}\rfloor,0)}$ can be useful in that it allows us to employ block-diagonal preconditioners within which the Schur complement approximation takes into account non-diagonal information of the Hessian matrix H. In certain cases, this can result in a faster convergence of IP-PMM, as compared to $P^C_{AS,(k_c,k_r)}$ (see Table 6).

The first problem that we consider is the two-dimensional $\mathrm{L}^1/\mathrm{L}^2$-regularized Poisson optimal control problem, with bound constraints on the control and free state, posed on the domain $\Omega = (0,1)^2$. Following [38, Sect. 5.1], we consider the constant control bounds $\mathrm{u_a} = -2$, $\mathrm{u_b} = 1.5$, and the desired state $\bar{\mathrm{y}} = \sin(\pi x_1)\sin(\pi x_2)$. In Table 5, we fix $\alpha_2 = 10^{-2}$ (which we find to be the most numerically interesting case), and we present the runs of the method using the different preconditioning approaches, with increasing problem size, and varying L^1 regularization parameter

Table 6 Comparison of QP preconditioners (Convection–Diffusion Control: problem size and varying regularization)

n	α_1	max nnz		Time (s)		Krylov (IPM) Its.	
		$P^C_{AS,(k_c,k_r)}$	$\widehat{P}^L_{AS,(\lvert\mathcal{N}\rvert,0)}$	$P^C_{AS,(k_c,k_r)}$	$\widehat{P}^L_{AS,(\lvert\mathcal{N}\rvert,0)}$	$P^C_{AS,(k_c,k_r)}$	$\widehat{P}^L_{AS,(\lvert\mathcal{N}\rvert,0)}$
$2.11 \cdot 10^4$	10^{-2}	$\mathbf{3.88 \cdot 10^5}$	$4.65 \cdot 10^5$	15.94	**11.67**	3947 (21)	**1,903(19)**
	10^{-4}	$\mathbf{3.88 \cdot 10^5}$	$4.65 \cdot 10^5$	31.24	**15.31**	7546 (25)	**2,721(22)**
	10^{-6}	$\mathbf{3.88 \cdot 10^5}$	$4.65 \cdot 10^5$	31.29	**16.42**	7489 (25)	**2,962(23)**
$8.32 \cdot 10^4$	10^{-2}	$\mathbf{2.12 \cdot 10^6}$	$2.21 \cdot 10^6$	**49.05**	65.98	3464 (19)	**1,937(19)**
	10^{-4}	$\mathbf{2.12 \cdot 10^6}$	$2.21 \cdot 10^6$	99.49	**93.19**	7198 (25)	**2,976(23)**
	10^{-6}	$\mathbf{2.12 \cdot 10^6}$	$2.21 \cdot 10^6$	**98.82**	98.90	7150 (25)	**3,178(24)**
$3.30 \cdot 10^5$	10^{-2}	$\mathbf{1.07 \cdot 10^7}$	$1.09 \cdot 10^7$	297.96	**285.77**	4037 (21)	**1,667(18)**
	10^{-4}	$\mathbf{1.07 \cdot 10^7}$	$1.09 \cdot 10^7$	357.43	**334.89**	4971 (22)	**2,542(22)**
	10^{-6}	$\mathbf{1.07 \cdot 10^7}$	$1.09 \cdot 10^7$	372.40	**354.23**	5418 (23)	**2,530(22)**
$1.32 \cdot 10^6$	10^{-2}	$5.51 \cdot 10^7$	$\mathbf{5.38 \cdot 10^7}$	**158.17**	436.49	384 (9)	**284(9)**
	10^{-4}	$5.51 \cdot 10^7$	$\mathbf{5.38 \cdot 10^7}$	**161.12**	438.44	385 (9)	**286(9)**
	10^{-6}	$5.51 \cdot 10^7$	$\mathbf{5.38 \cdot 10^7}$	**165.57**	443.67	385 (9)	**286(9)**

Bold numbers denote the best performance for a given metric

(that is α_1). To reflect the change in the grid size, we report the number of variables of the optimization problem after transforming it to the IP-PMM format. We also report the overall number of Krylov iterations required for IP-PMM to converge (and the number of IP-PMM iterations in brackets), the maximum number of nonzeros stored in order to apply the inverses of the associated preconditioners, as well as the required CPU time.

We can draw several observations from the results in Table 5. Firstly, one can observe that in this case, a diagonal approximation of H is sufficiently good to deliver very fast convergence of MINRES. The block-diagonal preconditioner using non-diagonal Hessian information (i.e. $\widehat{P}^L_{AS,(|\mathcal{N}|,0)}$) required consistently fewer MINRES iterations (and not necessarily more memory; see the three largest experiments), however, this did not result in a reduction in CPU time. There are several reasons for this. Firstly, the Hessian of the problem becomes “no less" diagonally dominant as the problem size is increased. As a result, the diagonal approximation of it remains robust with respect to the problem size for the problem under consideration. On the other hand, the algorithm uses the built-in MATLAB function `ldl` to factorize the preconditioner $\widehat{P}^L_{AS,(|\mathcal{N}|,0)}$. While this implementation is very stable, it employs a dynamic permutation at each IP-PMM iteration, which slows down the algorithm. In this case, a specialized method using preconditioner $\widehat{P}^L_{AS,(|\mathcal{N}|,0)}$ should employ a separate symbolic factorization step, that could be used in subsequent IP-PMM iterations, thus significantly reducing the CPU time. This is not done here, however, as we treat these PDE optimization problems as black-box (notice that the implementation allows the user to feed an approximation of the Hessian, but does not allow the user to use a different $LDL^\top$ decomposition). In all the previous runs,

the reported Krylov iterations include both the predictor and the corrector steps of IP-PMM. Thus, the systems solved in each case are twice the number of IPM iterations. We should note that for the problem under consideration employing a predictor–corrector scheme is not necessary, however, we wanted to keep the implementation as general and robust as possible, without tailoring it to specific applications. For this problem, we can also observe that IP-PMM was robust with respect to the problem size (i.e. IP-PMM convergence was not significantly affected by the size of the problem). This is often observed when employing an IPM for the solution of PDE optimization problems (e.g. see [36]), however, in general one should expect dependence of the IPM on the problem size.

Next we consider the optimal control of the convection–diffusion equation, i.e. $-\epsilon\Delta \mathrm{y} + \mathrm{w} \cdot \nabla y = u$, on the domain $\Omega = (0,1)^2$, where w is the wind vector given by $\mathrm{w} = [2x_2(1-x_1^2), -2x_1(1-x_2^2)]^\top$, with control bounds $\mathrm{u_a} = -2$, $\mathrm{u_b} = 1.5$ and free state (e.g. see [38, Sect. 5.2]). Once again, the problem is discretized using Q1 finite elements, employing the Streamline Upwind Petrov–Galerkin (SUPG) upwinding scheme implemented in [24]. We define the desired state as $\bar{\mathrm{y}} = \exp(-64((x_1-0.5)^2 + (x_2-0.5)^2))$ with zero boundary conditions. The diffusion coefficient ϵ is set as $\epsilon = 0.02$. The L^2 regularization parameter α_2 is set as $\alpha_2 = 10^{-2}$. We run IP-PMM with the two different preconditioning approaches on the aforementioned problem, with different L^1 regularization values (i.e. α_1) and with increasing problem size. The results are collected in Table 6.

In Table 6 we can observe that the IP-PMM convergence is improved when the problem size is increased, which relates to the good conditioning of the Hessian. On the other hand, IP-PMM convergence is affected by the L^1 regularization parameter α_1. Unlike in the Poisson control problem, we can see clear advantages of using $\widehat{P}^{L}_{AS,(|\mathcal{M}|,0)}$ instead of $P^{C}_{AS,(k_c,k_r)}$ in this case. We can observe that in this problem using non-diagonal Hessian information within the preconditioner is significantly more important, and the reduced number of Krylov iterations often translates into a reduction of the CPU time. As before, we should mention that the reported number of Krylov iterations includes the solution of both the predictor and the corrector steps for each IP-PMM iteration.

Overall, we observe that each of the presented approaches can be very successful on a wide range of problems, including those of very large scale. Although we have treated every problem as if we knew nothing about its structure for these numerical tests, our a priori knowledge of the preconditioners and of the problem's structure could in principle aid us in selecting a preconditioner without compromising their "general purpose" nature.

5 Conclusions

In this paper we have presented several general-purpose preconditioning methodologies suitable for primal-dual regularized interior point methods, applied to convex optimization problems. All presented preconditioners are positive definite and hence can be used within symmetric solvers such as PCG or MINRES. After analyzing and discussing the different preconditioning approaches, we have presented extensive numerical results,

showcasing their use and potential benefits for different types of practical applications of convex optimization. A robust and general IP-PMM implementation, using the proposed preconditioners, has been provided for the solution of convex quadratic programming problems, and one can readily observe its ability of reliably and efficiently solving general large-scale problems, with minimal input from the user.

As a future research direction, we would like to include certain matrix-free preconditioning methodologies that could be used as alternatives for huge-scale instances that cannot be solved by means of factorization-based preconditioners, due to memory requirements.

Appendix: Proof of Theorem 2

For simplicity of notation, let $P_{AS,(k_c,k_r)} \equiv P_{AS}$. In order to provide an outline of the proof of Theorem 2, which follows trivially by extending the result in [7, Theorem 2], we need to introduce the notion of the Rayleigh quotient for symmetric matrices. The numerical range of a symmetric matrix $U \in \mathbb{R}^{n\times n}$, denoted as $q(U)$, is defined as

$$q(U) := \left\{ z \in \mathbb{R}, \text{ s.t. } z = \frac{x^\top U x}{x^\top x}, \text{ for some } x \in \mathbb{R}^n, \ x \neq 0 \right\}.$$

Given the notation of Sect. 3.1, an element of the Rayleigh quotient of these matrices is represented as:

$$\gamma_{NE} \in q\left(\widetilde{M}_{NE}\right) = [\alpha_{NE}, \beta_{NE}], \qquad \gamma_F \in q\left(\widetilde{F}\right) = [\alpha_F, \beta_F].$$

Similarly, an arbitrary element of $q(P_{NE})$ is denoted by

$$\gamma_p \in \left[\lambda_{\min}(P_{NE}), \lambda_{\max}(P_{NE})\right], \quad \lambda_{\min}(P_{NE}) \geq \delta.$$

The eigenvalues of $P_{AS}^{-1}K$ are the same as those of

$$P_{AS}^{-1/2} K P_{AS}^{-1/2} = \begin{bmatrix} \widehat{F}^{-1/2} & 0_{n,m} \\ 0_{m,n} & P_{NE}^{-1/2} \end{bmatrix} \begin{bmatrix} -F & A^\top \\ A & \delta I_m \end{bmatrix} \begin{bmatrix} \widehat{F}^{-1/2} & 0_{n,m} \\ 0_{m,n} & P_{NE}^{-1/2} \end{bmatrix} = \begin{bmatrix} -\widetilde{F} & R^\top \\ R & \delta P_{NE}^{-1} \end{bmatrix},$$

where $\widetilde{F}$ is defined in (11) and $R := P_{NE}^{-1/2} A \widehat{F}^{-1/2}$. Any eigenvalue λ of the preconditioned matrix $P_{AS}^{-1/2} K P_{AS}^{-1/2}$ must therefore satisfy

$$-\widetilde{F} w_1 + R^\top w_2 = \lambda w_1, \tag{15}$$

$$R w_1 + \delta P_{NE}^{-1} w_2 = \lambda w_2. \tag{16}$$

First, note that

$$R R^\top = P_{NE}^{-1/2} A \widehat{F}^{-1} A^\top P_{NE}^{-1/2} = P_{NE}^{-1/2} \left(\widehat{M} - \delta I_m\right) P_{NE}^{-1/2} = \widetilde{M}_{NE} - \delta P_{NE}^{-1}.$$

If $\lambda \notin [-\beta_F, -\alpha_F]$ then $\widetilde{F} + \lambda I_n$ is symmetric positive (or negative) definite; moreover $R^\top w_2 \neq 0_n$. Then from (15) we obtain

$$w_1 = \left(\widetilde{F} + \lambda I_n\right)^{-1} R^\top w_2,$$

which, after substituting in (16), yields

$$R\left(\widetilde{F} + \lambda I_n\right)^{-1} R^\top w_2 + \delta P_{NE}^{-1} w_2 = \lambda w_2.$$

Premultiplying by $w_2^\top$ and dividing by $\|w_2\|^2$, we obtain the following equation, where we have set $z = R^\top w_2$:

$$\lambda = \frac{z^\top \left(\widetilde{F} + \lambda I_n\right)^{-1} z}{z^\top z} \frac{w_2^\top R R^\top w_2}{w_2^\top w_2} + \delta \frac{w_2^\top P_{NE}^{-1} w_2}{w_2^\top w_2} = \frac{1}{\gamma_F + \lambda}\left(\gamma_{NE} - \frac{\delta}{\gamma_p}\right) + \frac{\delta}{\gamma_p}.$$

Hence, λ must satisfy the following second-order algebraic equation:

$$\lambda^2 + \left(\gamma_F - \omega\right)\lambda - \left(\omega(\gamma_F - 1) + \gamma_{NE}\right) = 0,$$

where we have set $\omega = \dfrac{\delta}{\gamma_p}$ satisfying $\omega \leq 1$. Notice that $\gamma_{NE} - \omega \geq 0$ by construction.

All bounds, except for the lower bound of I_+, follow directly by following the developments in [7, Theorem 3]. Thus, we only derive the lower bound for the positive eigenvalues of the preconditioned matrix, which can be obtained by computing a lower bound for the positive eigenvalue solution of the previous algebraic equation. In particular, we have

$$\begin{aligned}\lambda_+ &= \frac{1}{2}\left[\omega - \gamma_F + \sqrt{(\gamma_F - \omega)^2 + 4(\omega\gamma_F - \omega + \gamma_{NE})}\right] \\ &= \frac{1}{2}\left[\omega - \gamma_F + \sqrt{(\gamma_F + \omega)^2 + 4(\gamma_{NE} - \omega)}\right].\end{aligned}$$

We notice that λ_+ is a decreasing function with respect to the variable γ_F and increasing with respect to γ_{NE}. Hence, we have that:

$$\begin{aligned}\lambda_+ &\geq \frac{1}{2}\left[\omega - \beta_F + \sqrt{(\beta_F + \omega)^2 + 4(\alpha_{NE} - \omega)}\right] \\ &\geq \frac{1}{2}\left[-\beta_F + \sqrt{\beta_F^2 + 4\alpha_{NE}}\right],\end{aligned}$$

where the last inequality follows because the penultimate expression is increasing with respect to ω.

Acknowledgements The authors are grateful to the two anonymous referees for their valuable comments. JG and SP were supported by the Google project "Fast $(1 + x)$-order Methods for

Linear Programming". JWP gratefully acknowledges support from the Engineering and Physical Sciences Research Council (UK) grant EP/S027785/1. We wish to remark that this study does not have any conflict of interest to disclose.

Data availability The linear programming problem data used in this paper can be accessed from https://netlib.org/lp. The quadratic programming instances taken from the Maros–Mészáros test set can be accessed from https://www.doc.ic.ac.uk/~im/. Finally, the PDE-constrained optimization instances have been generated using the code given in https://personalpages.manchester.ac.uk/staff/david.silvester/ifiss/, which is provided to the public under the GNU Lesser General Public License 2.1.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Altman, A., Gondzio, J.: Regularized symmetric indefinite systems in interior point methods for linear and quadratic optimization. Optim. Methods Softw. **11**(1–4), 275–302 (1999). https://doi.org/10.1080/10556789908805754
2. Andersen, E.D., Gondzio, J., Mészáros, C., Xu, X.: Implementation of interior point methods for large scale linear programming. In: Terlaky, T. (ed.) Interior Point Methods in Mathematical Programming. Kluwer Academic Publishers, Dordrecht, 199–252 (1996)
3. Armand, P., Omheni, R.: A mixed logarithmic barrier-augmented Lagrangian method for nonlinear optimization. J. Optim. Theor. Appl. **173**, 523–547 (2017). https://doi.org/10.1007/s10957-017-1071-x
4. Bellavia, S., De Simone, V., di Serafino, D., Morini, B.: On the update of constraint preconditioners for regularized KKT systems. Comput. Optim. Appl. **65**, 339–360 (2016). https://doi.org/10.1007/s10589-016-9830-4
5. Benzi, M.: Preconditioning techniques for large linear systems: a survey. J. Comput. Phys. **182**(2), 418–477 (2002). https://doi.org/10.1006/jcph.2002.7176
6. Benzi, M., Golub, G.H., Liesen, J.: Numerical solution of saddle point systems. Acta Numer. **14**, 1–137 (2005). https://doi.org/10.1017/S0962492904000212
7. Bergamaschi, L., Gondzio, J., Martínez, A., Pearson, J.W., Pougkakiotis, S.: A new preconditioning approach for an interior point-proximal method of multipliers for linear and convex quadratic programming. Numer. Linear Algebra Appl. **28**(4), e2361 (2020). https://doi.org/10.1002/nla.2361
8. Bergamaschi, L., Gondzio, J., Venturin, M., Zilli, G.: Inexact constraint preconditioners for linear systems arising in interior point methods. Comput. Optim. Appl. **36**, 137–147 (2007). https://doi.org/10.1007/s10589-006-9001-0. See also Erratum, Comput. Optim. Appl. **49**, 401–406 (2011)
9. Bergamaschi, L., Gondzio, J., Zilli, G.: Preconditioning indefinite systems in interior point methods for optimization. Comput. Optim. Appl. **28**, 149–171 (2004). https://doi.org/10.1023/B:COAP.0000026882.34332.1b
10. Bocanegra, S., Campos, F., Oliveira, A.: Using a hybrid preconditioner for solving large-scale linear systems arising from interior point methods. Comput. Optim. Appl. **36**, 149–164 (2007). https://doi.org/10.1007/s10589-006-9009-5

11. Cafieri, S., D'Appuzo, M., De Simone, V., di Serafino, D.: Stopping criteria for inner iterations in inexact potential reduction methods: a computational study. Comput. Optim. Appl. **36**, 165–193 (2007). https://doi.org/10.1007/s10589-006-9007-7
12. D'Apuzzo, M., De Simone, V., di Serafino, D.: On mutual impact of numerical linear algebra and large-scale optimization with focus on interior point methods. Comput. Optim. Appl. **45**, 283–310 (2010). https://doi.org/10.1007/s10589-008-9226-1
13. De Simone, V., di Serafino, D., Gondzio, J., Pougkakiotis, S., Viola, M.: Sparse approximations with interior point methods. SIAM Rev. **64**(4), 954–988 (2022). https://doi.org/10.1137/21M1401103
14. di Serafino, D., Orban, D.: Constraint-preconditioned Krylov solvers for regularized saddle-point systems. SIAM J. Sci. Comput. **43**(2), A1001–A1026 (2021). https://doi.org/10.1137/19M1291753
15. Dollar, H.S.: Constraint-style preconditioners for regularized saddle point problems. SIAM J. Matrix Anal. Appl. **29**(2), 672–684 (2007). https://doi.org/10.1137/050626168
16. Dollar, H.S., Gould, N.I.M., Schilders, W.H.A., Wathen, A.J.: Using constraint preconditioners with regularized saddle-point problems. Comput. Optim. Appl. **36**, 249–270 (2007). https://doi.org/10.1007/s10589-006-9004-x
17. Elman, H.C., Ramage, A., Silvester, D.J.: Algorithm 866: IFISS, a Matlab toolbox for modelling incompressible flow. ACM Trans. Math. Softw. **33**(2), 14 (2007). https://doi.org/10.1145/1236463.1236469
18. Elman, H.C., Ramage, A., Silvester, D.J.: IFISS: A computational laboratory for investigating incompressible flow problems. SIAM Rev. **52**(2), 261–273 (2014). https://doi.org/10.1137/120891393
19. Friedlander, M.P., Orban, D.: A primal-dual regularized interior-point method for convex quadratic programs. Math. Program. Comput. **4**, 71–107 (2012). https://doi.org/10.1007/s12532-012-0035-2
20. Gill, P.E., Murray, W., Ponceleón, D.B., Saunders, M.A.: Preconditioners for indefinite systems arising in optimization. SIAM J. Matrix Anal. Appl. **13**(1), 292–311 (1992). https://doi.org/10.1137/0613022
21. Greenbaum, A.: Iterative Methods for Solving Linear Systems. Frontiers in Applied Mathematics. SIAM, Philadelphia, USA (1997). https://doi.org/10.1137/1.9781611970937
22. Greenbaum, A., Pták, V., Strakoš, Z.: Any nonincreasing convergence curve is possible for GMRES. SIAM J. Matrix Anal. Appl. **17**(3), 465–469 (1996). https://doi.org/10.1137/S0895479894275030
23. Hestenes, M.R., Stiefel, E.: Method of conjugate gradients for solving linear systems. J. Res. Natl. Bur. Stand. **49**(6), 409–436 (1952). https://doi.org/10.6028/jres.049.044
24. Hughes, T.J.R., Brooks, A.N.: A multidimensional upwind scheme with no crosswind diffusion. In: Hughes, T.J.R. (ed.) Finite Element Methods for Convection Dominated Flows, vol. 34, pp. 19–35. American Society of Mechanical Engineers (1979)
25. Ipsen, I.C.F.: A note on preconditioning non-symmetric matrices. SIAM J. Sci. Comput. **23**(3), 1050–1051 (2001). https://doi.org/10.1137/S1064827500377435
26. Keller, C., Gould, N.I.M., Wathen, A.J.: Constraint preconditioning for symmetric indefinite linear systems. SIAM J. Matrix Anal. Appl. **21**(4), 1300–1317 (2000). https://doi.org/10.1137/S0895479899351805
27. Li, X., Sun, D., Toh, K.C.: An asymptotically superlinearly convergent semismooth Newton augmented Lagrangian method for linear programming. SIAM J. Optim. **30**(3), 2410–2440 (2020). https://doi.org/10.1137/19M1251795
28. Maros, I., Mészáros, C.: The role of the augmented system in interior point methods. Eur. J. Oper. Res. **107**(3), 720–736 (1998). https://doi.org/10.1016/S0377-2217(97)00074-X
29. Maros, I., Mészáros, C.: A repository of convex quadratic programming problems. Optim. Methods Softw. **11**(1–4), 671–681 (1999). https://doi.org/10.1080/10556789908805768
30. Murphy, M.F., Golub, G.H., Wathen, A.J.: A note on preconditioning for indefinite linear systems. SIAM J. Sci. Comput. **21**(6), 1969–1972 (2000). https://doi.org/10.1137/S1064827599355153
31. Netlib: http://netlib.org/lp (2011)
32. Notay, Y.: A new analysis of block preconditioners for saddle point systems. SIAM J. Matrix Anal. Appl. **35**(1), 143–173 (2014). https://doi.org/10.1137/130911962
33. Oliveira, A.R.L., Sorensen, D.C.: A new class of preconditioners for large-scale linear systems from interior point methods for linear programming. Linear Algebra Appl. **394**, 1–24 (2005). https://doi.org/10.1016/j.laa.2004.08.019

34. Orban, D.: Limited-memory LDL^T factorization of symmetric quasi-definite matrices with application to constrained optimization. Numer. Algorithms **70**, 9–41 (2015). https://doi.org/10.1007/s11075-014-9933-x
35. Paige, C.C., Saunders, M.A.: Solution of sparse indefinite systems of linear equations. SIAM J. Numer. Anal. **12**(4), 617–629 (1975). https://doi.org/10.1137/0712047
36. Pearson, J.W., Gondzio, J.: Fast interior point solution of quadratic programming problems arising from PDE-constrained optimization. Numer. Math. **137**, 959–999 (2017). https://doi.org/10.1007/s00211-017-0892-8
37. Pearson, J.W., Pestana, J.: Preconditioners for Krylov subspace methods: an overview. GAMM-Mitt. **43**(4), e202000015 (2020). https://doi.org/10.1002/gamm.202000015
38. Pearson, J.W., Porcelli, M., Stoll, M.: Interior-point methods and preconditioning for PDE-constrained optimization problems involving sparsity terms. Numer. Linear Algebra Appl. **27**(2), e2276 (2019). https://doi.org/10.1002/nla.2276
39. Pougkakiotis, S., Gondzio, J.: Dynamic non-diagonal regularization in interior point methods for linear and convex quadratic programming. J. Optim. Theory Appl. **181**(3), 905–945 (2019). https://doi.org/10.1007/s10957-019-01491-1
40. Pougkakiotis, S., Gondzio, J.: An interior point-proximal method of multipliers for convex quadratic programming. Comput. Optim. Appl. **78**, 307–351 (2021). https://doi.org/10.1007/s10589-020-00240-9
41. Pougkakiotis, S., Gondzio, J.: An interior point-proximal method of multipliers for linear positive semi-definite programming. J. Optim. Theory Appl. **192**, 97–129 (2022). https://doi.org/10.1007/s10957-021-01954-4
42. Saad, Y., Schultz, M.H.: GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Comput. **7**(3), 856–869 (1986). https://doi.org/10.1137/0907058
43. Saunders, M., Tomlin, J.A.: Solving regularized linear programs using barrier methods and KKT systems. Tech Rep SOL 96-4. Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford (1996)
44. Scott, J., Tůma, M.: A Schur complement approach to preconditioning sparse linear least-squares problems with some dense rows. Numer. Algorithms **79**, 1147–1168 (2018). https://doi.org/10.1007/s11075-018-0478-2
45. Silvester, D.J., Wathen, A.J.: Fast iterative solution of stabilized Stokes systems. Part II: using general block preconditioners. SIAM J. Numer. Anal. **31**(5), 1352–1367 (1994). https://doi.org/10.1137/0731070
46. Vanderbei, R.J.: Symmetric quasidefinite matrices. SIAM J. Optim. **5**(1), 100–113 (1995). https://doi.org/10.1137/0805005
47. Wathen, A.J.: Realistic eigenvalue bounds for the Galerkin mass matrix. IMA J. Numer. Anal. **7**(4), 449–457 (1987). https://doi.org/10.1093/imanum/7.4.449
48. Wathen, A.J.: Preconditioning. Acta Numer. **24**, 329–376 (2015). https://doi.org/10.1017/S0962492915000021