SIAM J. MATRIX ANAL. APPL.
Vol. 37, No. 3, pp. 1103–1128

NULL-SPACE PRECONDITIONERS FOR SADDLE POINT SYSTEMS*

JENNIFER PESTANA† AND TYRONE REES‡

Abstract. The null-space method is a technique that has been used for many years to reduce a saddle point system to a smaller, easier to solve, symmetric positive definite system. This method can be understood as a block factorization of the system. Here we explore the use of preconditioners based on incomplete versions of a particular null-space factorization and compare their performance with the equivalent Schur complement based preconditioners. We also describe how to apply the nonsymmetric preconditioners proposed using the conjugate gradient method (CG) with a nonstandard inner product. This requires an exact solve with the (1,1) block, and the resulting algorithm is applicable in other cases where Bramble–Pasciak CG is used. We verify the efficiency of the newly proposed preconditioners on a number of test cases from a range of applications.

Key words. preconditioning, saddle point systems, null-space method

AMS subject classifications. 65F08, 65F10, 65F50

DOI. 10.1137/15M1021349

1. Introduction.

We consider the fast solution of saddle point systems of the form

$$\mathcal{A}\boldsymbol{w} = \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} = \boldsymbol{b}, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times n}$, $m \leq n$. We suppose that B is of full rank, and that A is symmetric positive semidefinite, and positive definite on the kernel of B.

Solving systems with this structure is a vital component in numerous scientific computing algorithms; for example, such systems arise naturally in constrained optimization problems [7, 13, 15, 25], Stokes and Navier–Stokes equations in fluid mechanics [5, 9, 16], time-harmonic Maxwell equations [26, 32], and the application of Kirchhoff's laws in circuit simulation [46, 50]. For an overview of solution methods, see the survey paper of Benzi, Golub, and Liesen [4] and the references therein.

The *null-space method* is a technique for solving systems of the form (1.1). This method requires a particular solution $\widehat{\boldsymbol{x}} \in \mathbb{R}^n$ such that $B\widehat{\boldsymbol{x}} = \boldsymbol{g}$, and a matrix Z whose columns span the null-space of B. Then, since $\boldsymbol{x} = Z\boldsymbol{x}_{\mathrm{n}} + \widehat{\boldsymbol{x}}$, we can solve (1.1) by first finding $\boldsymbol{x}_{\mathrm{n}}$ from

$$Z^T A Z \boldsymbol{x}_{\mathrm{n}} = Z^T(\boldsymbol{f} - A\widehat{\boldsymbol{x}}) \tag{1.2}$$

and then recovering $\boldsymbol{y}$ from the overdetermined system $B^T\boldsymbol{y} = \boldsymbol{f} - A\boldsymbol{x}$; see, e.g., [4, Chapter 6] for more details. There exist many methods for obtaining a null-space

*Received by the editors May 13, 2015; accepted for publication (in revised form) by D. Orban June 16, 2016; published electronically August 18, 2016.

http://www.siam.org/journals/simax/37-3/M102134.html

Funding: The work of the first author was completed at the University of Manchester and was supported by Engineering and Physical Sciences Research Council grant EP/I005293. The work of the second author was supported by Engineering and Physical Sciences Research Council grant EP/I013067/1.

†Department of Mathematics and Statistics, University of Strathclyde, Glasgow G1 IX4, UK (jennifer.pestana@strath.ac.uk).

‡STFC Rutherford Appleton Laboratory, Chilton, Didcot, Oxfordshire OX11 0QX, UK (tyrone.rees@stfc.ac.uk).

matrix Z, but a common choice is the fundamental basis,

$$Z_f := \begin{bmatrix} -B_1^{-1}B_2 \\ I \end{bmatrix}, \tag{1.3}$$

where $B = [B_1 \ B_2]$ and, without loss of generality, $B_1 \in \mathbb{R}^{m\times m}$ is nonsingular. Note that because the rank of B is m, we can always permute B to obtain an invertible B_1.

Even for sparse $\mathcal{A}$ the symmetric positive definite matrix Z^TAZ is often dense, and forming it could be costly. For larger problems, therefore, it may be better to solve (1.1) rather than (1.2). It is possible to write down a number of matrix factorizations that are equivalent to the null-space method—recently Rees and Scott gave an overview [44]. We can obtain one particular factorization of $\mathcal{A}$ which falls into this category by taking a 3×3 blocking of $\mathcal{A}$ as

$$\mathcal{A} = \begin{bmatrix} A_{11} & A_{12} & B_1^T \\ A_{21} & A_{22} & B_2^T \\ B_1 & B_2 & 0 \end{bmatrix},$$

where $B_1 \in \mathbb{R}^{m\times m}$ is, as in (1.3), assumed to be nonsingular. Consider a permuted version of $\mathcal{A}$,

$$\widehat{\mathcal{A}} := \Pi\mathcal{A}\Pi^T = \left[\begin{array}{cc|c} A_{11} & B_1^T & A_{12} \\ B_1 & 0 & B_2 \\ \hline A_{21} & B_2^T & A_{22} \end{array}\right] =: \begin{bmatrix} \widehat{A} & \widehat{B}^T \\ \widehat{B} & A_{22} \end{bmatrix}, \quad \Pi = \begin{bmatrix} I_m & & \\ & & I_m \\ & I_{n-m} & \end{bmatrix}, \tag{1.4}$$

where I_m is the identity matrix of dimension m. The matrix $\widehat{A}$ is invertible with inverse

$$\widehat{A}^{-1} = \begin{bmatrix} 0 & B_1^{-1} \\ B_1^{-T} & -B_1^{-T}A_{11}B_1^{-1} \end{bmatrix}.$$

Therefore, we can apply the standard block-LDL^T factorization for saddle point systems [4, equation (3.1)], obtaining

$$\widehat{\mathcal{A}} = \begin{bmatrix} I & 0 \\ \widehat{B}\widehat{A}^{-1} & I \end{bmatrix} \begin{bmatrix} \widehat{A} & 0 \\ 0 & A_{22} - \widehat{B}\widehat{A}^{-1}\widehat{B}^T \end{bmatrix} \begin{bmatrix} I & \widehat{A}^{-1}\widehat{B}^T \\ 0 & I \end{bmatrix}. \tag{1.5}$$

This factorization has a connection with standard Schur complement preconditioners and the range-space method. However, because of the permutation, it is also closely related to the null-space method with the fundamental basis since, using (1.3), we can rewrite (1.5) as

$$\widehat{\mathcal{A}} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ B_2^TB_1^{-T} & XB_1^{-1} & I \end{bmatrix} \begin{bmatrix} A_{11} & B_1^T & 0 \\ B_1 & 0 & 0 \\ 0 & 0 & N \end{bmatrix} \begin{bmatrix} I & 0 & B_1^{-1}B_2 \\ 0 & I & B_1^{-T}X^T \\ 0 & 0 & I \end{bmatrix}, \tag{1.6}$$

where

$$X := Z_f^T \begin{bmatrix} A_{11} \\ A_{21} \end{bmatrix} \text{ and } N := Z_f^TAZ_f = A_{22} - \widehat{B}\widehat{A}^{-1}\widehat{B}^T. \tag{1.7}$$

Our assumption that A is positive definite on the null-space of B means that the null-space matrix N is symmetric positive definite. The approach in section 2 of Rees and

Scott [44] applied here shows that solving $\widehat{\mathcal{A}}(\Pi \boldsymbol{w}) = \Pi \boldsymbol{b}$ is equivalent to the null-space method with the fundamental null-space (1.3) and the particular solution

$$\widehat{\boldsymbol{x}} = \begin{bmatrix} B_1^{-1}\boldsymbol{g} \\ \mathbf{0} \end{bmatrix}. \tag{1.8}$$

Applying the inverse permutations Π and Π^T in (1.4) to $\widehat{\mathcal{A}}$ in (1.6), we can easily recover $\mathcal{A}$.

One interpretation of the null-space method is therefore that the null-space matrix $Z_f^T A Z_f$ is equivalent to the Schur complement of the subblock $\widehat{A}$ in $\widehat{\mathcal{A}}$. Note that when the block A is invertible, the null-space matrix is equivalent to the Schur complement of the dual saddle point system

$$\begin{bmatrix} A^{-1} & Z \\ Z^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{x}_{\mathrm{n}} \end{bmatrix} = \begin{bmatrix} -\widehat{\boldsymbol{x}} \\ -Z^T \boldsymbol{f} \end{bmatrix}$$

of (1.1) [4, page 32].

In this paper we present four preconditioners based on incomplete or approximate versions of the factorization (1.6). Since these preconditioners are intimately related to the Schur complement decomposition, we can apply them using conjugate gradients in a nonstandard inner product [10]. However, the usual application of such preconditioners requires certain quantities to be symmetric positive definite, and this is typically attained by scaling the blocks appropriately. In our case this is not possible without destroying the structure that we exploit, since we assume that we can solve with a certain submatrix of $\mathcal{A}$ exactly. Accordingly, we extend the current theory of conjugate gradients in a nonstandard inner product to allow for the case where one of the subblock solves is exact.

The rest of this paper is organized as follows. In section 2, we give an eigenanalysis of the preconditioned systems and show that the eigenvalues of the preconditioned matrices are clustered when a good approximation to the null-space matrix (1.7) is known. We describe the nonstandard inner product CG in section 3 and describe how our constraint preconditioner can be used within a projected Krylov subspace method. In section 4, we compare our preconditioners to standard Schur complement based methods based on the factorization

$$\begin{bmatrix} A & B^T \\ B & -C \end{bmatrix} = \begin{bmatrix} I & 0 \\ BA^{-1} & I \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & -S \end{bmatrix} \begin{bmatrix} I & A^{-1}B^T \\ 0 & I \end{bmatrix}, \tag{1.9}$$

where $S = C + BA^{-1}B^T$ is the Schur complement of A. Note that this factorization can only be applied if A is invertible, whereas the null-space method may be applied even when A is singular. The problems we consider range from academic to practical and illustrate the merits and limitations of our preconditioners. We give some conclusions in section 5.

2. Null-space preconditioners. Taking incomplete versions of the block LDL^T factorization (1.9) has proved to be an effective way of constructing preconditioners for saddle point problems—see, e.g., [7, 16, 27, 30, 31, 43, 51]. The key component of such methods is an approximation of the matrix S, which is often dense. In the following we present, and give theory for, preconditioners based on the alternative

decomposition (1.5). In particular, we use the null-space decomposition

$$\mathcal{A} = \underbrace{\begin{bmatrix} I & 0 & 0 \\ B_2^T B_1^{-T} & I & X B_1^{-1} \\ 0 & 0 & I \end{bmatrix}}_{\mathcal{L}} \underbrace{\begin{bmatrix} A_{11} & 0 & B_1^T \\ 0 & N & 0 \\ B_1 & 0 & 0 \end{bmatrix}}_{\mathcal{D}} \underbrace{\begin{bmatrix} I & B_1^{-1} B_2 & 0 \\ 0 & I & 0 \\ 0 & B_1^{-T} X^T & I \end{bmatrix}}_{\mathcal{L}^T} \tag{2.1}$$

as the basis of our preconditioners; see [4, equation 10.35], [44]. We replace N by a symmetric positive definite approximation $\widetilde{N}$ and possibly drop one or both of $\mathcal{L}$ and $\mathcal{L}^T$.

2.1. The central-null preconditioner. First, we consider the preconditioner formed by dropping both the $\mathcal{L}$ and $\mathcal{L}^T$ terms from the factorization (2.1),

$$\mathcal{P}_{cn} = \begin{bmatrix} A_{11} & 0 & B_1^T \\ 0 & \tilde{N} & 0 \\ B_1 & 0 & 0 \end{bmatrix},$$

where $\tilde{N} \approx N = Z_f^T A Z_f$. This corresponds to the block diagonal preconditioner in the decomposition (1.9), because $\Pi \mathcal{P}_{cn} \Pi^T$ is block diagonal. Accordingly, the cost of applying $\mathcal{P}_{cn}$ is a solve with each of B_1, B_1^T, and $\widetilde{N}$, as well as a matrix-vector product with A_{11}. Note that in our numerical experiments we use the matrix $\mathcal{P}_{cn}$ rather than $\Pi \mathcal{P}_{cn} \Pi^T$, but either could be implemented in practice. This comment also applies to the other preconditioners we introduce.

First, we give an eigenanalysis of the preconditioned system.

THEOREM 2.1. *Let $\tilde{N}$ be a symmetric positive definite approximation to N. Then the generalized eigenvalues λ of*

$$\begin{bmatrix} A_{11} & A_{12} & B_1^T \\ A_{21} & A_{22} & B_2^T \\ B_1 & B_2 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{y} \end{bmatrix} = \lambda \begin{bmatrix} A_{11} & 0 & B_1^T \\ 0 & \tilde{N} & 0 \\ B_1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{y} \end{bmatrix} \tag{2.2}$$

satisfy $\lambda = 1$, or

$$\lambda = \frac{\mu + \sigma \pm \sqrt{(\sigma + \mu)^2 - 4\mu}}{2\mu}, \tag{2.3}$$

where $\sigma = \boldsymbol{y}^T A_{22} \boldsymbol{y} / \boldsymbol{y}^T N \boldsymbol{y}$ and $\mu = \boldsymbol{y}^T \widetilde{N} \boldsymbol{y} / \boldsymbol{y}^T N \boldsymbol{y}$.

Proof. Since Π from (1.4) is orthogonal, the eigenvalues of $\mathcal{P}_{cn}^{-1} \mathcal{A}$ are the same as those of $\Pi \mathcal{P}_{cn}^{-1} \mathcal{A} \Pi^T = (\Pi \mathcal{P}_{cn} \Pi^T)^{-1} (\Pi \mathcal{A} \Pi^T)$, and therefore the eigenvalues needed are those of the generalized eigenvalue problem

$$\begin{bmatrix} \widehat{A} & \widehat{B}^T \\ \widehat{B} & A_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \lambda \begin{bmatrix} \widehat{A} & \\ & \widetilde{N} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}. \tag{2.4}$$

The first block row of (2.4) gives $\widehat{B}^T \boldsymbol{y} = (\lambda - 1) \widehat{A} \boldsymbol{x}$, which implies that $\lambda = 1$ or $\boldsymbol{x} = \widehat{A}^{-1} \widehat{B}^T \boldsymbol{y} / (\lambda - 1)$. The second block row of (2.4) gives $\widehat{B} \boldsymbol{x} + A_{22} \boldsymbol{y} = \lambda \widetilde{N} \boldsymbol{y}$. If we assume that $\lambda \neq 1$, then substituting for $\boldsymbol{x}$ gives that

$$\widehat{B} \widehat{A}^{-1} \widehat{B}^T \boldsymbol{y} + (\lambda - 1) A_{22} \boldsymbol{y} = \lambda (\lambda - 1) \widetilde{N} \boldsymbol{y}.$$

Using (1.7), we have that $(\lambda A_{22} - N)\boldsymbol{y} = \lambda(\lambda - 1)\widetilde{N}\boldsymbol{y}$. Premultiplying by $\boldsymbol{y}^T$ and setting $\mu := \boldsymbol{y}^T\widetilde{N}\boldsymbol{y}/\boldsymbol{y}^T N\boldsymbol{y}$, we get $\boldsymbol{y}^T N\boldsymbol{y} - \lambda\boldsymbol{y}^T A_{22}\boldsymbol{y} = \lambda(1-\lambda)\mu\boldsymbol{y}^T N\boldsymbol{y}$, and hence $\lambda^2 - (1 + \sigma/\mu)\lambda + 1/\mu = 0$, where $\sigma = \boldsymbol{y}^T A_{22}\boldsymbol{y}/\boldsymbol{y}^T N\boldsymbol{y}$.

Thus,

$$\lambda = \frac{1 + \sigma/\mu \pm \sqrt{(\sigma/\mu)^2 + (2\sigma - 4)/\mu + 1}}{2} = \frac{\mu + \sigma \pm \sqrt{(\sigma+\mu)^2 - 4\mu}}{2\mu},$$

as required. ∎

We now present a few results that give a better understanding of the behavior of these eigenvalues.

COROLLARY 2.2. *Suppose that λ is a real eigenvalue of the generalized eigenvalue problem* (2.2). *Then*

$$\frac{1}{(\mu+\sigma)_{\max}} \le \lambda < 1 + \lambda_{max}(\widetilde{N}^{-1}A_{22}),$$

where $0 \le \lambda_{max}(\widetilde{N}^{-1}A_{22})$ is the largest eigenvalue of $\widetilde{N}^{-1}A_{22}$ and $(\mu+\sigma)_{\max}$ is the largest value of $\mu + \sigma$.

Proof. For λ to be real we must have that $(\mu+\sigma)^2 \ge 4\mu$. The larger of the eigenvalues in (2.3), λ_+, satisfies

$$\lambda_+ = \frac{1}{2\mu}\left(\mu + \sigma + \sqrt{(\mu+\sigma)^2 - 4\mu}\right) < \frac{1}{2\mu}\left(\mu + \sigma + \sqrt{(\mu+\sigma)^2}\right) = 1 + \frac{\sigma}{\mu}.$$

Now, since $\sigma/\mu = \boldsymbol{y}^T A_{22}\boldsymbol{y}/\boldsymbol{y}^T\widetilde{N}\boldsymbol{y} \le \lambda_{max}(\widetilde{N}^{-1}A_{22})$, we obtain the upper bound.

Now consider the smaller eigenvalue, λ_-. Note that $\lambda_- = (\gamma - \sqrt{\gamma^2 - \delta})/2$, where $\gamma = 1 + \sigma/\mu$, $\delta = 4/\mu$, and $\delta \le \gamma^2$. For any $x \in [0,1]$, $1 - \sqrt{1-x} \ge x/2$, and so

$$\lambda_- \ge \frac{1}{4}\frac{\delta}{\gamma} = \frac{1}{\mu+\sigma} \ge \frac{1}{(\mu+\sigma)_{\max}}.$$ ∎

It is clear from the lower bound that the real eigenvalues are positive.

Remark 2.3. In the ideal case where $\mu_{\min} = \mu_{\max} = 1$, which corresponds to $\widetilde{N} = N$, the lower bound becomes $\lambda_- > \frac{1}{1+\sigma_{\max}}$. Furthermore, in this case the eigenvalues are all real if $\sigma_{\min} \ge 1$.

COROLLARY 2.4. *The complex eigenvalues of* (2.2) *with nonzero imaginary part satisfy*

$$\frac{1}{2} \le \mathrm{Re}(\lambda) < \frac{1}{\sqrt{\mu_{\min}}}, \qquad |\mathrm{Im}(\lambda)| < \frac{1}{\sqrt{\mu_{\min}}}.$$

Proof. Since the eigenvalues of $N^{-1}A_{22}$ are nonnegative, any complex eigenvalue λ satisfies $\mathrm{Re}(\lambda) \ge \frac{1}{2}$. Furthermore, $\mathrm{Re}(\lambda) \le 1/2\,(1 + \sigma_{\max}/\mu_{\min}) < 1/\sqrt{\mu_{\min}}$, with the last step holding since $\mu + \sigma < 2\sqrt{\mu} \Rightarrow (\mu+\sigma)/2\mu < 1/\sqrt{\mu_{\min}}$. Now, the imaginary part satisfies

$$|\mathrm{Im}(\lambda)|^2 = \frac{(\sigma+\mu)^2 - 4\mu}{4\mu^2} \le \frac{(\sigma+\mu)^2}{4\mu^2} < \frac{1}{\mu_{\min}}.$$ ∎

Note that the bound on the real part only holds for eigenvalues with a nontrivial imaginary part. For real eigenvalues there may be some $\lambda < 0.5$ if any eigenvalue of

TABLE 1

Extreme values of μ and σ for $\widetilde{N} = I$ and for incomplete Cholesky (IC) factorizations of N. The IC drop tolerance is given in parentheses.

	AUG3CD			MOSARQP1			STCQP2		
	I	IC (10^{-3})	IC (10^{-4})	I	IC (10^{-1})	IC (10^{-2})	I	IC (10^{-1})	IC (10^{-2})
μ	[0.0029,0.5]	[0.53,1.8]	[0.93,1.1]	[0.088,0.47]	[0.33,1.6]	[0.91,1.1]	[0.0019,0.33]	[0.23,4.7]	[0.33,2.1]
σ	[0.0058, 1]			[0.21, 1]			[1, 1]		

$N^{-1}A_{22}$ is larger than 1.5. It is also worth remarking that complex eigenvalues can be bounded independently of σ.

One final special case we wish to highlight is when A_{22} is zero and $\mu_{\min} = \mu_{\max} = 1$. Here the central-null preconditioned matrix has only three distinct eigenvalues, $\{1, \frac{1\pm\sqrt{3}i}{2}\}$, and this guarantees fast convergence of certain Krylov subspace methods. (We could also obtain these same three eigenvalues in this special case by Schur complement based arguments, following approaches found in, e.g., [18, 38].) An example of where this structure arises naturally is in the interior point method for linear programs [54], where A_{22} approaches zero at convergence. Note that when $\mu_{\min} \approx 1$ and $\mu_{\max} \approx 1$, that is, when we have a good approximation $\widetilde{N}$ to N, the above conclusions for the ideal case are approximately satisfied.

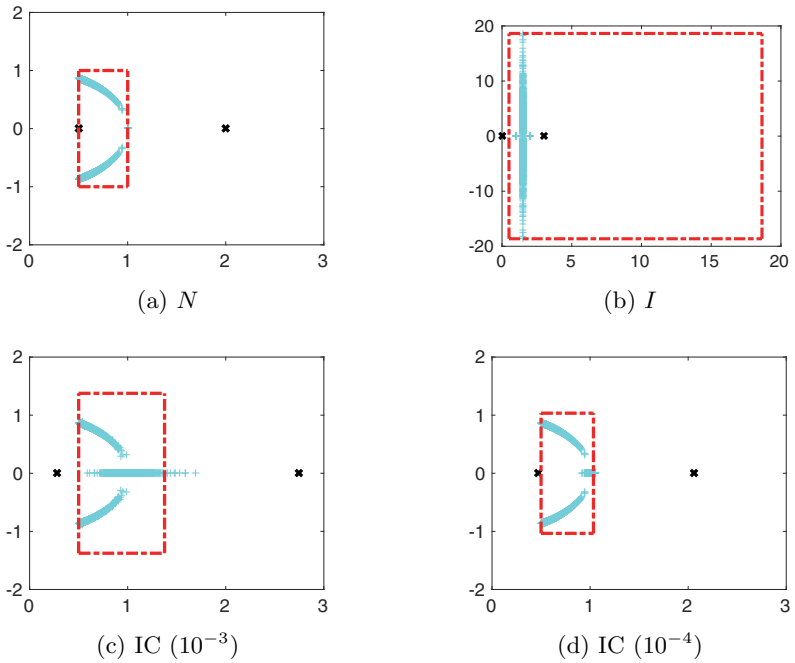
To conclude this section we examine the bounds in Corollaries 2.2 and 2.4 for the matrices AUG3DC, MOSARQP1, and STCQP2 from the CUTEst test set (see section 4.2 for details). We first tabulate μ and σ (see Table 1) and find that, as expected, $\mu_{\min}$ and $\mu_{\max}$ are close to 1 when $\widetilde{N}$ is a good approximation of N, and are often far from 1 otherwise. Note that if $\widetilde{N} = N$, then $\mu_{\min} = \mu_{\max} = 1$. In contrast, σ depends only on the properties of $\mathcal{A}$. For STCQP2, $B_2 = 0$, and so $A_{22} = N$ and $\sigma_{\min} = \sigma_{\max} = 1$. However, $\sigma_{\min} \ll 1$ for AUG3DC or MOSARQP1. For all three matrices $\sigma_{\max} = 1$, although this is not guaranteed in general.

In Figures 1–3 we examine the effect of μ and σ on the eigenvalues of $\mathcal{P}_{cn}^{-1}\mathcal{A}$ and the bounds in Corollaries 2.2 and 2.4. We first note that real eigenvalues seem to cluster near 1 when $\mu_{\min}$ and $\mu_{\max}$ are close to 1, but may be far from 1 when $\widetilde{N}$ is not a good approximation. For all three matrices, the lower bound on real eigenvalues in Corollary 2.2 is about half the value of the smallest real eigenvalue, except when $\mu_{\min}$ is much smaller than 1, i.e., when $\widetilde{N}$ does not approximate N well. For AUG3DC the upper bound is also approximately twice as big as the largest real eigenvalue, while for MOSARQP1 it is up to four times as big. However, for STCQP2 (for which $\sigma_{\min} = \sigma_{\max} = 1$) the upper bound is a fairly good approximation, and is particularly descriptive when $\widetilde{N}$ is not as good an approximation of N.

Our bounds for complex eigenvalues also indicate that a good approximation of N ensures that the complex eigenvalues are bounded in a small region away from the origin. For both AUG3DC and MOSARQP1, the bounds on the real part of complex eigenvalues in Corollary 2.4 seem most descriptive when $\mu_{\min}$ is close to 1, and the upper bound is particularly poor when $\widetilde{N} = I$. On the other hand, the upper bound on the imaginary part of complex eigenvalues seems to be fairly descriptive even when $\widetilde{N}$ is a poor approximation to N. (Note that all eigenvalues of STCQP2 are real since $A_{22} = N$.)

Overall, these results indicate that for these matrices the quality of $\widetilde{N}$ has the greatest effect on the location of eigenvalues, and that our bounds are reasonably descriptive when $\widetilde{N}$ is a reasonable approximation of N. However, the distribution of eigenvalues within the region prescribed by the bounds is more difficult to ascertain and is influenced by the eigenvalues of $N^{-1}A_{22}$. This eigenvalue distribution can

affect the convergence of Krylov methods for the central-null preconditioner, as we will see in section 4.



(a) N (b) I

(c) IC (10^{-3}) (d) IC (10^{-4})

FIG. 1. *AUG3DC: The eigenvalues (+) of $\mathcal{P}_{cn}^{-1}\mathcal{A}$ and the bounds in Corollaries* 2.2 *and* 2.4 *for different choices of $\tilde{N}$. The black crosses show the bounds on real eigenvalues, and the red dashed lines show the bounds on complex eigenvalues with nonzero imaginary part.*

2.2. The lower-null and upper-null preconditioners. Next, we drop the $\mathcal{L}^T$-term in (2.1) to form the preconditioner

$$\mathcal{P}_{ln} := \begin{bmatrix} A_{11} & 0 & B_1^T \\ A_{21} & \tilde{N} & B_2^T \\ B_1 & 0 & 0 \end{bmatrix}. \tag{2.5}$$

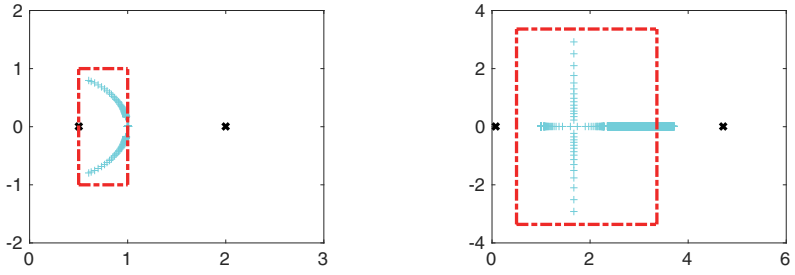
Since $\Pi\mathcal{P}_{ln}\Pi^T$ corresponds to a block lower-triangular preconditioner, we refer to this as the lower-null preconditioner. Indeed, this preconditioner is "psychologically block-lower triangular" in that we can easily identify blocks with which we can solve this system using a substitution method. As well as solves with B_1, B_1^T, and $\tilde{N}$, and a matrix-vector product with A_{11}, we require matrix-vector products with A_{21} and B_2^T.

The following result holds for the eigenvalues.

THEOREM 2.5. *Let $\tilde{N}$ be an invertible approximation to N. Consider the generalized eigenvalue problem $\mathcal{P}_{ln}\boldsymbol{z} = \lambda\mathcal{A}\boldsymbol{z}$. Then $\lambda = 1$ or λ is an eigenvalue of $\tilde{N}^{-1}N$.*

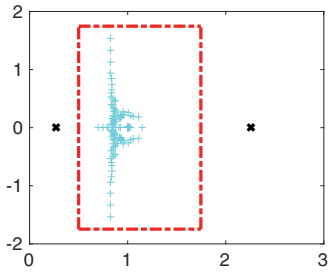
Proof. As in the proof of Theorem 2.1, we use the fact that the required eigenvalues are the same as those of $(\Pi\mathcal{P}_{ln}\Pi^T)^{-1}\Pi\mathcal{A}\Pi^T$. Recalling (1.7), we have that

$$(\Pi\mathcal{P}_{ln}\Pi^T)^{-1}\Pi\mathcal{A}\Pi = \begin{bmatrix} I & \widehat{A}^{-1}\widehat{B} \\ 0 & \tilde{N}^{-1}(A_{22} - \widehat{B}\widehat{A}^{-1}\widehat{B}^T) \end{bmatrix} = \begin{bmatrix} I & \widehat{A}^{-1}\widehat{B} \\ 0 & \tilde{N}^{-1}N \end{bmatrix}.$$

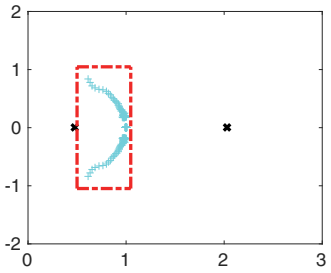


(a) N.

(b) I.

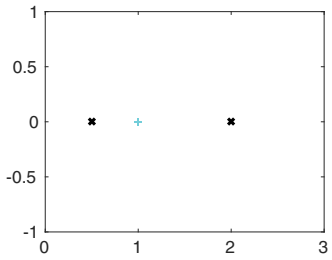


(c) IC (10^{-1}).

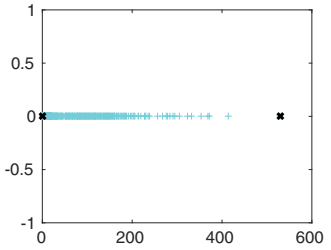


(d) IC (10^{-2}).

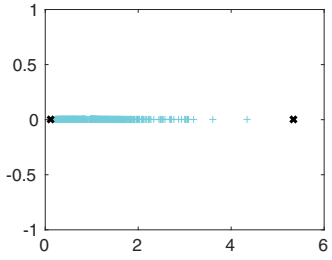
FIG. 2. *MOSARQP1: The eigenvalues (+) of $\mathcal{P}_{cn}^{-1}\mathcal{A}$ and the bounds in Corollaries* 2.2 *and* 2.4 *for different choices of $\widetilde{N}$. The black crosses show the bounds on real eigenvalues, and the red dashed lines show the bounds on complex eigenvalues with nonzero imaginary part.*



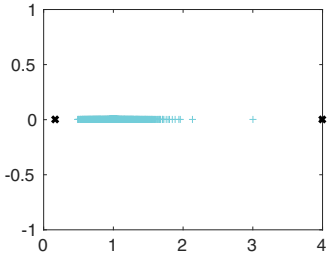
(a) N.



(b) I.



(c) IC (10^{-1}).



(d) IC (10^{-2}).

FIG. 3. *STCQP2: The eigenvalues (+) of $\mathcal{P}_{cn}^{-1}\mathcal{A}$ and the bounds in Corollaries* 2.2 *and* 2.4 *for different choices of $\widetilde{N}$. The black crosses show the bounds on real eigenvalues.*

The result follows. ∎

An apparent drawback of this preconditioner is that $\mathcal{P}_{ln}$ is a nonsymmetric preconditioner for a symmetric problem, and hence we have to use a nonsymmetric iterative method, such as GMRES [48] or BiCGStab [53]. However, Theorem 2.5 shows that if $N = \widetilde{N}$, GMRES applied to (1.1) with preconditioner $\mathcal{P}_{ln}$ converges in two steps. When $\widetilde{N} \neq N$ the eigenvalues may not tell us everything about convergence [24], although it is commonly observed that tightly clustered eigenvalues do predict convergence of GMRES in nonpathological cases—see, e.g., Pestana and Wathen [40] for a discussion. An additional benefit of using $\tilde{N}$ within a preconditioner for the whole matrix $\mathcal{A}$, and not explicitly for the null-space matrix, is that, in principle, all that is needed is (the action of the inverse of) an approximation, $\tilde{N}$. We envisage (as is often the case with standard Schur complement preconditioning; see, e.g., [16]) applications where $\tilde{N}$ can be applied without explicitly performing the (often costly) procedure of forming N. To give two examples, the matrix N is known as the *system flexibility matrix* in structural analysis [28] and the *reduced Hessian matrix* in optimization [37, section 16.5]; in both of these cases, approximations to this matrix, which will fit into this framework, have been proposed in the literature (see, e.g., [8, 19, 41]). Finally, we note that in some cases $B_2 = 0$, as is the case for the CUTEst test matrix STCQP2 discussed at the end of section 2.1. In this case $N = A_{22}$, which is trivial to compute. See [34] for the construction of ideal preconditioners for Schur complement methods.

If we instead drop the $\mathcal{L}$ term from (2.1), we get

$$\mathcal{P}_{un} := \begin{bmatrix} A_{11} & A_{12} & B_1^T \\ 0 & \tilde{N} & 0 \\ B_1 & B_2 & 0 \end{bmatrix}.$$

For reasons analogous to those given above, we refer to this as the upper-null preconditioner, and we have the following result.

THEOREM 2.6. *Let $\tilde{N}$ be an invertible approximation to N. Consider the generalized eigenvalue problem $\mathcal{P}_{un}\boldsymbol{z} = \lambda\mathcal{A}\boldsymbol{z}$. Then $\lambda = 1$ or λ is an eigenvalue of $\tilde{N}^{-1}N$.*

Proof. Since $(\Pi\mathcal{P}_{un}\Pi^T)^{-1}(\Pi\mathcal{A}\Pi^T)$ is similar to $(\Pi\mathcal{A}\Pi^T)(\Pi\mathcal{P}_{un}\Pi^T)^{-1}$, a similar argument to that in the proof of Theorem 2.5 gives the result. ∎

The preconditioner $\mathcal{P}_{un}$ therefore has the same eigenvalues, with the same multiplicity, as $\mathcal{P}_{ln}$. In spite of possible effects of nonnormality, in practice upper and lower block triangular preconditioners often exhibit similar behavior (see [39] and the references therein), and this was our experience in the tests reported in section 4. Additionally, the cost of applying this preconditioner is the same as the cost of applying $\mathcal{P}_{ln}$.

2.3. A constraint preconditioner. Now consider the preconditioner obtained by replacing N by $\tilde{N}$ in (2.1) without dropping any terms, namely,

$$\mathcal{P}_{con} = \begin{bmatrix} A_{11} & 0 & B_1^T \\ A_{21} & \tilde{N} & B_2^T \\ B_1 & 0 & 0 \end{bmatrix} \begin{bmatrix} I & B_1^{-1}B_2 & 0 \\ 0 & I & 0 \\ 0 & B_1^{-T}X^T & I \end{bmatrix}.$$

Note that this preconditioner is numerically the same as the preconditioner defined by the GALAHAD [22] subroutine SBLS. The authors are not aware of an eigenanalysis of this preconditioner in the literature.

THEOREM 2.7. *The preconditioner $\mathcal{P}_{con}$ is a constraint preconditioner, i.e., $\mathcal{P}_{con} = \left[\begin{smallmatrix} G & B^T \\ B & 0 \end{smallmatrix}\right]$ for some matrix G.*

Proof. Direct computation shows that

$$\Pi\mathcal{P}_{con}\Pi^T = \begin{bmatrix} \widehat{A} & \widehat{B}^T \\ \widehat{B} & A_{22} - N + \widetilde{N} \end{bmatrix} \text{ and } \mathcal{P}_{con} = \begin{bmatrix} A_{11} & A_{12} & B_1^T \\ A_{21} & A_{22} - N + \widetilde{N} & B_2^T \\ B_1 & B_2 & 0 \end{bmatrix}. \quad (2.6)$$

∎

We can show the following result about the eigenvalues for the constraint preconditioner here.

THEOREM 2.8. *Let $\widetilde{N}$ be an invertible approximation to N. Consider the generalized eigenvalue problem $\mathcal{A}\boldsymbol{z} = \lambda\mathcal{P}_{con}\boldsymbol{z}$. Then $\lambda = 1$ or λ is an eigenvalue of $\widetilde{N}^{-1}N$.*

Proof. As in previous sections, we can compute the eigenvalues of $\mathcal{P}_{con}^{-1}\mathcal{A}$ by solving a generalized eigenvalue problem to find the eigenvalues of $(\Pi\mathcal{P}_{con}\Pi^T)^{-1}(\Pi\mathcal{A}\Pi^T)$. In particular, using (2.6), we have that

$$\begin{bmatrix} \widehat{A} & \widehat{B}^T \\ \widehat{B} & A_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \lambda \begin{bmatrix} \widehat{A} & \widehat{B}^T \\ \widehat{B} & A_{22} - N + \widetilde{N} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}.$$

The first equation shows that $\lambda = 1$ or that $\widehat{A}\boldsymbol{x} + \widehat{B}^T\boldsymbol{y} = 0$. Since $\widehat{A}$ is invertible, in the latter case we have that $\boldsymbol{x} = -\widehat{A}^{-1}\widehat{B}^T\boldsymbol{y}$.

The second equation gives that $(1-\lambda)\widehat{B}\boldsymbol{x} + (1-\lambda)A_{22}\boldsymbol{y} = \lambda(\widetilde{N} - N)\boldsymbol{y}$. If $\lambda \neq 1$, then after substituting for $\boldsymbol{x}$ we find that

$$-(1-\lambda)\widehat{B}\widehat{A}^{-1}\widehat{B}^T\boldsymbol{y} + (1-\lambda)A_{22}\boldsymbol{y} = \lambda(\widetilde{N} - N)\boldsymbol{y}.$$

Using (1.7) and simplifying shows that $N\boldsymbol{y} = \lambda\widetilde{N}\boldsymbol{y}$ as required. ∎

Comparison with Theorems 2.5 and 2.6 shows that all three preconditioned matrices $\mathcal{P}_{ln}^{-1}\mathcal{A}$, $\mathcal{P}_{un}^{-1}\mathcal{A}$, and $\mathcal{P}_{con}^{-1}\mathcal{A}$ have the same eigenvalues, with the same multiplicities. The constraint preconditioner benefits from certain advantages, as we discuss in sections 2.4 and 3.4 below. However, this comes at a price, as $\mathcal{P}_{con}$ is more expensive to apply than $\mathcal{P}_{ln}$ and $\mathcal{P}_{un}$. To see this, first note that $\mathcal{P}_{con} = \mathcal{P}_{ln}\mathcal{P}_r$, where

$$\mathcal{P}_r = \begin{bmatrix} I & B_1^{-1}B_2 & 0 \\ 0 & I & 0 \\ 0 & B_1^{-T}X^T & I \end{bmatrix}.$$

The additional cost of solving a system with $\mathcal{P}_{con}$, rather than $\mathcal{P}_{ln}$ or $\mathcal{P}_{un}$, is thus one solve with each of B_1 and B_1^T and a matrix-vector multiplication with each of B_2, A_{11}, and A_{12}.

2.4. Discussion. The preconditioners described above are the result of thinking about the null-space method in terms of a matrix factorization. The preconditioners $\mathcal{P}_{ln}$ and $\mathcal{P}_{un}$ are particularly promising. They have the drawback that applying them requires solves with B_1 and B_1^T, as well as the solve with $\tilde{N}$ and a number of matrix-vector multiplications. It is also somewhat jarring that we are proposing nonsymmetric preconditioners for a symmetric problem (although a short-term recurrence method in a nonstandard inner product can be applied as discussed in section 3.3). Balancing these issues is the fact that the eigenvalue clustering is as good as possible.

The central-null preconditioner is indefinite, which means that short-term recurrence methods cannot be straightforwardly applied. Additionally, the spectrum of the preconditioned matrix depends on the eigenvalues of $N^{-1}A_{22}$ (or $\widetilde{N}^{-1}A_{22}$), and we shall see in section 4 that $\mathcal{P}_{cn}$ is most useful when these eigenvalues are tightly clustered.

Constraint preconditioners possess favorable properties, such as the iterates staying on a certain manifold [47], which can be useful in certain applications. If such properties are desired, then $\mathcal{P}_{con}$ provides them, even with an approximate $\tilde{N}$, at the expense of extra linear system solves with the matrices B_1 and B_1^T. This is in contrast to the equivalent Schur complement formulation [7], which gives an *inexact* constraint preconditioner. As such, null-space preconditioners could be useful in optimization, where solution methods that remain on the constraint manifold are often required; see, e.g., [1]. Additionally, it is possible to use a projected CG or MINRES method in this case, as described in section 3.4. We envisage that this preconditioner will be particularly useful in fields where many systems have to be solved with the same B block, possibly with A changing; an important example that generates linear systems with this structure is the interior point method in optimization [54].

Null-space preconditioners require us to find an invertible subset of the constraint matrix B, which is an additional computational cost that is not present in, for instance, Schur complement based approaches. However, there are a number of applications we are aware of where this is not problematic; we discuss a few of these in more detail in section 4.

We note that for problems with maximally rank-deficient A, i.e., problems for which $\operatorname{rank}(A) = n - m$, alternative block diagonal preconditioners were recently proposed by Estrin and Greif [17] that rely on a matrix C whose columns span the null-space of A. Estrin and Greif show that under certain conditions, the preconditioned systems can be solved by (standard) conjugate gradients; otherwise a standard nonsymmetric Krylov method can be used.

3. Using conjugate gradients and MINRES with the null-space preconditioners. As discussed in section 2.4, although $\mathcal{P}_{ln}^{-1}\mathcal{A}$, $\mathcal{P}_{un}^{-1}\mathcal{A}$, and $\mathcal{P}_{con}^{-1}\mathcal{A}$ have nice spectra when $\widetilde{N}$ is a good approximation of N, the preconditioners are not symmetric positive definite. Accordingly, they cannot be used with standard MINRES or CG. However, since $\mathcal{P}_{con}$ is a constraint preconditioner, it can be used with the projected CG [20] or projected MINRES [21] methods. On the other hand, although $\mathcal{P}_{ln}$ is nonsymmetric, it can be used in conjunction with CG in a nonstandard inner product. We discuss both these approaches in this section.

3.1. Nonstandard inner products. In this section, we show that it is possible to use CG in a nonstandard inner product to solve (1.1) with the preconditioner (2.5). We consider general equations of the form

$$\underbrace{\begin{bmatrix} A & B^T \\ B & C \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix}, \tag{3.1}$$

where we assume that $\mathcal{A}$ is invertible, $A \in \mathbb{R}^{n\times n}$ is symmetric and invertible, $B \in \mathbb{R}^{m\times n}$, $m \le n$, and $C \in \mathbb{R}^{m\times m}$ is symmetric. We additionally assume that the Schur complement $C - BA^{-1}B^T$ is positive definite, although our results extend trivially to the case where $C - BA^{-1}B^T$ is negative definite. Note that the proceeding results hold for $A = \widehat{A}$, $B = \widehat{B}$, and $C = A_{22}$ in (1.4), as we show in section 3.3, but are more generally applicable.

Although the saddle point matrix $\mathcal{A}$ is indefinite, so that standard CG cannot be reliably applied to solve (1.1), a judicious choice of preconditioner can make $\mathcal{A}$ self-adjoint and positive definite with respect to a nonstandard inner product. A number of preconditioners that achieve this goal have been proposed [6, 10, 14, 15, 33, 51]. Many, although not all, fall into the class of preconditioners and inner products discussed by Krzyżanowski [30], who showed the following.

PROPOSITION 3.1 (Krzyżanowski [30], Proposition 2.1). *Suppose we wish to solve the system* (3.1). *Consider the preconditioner given by*

$$\mathcal{P}^{-1} = \begin{bmatrix} I & -dA_0^{-1}B^T \\ 0 & I \end{bmatrix} \begin{bmatrix} A_0^{-1} & 0 \\ 0 & S_0^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -cBA_0^{-1} & I \end{bmatrix}, \tag{3.2}$$

for fixed scalars c, d, and where A_0 and S_0 are symmetric and nonsingular. Let $\delta \in \{-1, +1\}$ and $\mathcal{H}$ be the block diagonal matrix

$$\mathcal{H} = \delta \begin{bmatrix} A_0 - cA & 0 \\ 0 & S_0 + cdBA_0^{-1}B^T - dC \end{bmatrix}. \tag{3.3}$$

Then $\mathcal{H}\mathcal{P}^{-1}\mathcal{A}$ is symmetric.

This means that, even though $\mathcal{P}^{-1}\mathcal{A}$ is nonsymmetric, it is self-adjoint with respect to the bilinear form $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, where $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{H}} = \boldsymbol{y}^T \mathcal{H} \boldsymbol{x}$. If A_0, S_0, and δ are chosen so that $\mathcal{H}$ is symmetric positive definite, and inner products are understood to be $\mathcal{H}$-inner products, then we can apply CG to solve (1.1). Algorithm 1, adapted from Algorithm 3.2 of Dollar et al. [15], is one such method which does this.

Given $\boldsymbol{x}_0$, set $\boldsymbol{r}_0 = \boldsymbol{b} - \mathcal{A}\boldsymbol{x}_0$, $\boldsymbol{z}_0 = \mathcal{P}^{-1}\boldsymbol{r}_0$, and $\boldsymbol{p}_0 = \boldsymbol{z}_0$;
for $k = 0, 1, \ldots$ **do**
  $\alpha = \frac{\boldsymbol{z}_k^T \mathcal{H} \boldsymbol{z}_k}{\boldsymbol{p}_k^T \mathcal{H} \mathcal{P}^{-1} \mathcal{A} \boldsymbol{p}_k}$;
  $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha \boldsymbol{p}_k$;
  $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha \mathcal{A} \boldsymbol{p}_k$;
  $\boldsymbol{z}_{k+1} = \mathcal{P}^{-1} \boldsymbol{r}_{k+1}$;
  $\beta = \frac{\boldsymbol{z}_{k+1}^T \mathcal{H} \boldsymbol{z}_{k+1}}{\boldsymbol{z}_k^T \mathcal{H} \boldsymbol{z}_k}$;
  $\boldsymbol{p}_{k+1} = \boldsymbol{z}_{k+1} + \beta \boldsymbol{p}_k$;
end

Algorithm 1: CG in the scalar product defined by $\mathcal{H}$.

3.2. Semidefinite $\mathcal{H}$. Although $\mathcal{P}_{ln}$ is not block lower triangular, $\Pi \mathcal{P}_{ln} \Pi^T$ is, while $\widehat{\mathcal{A}} = \Pi \mathcal{A} \Pi^T$ is a generalized saddle point matrix of the form in (3.1). Accordingly, we can apply the results of Krzyżanowski to this permuted matrix and preconditioner. Since the (1,1) block of $\Pi \mathcal{P}_{ln} \Pi^T$ is identical to that of $\widehat{\mathcal{A}}$, we are in the situation in which $c = 1$, $d = 0$, and $A_0 = A$ in (3.2). This may indicate that $\mathcal{P}_{ln}$ is a good preconditioner, but it also means that $\mathcal{H}$ in (3.3) is singular. Despite this, a more careful examination of Algorithm 1 will reveal that this singularity causes no difficulties for computing the solution of (3.1).

Accordingly, we will consider the case $A_0 = A$, $c = 1$, and $d = 0$ in (3.2) and (3.3) in more detail, which arises in our application, but also more widely when solving, e.g., problems in PDE constrained optimization [42, 45]. We assume that S_0 is symmetric

positive definite, giving

$$\mathcal{H}_{1,0} = \begin{bmatrix} 0 & 0 \\ 0 & S_0 \end{bmatrix}, \quad \mathcal{P}_{1,0} = \begin{bmatrix} A & 0 \\ B & S_0 \end{bmatrix}. \tag{3.4}$$

Thus, the matrix $\mathcal{H}_{1,0}$ is semidefinite with rank m.

Letting each vector $\boldsymbol{v}$ in Algorithm 1 be decomposed as $\boldsymbol{v} = \left[(\boldsymbol{v}^{(1)})^T \ (\boldsymbol{v}^{(2)})^T\right]^T$, $\boldsymbol{v}^{(1)} \in \mathbb{R}^n$, it is straightforward to show that Algorithm 1 is equivalent to Algorithm 2. However, it is not yet clear that the iterates generated by Algorithms 1 and 2 converge to the solution of (3.1). To show that Algorithm 2 does indeed solve (3.1), we compare Algorithm 2 to preconditioned CG applied to the system obtained from the range-space method, i.e., the method based on a factorization like (1.9).

Given
$\boldsymbol{x}_0 = \begin{bmatrix} \boldsymbol{x}_0^{(1)} \\ \boldsymbol{x}_0^{(2)} \end{bmatrix}$, set $\boldsymbol{z}_0 = \begin{bmatrix} A^{-1}(\boldsymbol{c} - B^T\boldsymbol{x}_0^{(2)}) - \boldsymbol{x}_0^{(1)} \\ S_0^{-1}(\boldsymbol{d} - BA^{-1}\boldsymbol{c} - (C - BA^{-1}B^T)\boldsymbol{x}_0^{(2)}) \end{bmatrix}$ and $\boldsymbol{p}_0 = \boldsymbol{z}_0$;
for $k = 0, 1, \ldots$ **do**
$\alpha = \frac{(\boldsymbol{z}_k^{(2)})^T S_0 (\boldsymbol{z}_k^{(2)})}{(\boldsymbol{p}_k^{(2)})^T (C - BA^{-1}B^T)\boldsymbol{p}_k^{(2)}}$;
$\begin{bmatrix} \boldsymbol{x}_{k+1}^{(1)} \\ \boldsymbol{x}_{k+1}^{(2)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_k^{(1)} + \alpha \boldsymbol{p}_k^{(1)} \\ \boldsymbol{x}_k^{(2)} + \alpha \boldsymbol{p}_k^{(2)} \end{bmatrix}$;
$\begin{bmatrix} \boldsymbol{z}_{k+1}^{(1)} \\ \boldsymbol{z}_{k+1}^{(2)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_k^{(1)} - \alpha(\boldsymbol{p}_k^{(1)} + A^{-1}B^T\boldsymbol{p}_k^{(2)}) \\ \boldsymbol{z}_k^{(2)} - \alpha S_0^{-1}(C - BA^{-1}B^T)\boldsymbol{p}_k^{(2)} \end{bmatrix}$;
$\beta = \frac{(\boldsymbol{z}_{k+1}^{(2)})^T S_0 (\boldsymbol{z}_{k+1}^{(2)})}{(\boldsymbol{z}_k^{(2)})^T S_0 (\boldsymbol{z}_k^{(2)})}$;
$\begin{bmatrix} \boldsymbol{p}_{k+1}^{(1)} \\ \boldsymbol{p}_{k+1}^{(2)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_{k+1}^{(1)} + \beta \boldsymbol{p}_k^{(1)} \\ \boldsymbol{z}_{k+1}^{(2)} + \beta \boldsymbol{p}_k^{(2)} \end{bmatrix}$;
end

Algorithm 2: Simplified CG in the scalar product defined by $\mathcal{H}_{1,0}$.

The range-space method—which is equivalent to solving (3.1)—proceeds in two stages; first we seek a solution to

$$(C - BA^{-1}B^T)\boldsymbol{v} = \boldsymbol{d} - BA^{-1}\boldsymbol{c} \tag{3.5}$$

before recovering $\boldsymbol{u}$ by solving

$$\boldsymbol{u} = A^{-1}(\boldsymbol{c} - B^T\boldsymbol{v}). \tag{3.6}$$

More details can be found in, e.g., [4, Chapter 5].

Since $C - BA^{-1}B^T$ is positive definite, we can solve (3.5) by preconditioned CG with a symmetric positive definite preconditioner S_0 as in Algorithm 3. Note that when applied to (1.2), our Schur complement approach is actually a null-space method, as we show in section 3.3 (cf. Algorithm 2.1 in Gould, Hribar, and Nocedal [20]).

Comparison of Algorithms 2 and 3 shows that whenever $\boldsymbol{x}_0^{(2)} = \boldsymbol{v}_0$, the vectors $\boldsymbol{z}_0^{(2)}$ and $\boldsymbol{p}_0^{(2)}$ in Algorithm 2 are the same as $\boldsymbol{z}_0$ and $\boldsymbol{p}_0$ in Algorithm 3. Moreover, since the scalars α and β in the two algorithms are equivalent, $\boldsymbol{x}_k^{(2)}$, $\boldsymbol{z}_k^{(2)}$, and $\boldsymbol{p}_k^{(2)}$ in Algorithm 2 are the same as $\boldsymbol{v}_k$, $\boldsymbol{z}_k$, and $\boldsymbol{p}_k$ in Algorithm 3 for all iterations $k \geq 0$. It follows from the convergence theory for Algorithm 3 that α, β, $\boldsymbol{x}_k^{(2)}$, $\boldsymbol{z}_k^{(2)}$, and $\boldsymbol{p}_k^{(2)}$ are all well defined and that the iterates $\boldsymbol{x}_k^{(2)}$ in Algorithm 2 are approximations of $\boldsymbol{v}$. However, Algorithm 2 also yields approximations of $\boldsymbol{u}$, as the next result shows.

Given $\boldsymbol{v}_0$, set $\boldsymbol{z}_0 = S_0^{-1}(\boldsymbol{d} - BA^{-1}\boldsymbol{c} - (C - BA^{-1}B^T)\boldsymbol{v}_0)$ and $\boldsymbol{p}_0 = \boldsymbol{z}_0$;
for $k = 0, 1, \ldots$ **do**
  $\alpha = \frac{\boldsymbol{z}_k^T S_0 \boldsymbol{z}_k}{\boldsymbol{p}_k^T (C - BA^{-1}B^T)\boldsymbol{p}_k}$;
  $\boldsymbol{v}_{k+1} = \boldsymbol{v}_k + \alpha \boldsymbol{p}_k$;
  $\boldsymbol{z}_{k+1} = \boldsymbol{z}_k - \alpha S_0^{-1}(C - BA^{-1}B^T)\boldsymbol{p}_k$;
  $\beta = \frac{\boldsymbol{z}_{k+1}^T S_0 \boldsymbol{z}_{k+1}}{\boldsymbol{z}_k^T S_0 \boldsymbol{z}_k}$;
  $\boldsymbol{p}_{k+1} = \boldsymbol{z}_{k+1} + \beta \boldsymbol{p}_k$;
end

Algorithm 3: CG for the reduced system (3.5).

LEMMA 3.2. *Let* $\boldsymbol{x}_k^{(1)}$, $\boldsymbol{x}_k^{(2)}$, *and* $\boldsymbol{z}_k^{(1)}$ *be as in Algorithm* 2, $k \geq 0$. *Additionally, let*

$$\boldsymbol{u}_k = A^{-1}(\boldsymbol{c} - B^T \boldsymbol{x}_k^{(2)}). \tag{3.7}$$

Then $\boldsymbol{u}_k = \boldsymbol{x}_k^{(1)} + \boldsymbol{z}_k^{(1)}$.

Proof. We show that $\boldsymbol{z}_k^{(1)} = \boldsymbol{u}_k - \boldsymbol{x}_k^{(1)}$ by induction. First, since $\boldsymbol{z}_0^{(1)} = A^{-1}(\boldsymbol{c} - B^T \boldsymbol{x}_0^{(2)}) - \boldsymbol{x}_0^{(1)}$, we see from (3.7) that $\boldsymbol{z}_0^{(1)} = \boldsymbol{u}_0 - \boldsymbol{x}_0^{(1)}$. Now assume that for some $j \geq 0$, $\boldsymbol{z}_j^{(1)} = \boldsymbol{u}_j - \boldsymbol{x}_j^{(1)}$. From the updates for $\boldsymbol{x}_{j+1}^{(1)}$ and $\boldsymbol{x}_{j+1}^{(2)}$ in Algorithm 2 we see that $\alpha \boldsymbol{p}_j^{(1)} = \boldsymbol{x}_{j+1}^{(1)} - \boldsymbol{x}_j^{(1)}$ and $\alpha \boldsymbol{p}_j^{(2)} = \boldsymbol{x}_{j+1}^{(2)} - \boldsymbol{x}_j^{(2)}$. Substituting these formulae into the equation for $\boldsymbol{z}_{j+1}^{(1)}$ and using (3.7) gives that $\boldsymbol{z}_{j+1}^{(1)} = \boldsymbol{z}_j^{(1)} + (\boldsymbol{x}_j^{(1)} - \boldsymbol{x}_{j+1}^{(1)}) + (\boldsymbol{u}_{j+1} - \boldsymbol{u}_j)$. This shows that whenever $\boldsymbol{z}_j^{(1)} = \boldsymbol{u}_j - \boldsymbol{x}_j^{(1)}$ holds, $\boldsymbol{z}_{j+1}^{(1)} = \boldsymbol{u}_{j+1} - \boldsymbol{x}_{j+1}^{(1)}$. Since $\boldsymbol{z}_0 = \boldsymbol{u}_0 + \boldsymbol{x}_0^{(1)}$, the stated result is proved. ∎

Now, because $\boldsymbol{x}_k^{(2)}$ approximates $\boldsymbol{v}$, the vector $\boldsymbol{x}_k^{(1)} + \boldsymbol{z}_k^{(1)}$ approximates $\boldsymbol{u}$, and so we obtain approximations of both $\boldsymbol{u}$ and $\boldsymbol{v}$ from Algorithm 2. Thus, Algorithm 2 is well defined and can be used to solve (3.1).

As a final point, since to solve (3.1) we only need $\boldsymbol{u}_k = \boldsymbol{x}_k^{(1)} + \boldsymbol{z}_k^{(1)}$ and $\boldsymbol{p}_k$, it is straightforward to show that nothing is lost in Algorithm 2 by setting $\boldsymbol{p}_k^{(1)} = \boldsymbol{0}$, $k > 0$, and that some computational savings are made by avoiding the vector update. Additionally, in our experience such a step can be useful for reducing the effect of rounding errors.

3.3. A nonstandard conjugate gradient method for $\mathcal{P}_{ln}$. Now let us apply the results of this section to our system (1.1) with preconditioner (2.5).

Letting

$$\Pi \mathcal{H}_{\ln} \Pi^T = \begin{bmatrix} 0 & \\ & \widetilde{N} \end{bmatrix},$$

and recalling (1.4) and (2.5), we see that $\Pi \mathcal{H}_{\ln} \Pi^T$, $\Pi \mathcal{P}_{ln} \Pi^T$, and $\Pi \mathcal{A} \Pi^T$ are in the form of $\mathcal{H}_{1,0}$ and $\mathcal{P}_{1,0}$ in (3.4) and $\mathcal{A}$ in (3.1). Accordingly, we can apply nonstandard CG in Algorithm 2 to

$$\Pi(\mathcal{H}_{\ln} \mathcal{P}_{ln}^T \mathcal{A}) \Pi^T (\Pi \boldsymbol{w}) = \Pi \mathcal{H}_{\ln} \mathcal{P}_{ln} \boldsymbol{b}, \tag{3.8}$$

where $\boldsymbol{w} = \begin{bmatrix} \boldsymbol{x}_1^T, & \boldsymbol{x}_2^T, & \boldsymbol{y}^T \end{bmatrix}^T$ and $\boldsymbol{b} = \begin{bmatrix} \boldsymbol{f}_1, & \boldsymbol{f}_2, & \boldsymbol{g} \end{bmatrix}^T$, with $\boldsymbol{x} = [\boldsymbol{x}_1^T, \ \boldsymbol{x}_2^T]^T$ and $\boldsymbol{f} = [\boldsymbol{f}_1^T, \ \boldsymbol{f}_2^T]^T$ and $\boldsymbol{x}_1, \boldsymbol{f}_1 \in \mathbb{R}^m$. As shown in the previous section, this is equivalent to

applying preconditioned CG to the reduced system (3.5), given by $N\boldsymbol{v} = \boldsymbol{d} - \widehat{B}\widehat{A}^{-1}\boldsymbol{c}$ where, because of the permutation matrix Π, $\boldsymbol{v} = \boldsymbol{x}_2$, $\boldsymbol{d} = \boldsymbol{f}_2$, and $\boldsymbol{c} = [\boldsymbol{f}_1^T,\ \boldsymbol{g}^T]^T$. The vector $\boldsymbol{u}$ in (3.6) is $\boldsymbol{u} = [\boldsymbol{x}_1^T,\ \boldsymbol{y}^T]^T$.

Comparison with the null-space method shows that solving this reduced system is the same as using the null-space method with the fundamental basis Z_f and the particular solution (1.8). It follows that applying nonstandard CG to (3.8) is equivalent to applying the null-space method using the fundamental basis (1.3).

3.4. The constraint preconditioner, $\mathcal{P}_{cn}$. Finally, for this section, we mention the preconditioner $\mathcal{P}_{cn}$. As this is a constraint preconditioner, we can apply it with projected conjugate gradients [20], provided that the (1,1) block is symmetric positive definite on the nullspace of B. If A is indefinite, or if we require a method that minimizes the residual, then we can still utilize a short-term recurrence method by using projected MINRES [21]. Therefore, standard methods work *out-of-the-box*, and no extra theory is needed in this case.

4. Numerical results. In this section we apply null-space preconditioners to matrices arising in applications, each chosen to highlight a specific feature of the proposed preconditioners. We compare their behavior with the Schur complement based preconditioners

$$\mathcal{P}_{\text{ls}} := \begin{bmatrix} A & 0 \\ B & -S_0 \end{bmatrix},\ \mathcal{P}_{\text{cs}} := \begin{bmatrix} A & 0 \\ 0 & S_0 \end{bmatrix},\ \mathcal{P}_{\text{cons}} := \begin{bmatrix} A & B^T \\ B & BA^{-1}B^T - S_0 \end{bmatrix}, \tag{4.1}$$

and we use a similar approximation to the Schur complement and the null-space matrix in each case. We also tested $\mathcal{P}_{\text{us}}$ and $\mathcal{P}_{\text{un}}$, the transposes of $\mathcal{P}_{\text{ls}}$ and $\mathcal{P}_{\text{ln}}$, but the results were broadly the same as for $\mathcal{P}_{\text{ls}}$ and $\mathcal{P}_{\text{ln}}$.

To apply $\mathcal{P}_{\text{ls}}$ we must solve a system with A and another with S_0, while $\mathcal{P}_{\text{ln}}$ additionally requires a matrix-vector product with B. Applying $\mathcal{P}_{\text{con}}$ requires the same operations as for $\mathcal{P}_{\text{ln}}$, as well as an extra matrix-vector product with B^T and a solve with A.

Unless otherwise stated, we apply right-preconditioned GMRES or nonstandard CG, which we terminate when the relative residual satisfies $\|\boldsymbol{r}_k\|_2/\|\boldsymbol{r}_0\|_2 < 10^{-8}$, or when $\min\{n+m, maxit\}$ iterations are reached, where $maxit$ is specified for each example below.

For some problems we approximate N and S by incomplete Cholesky factorizations. These we compute by the MATLAB routine `ichol` with a default drop tolerance of 10^{-2}, which seemed suitable for most of our problems. If `ichol` fails, we reduce the drop tolerance by a factor of 10 until a factorization is computed. The smallest drop tolerance used is 10^{-8}. We point out that this factorization is rather costly to compute, as it first involves forming N or S exactly. Moreover, the cost of the incomplete Cholesky factorization depends on the size of N (or S), its sparsity pattern, and the drop tolerance used; here we use larger drop tolerances where possible to reduce this cost. However, these incomplete Cholesky preconditioners may be cheaper to apply than N and S, and they may provide information about what to expect for reasonable approximations to N and S.

4.1. Random saddle point matrices.

Example 4.1. Consider the pseudorandom sparse matrix generated by the following MATLAB code:

```
A = sprandsym(n,0.1,1e-2,1);
B = sprand(m,n,0.5);
```

```
K = [A B'; B sparse(m,m)];
```
We take `n = 100` and `m = 10, 50, 90`.

For these problems $maxit = 200$.

In this example we test values of $n - m$ that are small, moderate, and large in comparison with n. We compare the null-space preconditioners $\mathcal{P}_{cn}$ and $\mathcal{P}_{un}$, and corresponding Schur complement preconditioners in (4.1). For our approximations to the Schur complement and the null-space matrix we take the identity matrix of the appropriate dimension. We apply these preconditioners using the MATLAB inbuilt `GMRES` routine and report the results in Figure 4. Note that the null-space preconditioners $\mathcal{P}_{un}$ and $\mathcal{P}_{con}$ give broadly the same behavior as $\mathcal{P}_{ln}$, as do $\mathcal{P}_{\text{us}}$ and $\mathcal{P}_{\text{cons}}$ when compared with $\mathcal{P}_{\text{ls}}$, and so for clarity those results are omitted from Figure 4.

Here, by choosing a weak approximation of the Schur complement and the null-space matrix (namely, the identity matrix), convergence depends entirely on how important this component piece is to the overall approximation. Therefore, the null-space based preconditioners do well for small $n - m$, the Schur complement based preconditioners do well for $n - m$ close to n, and there is no clear winner in the intermediate case when $n = m/2$. This suggests that null-space preconditioners may be a better choice over Schur complement preconditioners if we have an application where $n - m$ is small, particularly if we do not have a good Schur complement approximation. We see that for both the Schur complement and null-space preconditioners, the central approximations take roughly twice the number of iterations but are cheaper to apply. This phenomenon has been noted by others; see, e.g., [18].

Example 4.2. Consider now the sparse matrix generated by the MATLAB code

```
A = 10*speye(n,n);
B = sprand(m,n,0.5);
K = [A B'; B sparse(m,m)];
```
We take `n = 100` and `m = 10, 50, 90`.

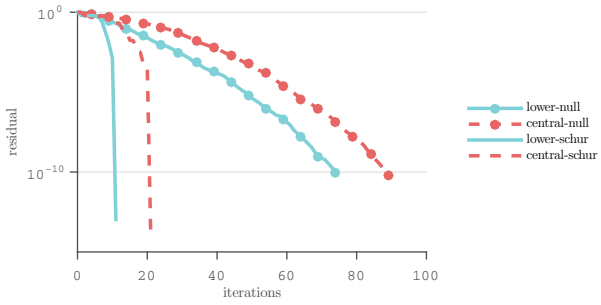
In Example 4.2 we have the same B as in Example 4.1, but instead we take a scaled identity matrix for A. We test the same preconditioners that we used in Example 4.1, namely using an identity matrix as the Schur complement/null space approximation. We give the results in Figure 5. Again, $maxit = 200$, and since $\mathcal{P}_{un}$, (resp., $\mathcal{P}_{\text{us}}$) differs only slightly from $\mathcal{P}_{ln}$ and $\mathcal{P}_{con}$ (resp., $\mathcal{P}_{\text{us}}$ and $\mathcal{P}_{\text{cons}}$), we report only the former.

Here, in contrast to Example 4.1, the null-space preconditioners perform well for both large and small values of $n - m$. This is because the (1,1) block in $\mathcal{A}$ is a scaled identity matrix here, and, as a consequence of Theorem 2.1, the eigenvalues of $\mathcal{P}_{cn}^{-1}\mathcal{A}$ are well clustered. Again, the pattern in which the central-based preconditioners take about twice the iterations of the others is in evidence.

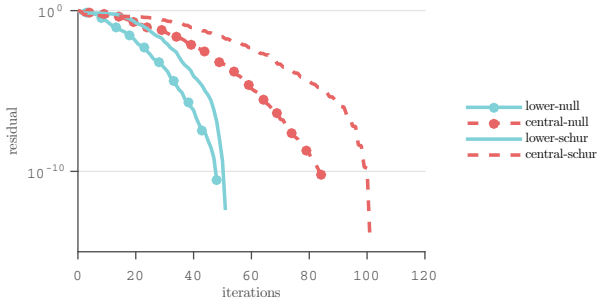
4.2. Optimization and interior point methods. Here we consider quadratic programming problems of the form

$$\begin{aligned} \min \ & \frac{1}{2}\boldsymbol{x}^T H\boldsymbol{x} + \boldsymbol{f}^T\boldsymbol{x} \\ \text{s.t. } & B\boldsymbol{x} = \boldsymbol{g}, \\ & \boldsymbol{x} \geq \underline{0}. \end{aligned} \tag{4.2}$$

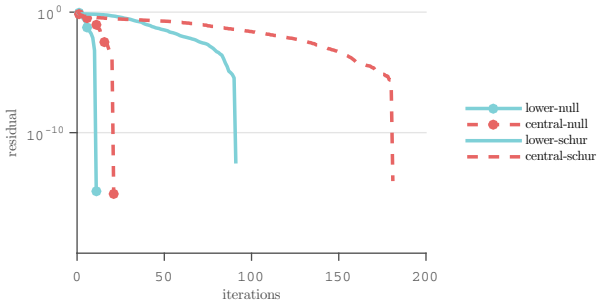
If we solve such a problem using a primal-dual interior point method [54], then at iteration k of the optimization algorithm we must solve a system of the form (1.1), where $A = H + X_k^{-1}Z_k$ for diagonal matrices X_k, Z_k.



(a) $m = 10$.



(b) $m = 50$.

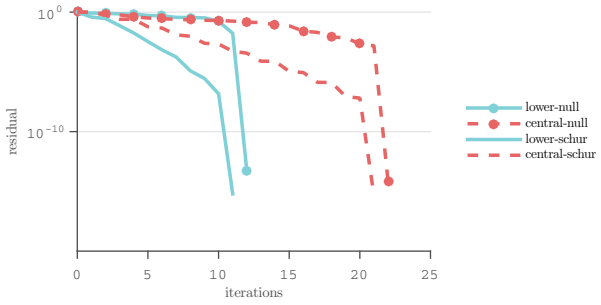


(c) $m = 90$.

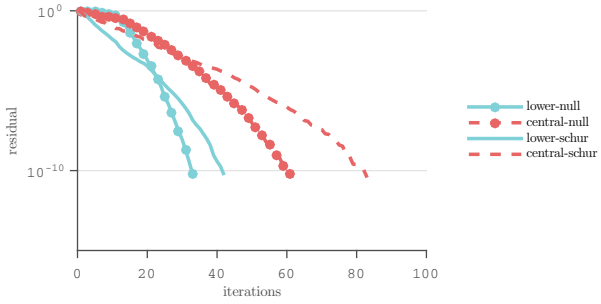
FIG. 4. *Comparison: Schur complement and null-space preconditioners, pseudorandom example from Example* 4.1. *Note that upper-null and constraint-null (and the respective Schur complement preconditioners) give essentially the same behavior as lower-null or upper-Schur (as appropriate), and so are omitted.*

In this context it is common to solve the linear system (1.1) by reducing it to the null-space matrix N, which the optimization community refers to as the reduced Hessian [8], [37, section 16.2]. Since forming the matrix N is expensive, it is common in optimization to approximate this, e.g., by omitting cross terms [12, 36].

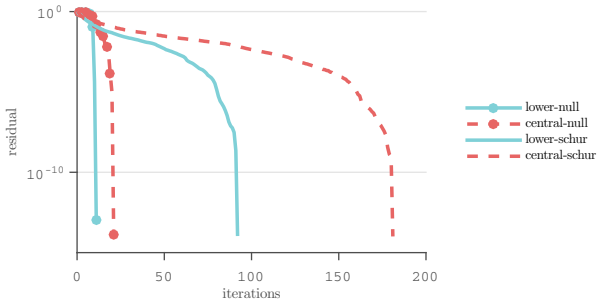
In this setting we need to solve a sequence of linear systems as the interior point method converges, but the "constraint" blocks B do not change. Therefore, we may justify the cost of using a direct method such as LUSOL [49], say, to find a basis of B, since we can reuse this basis over all interior point iterations. Although we do not explore the possibility here, it is also possible to use the interior point method itself



(a) $m = 10$.



(b) $m = 50$.



(c) $m = 90$.

FIG. 5. *Comparison: Schur complement and null-space preconditioners, pseudorandom example from Example* 4.2. *Note that upper-null and constraint-null (and the respective Schur complement preconditioners) give essentially the same behavior as lower-null or upper-Schur (as appropriate), and so are omitted.*

to predict an invertible subblock B_1—see, e.g., Al-Jeiroudi, Gondzio, and Hall [2]. Note that, by using LUSOL to find a suitable B_1, we can subsequently solve with B_1 without further factorization.

In these experiments, we solve for A_{11} in the null-space preconditioner case and A in the Schur complement preconditioner case using a direct method (`backslash`). In many of the examples here these matrices are diagonal, and so such a solve is not a problem. If A is not diagonal, and a factorization is required, we note that this must only be done once in the interior point method, and the factorization can then be updated using, for example, CHOLMOD [11].

TABLE 2
Problem sizes, CUTEst *set matrices from Example* 4.2.

Matrix	n	m	Matrix	n	m
AUG3DC	3873	1000	HUESTIS	10000	2
AUG3DCQP	3873	1000	LASER	1002	1000
CONT-050	2597	2401	LISWET1	10002	10000
CONT-100	10197	9801	MOSARQP1	2500	700
CONT-101	10197	10098	MOSARQP2	900	600
CONT-200	40397	39601	PRIMAL1	325	85
CVXQP3_S	100	75	QPCSTAIR	467	356
DTOC3	14999	9998	STCQP2	4097	2052
GOULDQP3	699	349	YAO	2002	2000
HUES-MOD	10000	2			

To give an idea of how we can expect such methods to work, we run through some problems from the CUTEst test set [23], comparing standard Schur complement preconditioners and null-space preconditioners. These problems, and their dimensions, are listed in Table 2. We highlight that, as described in section 2.4, in optimization it is often important that the inexact solution of this subproblem remains on the constraint manifold. This is a property afforded by constraint preconditioners, and the only true constraint preconditioner tested here is $\mathcal{P}_{con}$.

In our experiments we choose X_k and Z_k so that $X_k^{-1}Z_k = I$, the identity matrix of dimension n; a system of this form may be used in practice to find an initial guess for the interior point method. Additionally, we set $maxit = 1000$.

Our first tests are for the ideal case where we take the exact matrices S or N (see Table 3); these are not practical but give an idea of the best we can expect from the respective method in practice. The fast convergence rates for both the Schur complement and null-space preconditioners, with the exception of the central-null preconditioner, are to be expected from theoretical spectral results (see Theorems 2.5, 2.6, and 2.8 for the null-space preconditioners and [29, 35] for the Schur complement preconditioners). Note that the performance of the central-null preconditioner, in contrast to the other null-space and Schur complement preconditioners, depends on the eigenvalues of $N^{-1}A_{22}$ (see Theorem 2.1), which are not necessarily clustered. We find that iteration counts are slightly higher for CONT-101 and CONT-200 than the theory predicts. However, for these matrices N and S are quite ill-conditioned.

In a further test we consider the simplest approximation to the matrices S and N, namely the identity matrix of appropriate size (see Table 4). Since the identity is generally a poor approximation of S and N, we find that, similarly to Example 4.1, the size of $n-m$ relative to m largely determines whether the null-space based preconditioners are more effective than the Schur complement based preconditioners. When m is large, the Schur complement preconditioned iterative methods perform poorly, as might be expected, and do not always converge to the desired tolerance within 1000 iterations. On the other hand, the null-space preconditioners can perform badly when $n - m$ is large, although these preconditioners do seem somewhat more robust.

Both the central-null and central-Schur preconditioners tend to require twice as many iterations as the other preconditioners. The constraint preconditioner $\mathcal{P}_{con}$ seems to be generally less effective than $\mathcal{P}_{cn}$ when the approximation to the null-space is poor. However, it does have the benefit of being an exact constraint preconditioner. The nonstandard inner product CG can also require more iterations than using the lower-, upper-, or constraint-preconditioners with GMRES, but requires only short-term recurrences.

TABLE 3

Iteration counts for the Schur complement preconditioners with $S_0 = S$ and the null-space preconditioners with $N_0 = N$ for the CUTEst *set matrices from Example* 4.2. ** stands for "did not converge after* 1000 *iterations."*

Matrix	$\mathcal{P}_{ls}$	$\mathcal{P}_{cs}$	$\mathcal{P}_{cons}$	$\mathcal{P}_{ls}$ (NSCG)	$\mathcal{P}_{ln}$	$\mathcal{P}_{cn}$	$\mathcal{P}_{conn}$	$\mathcal{P}_{ln}$ (NSCG)
AUG3DC	2	3	1	1	2	27	1	1
AUG3DCQP	2	3	1	1	2	27	1	1
CONT-050	2	3	1	1	2	20	1	1
CONT-100	2	3	1	1	2	23	1	1
CONT-101	4	5	1	1	4	30	2	*
CONT-200	4	5	1	1	8	25	1	1
CVXQP3_S	2	2	1	1	2	34	1	1
DTOC3	2	2	1	1	2	8	1	1
GOULDQP3	2	3	1	1	2	27	1	1
HUES-MOD	2	2	1	1	2	4	1	1
HUESTIS	2	2	1	1	3	4	2	2
LASER	2	2	1	1	2	3	1	1
LISWET1	4	3	1	2	2	4	1	1
MOSARQP1	2	3	1	1	2	21	1	1
MOSARQP2	2	3	1	1	2	19	1	1
PRIMAL1	2	2	1	1	2	22	1	1
QPCSTAIR	2	2	1	1	2	31	1	1
STCQP2	2	2	1	1	2	3	1	1
YAO	3	3	1	2	2	5	1	1

TABLE 4

Iteration counts for the Schur complement preconditioners with $S_0 = I_n$ and the null-space preconditioners with $N_0 = I_{n-m}$ for the CUTEst *set matrices from Example* 4.2. ** stands for "did not converge after* 1000 *iterations."*

Matrix	$\mathcal{P}_{ls}$	$\mathcal{P}_{cs}$	$\mathcal{P}_{cons}$	$\mathcal{P}_{ls}$ (NSCG)	$\mathcal{P}_{ln}$	$\mathcal{P}_{cn}$	$\mathcal{P}_{conn}$	$\mathcal{P}_{ln}$ (NSCG)
AUG3DC	37	71	35	35	88	166	91	100
AUG3DCQP	38	73	36	36	88	166	91	100
CONT-050	*	*	*	*	16	30	15	16
CONT-100	*	*	*	*	21	41	21	21
CONT-101	*	*	*	*	21	40	31	*
CONT-200	*	*	*	*	56	55	28	29
CVXQP3_S	83	150	83	*	26	44	26	29
DTOC3	*	*	*	*	5	10	6	7
GOULDQP3	21	39	19	19	40	71	41	38
HUES-MOD	4	4	3	2	3	4	9	2
HUESTIS	4	4	3	2	3	4	11	3
LASER	66	130	65	67	2	3	2	1
LISWET1	*	*	*	*	3	5	4	2
MOSARQP1	467	927	464	550	15	29	15	14
MOSARQP2	447	889	444	614	17	38	17	15
PRIMAL1	79	154	77	137	41	79	41	71
QPCSTAIR	249	490	247	551	53	93	53	69
STCQP2	269	528	267	615	94	95	93	93
YAO	*	*	*	*	3	5	4	2

Finally, we give results (Table 5) for the same tests with a more accurate approximation of S and N, namely the incomplete Cholesky factorization described at the start of this section. Generally, using these better approximations of N and S improves the iteration counts for the Schur complement preconditioners and the null-space preconditioners. Again, the null-space preconditioners are more robust than their Schur complement counterparts, and for no problems do we see the high iteration counts that the Schur complement preconditioners give for CONT-100, CONT-101,

TABLE 5

Iteration counts for the Schur complement preconditioners and the null-space preconditioners, with incomplete Cholesky preconditioners for S_0 and N_0, for the CUTEst *set matrices from Example* 4.2. ** stands for "did not converge after* 1000 *iterations."*

Matrix	$\mathcal{P}_{ls}$	$\mathcal{P}_{cs}$	$\mathcal{P}_{cons}$	$\mathcal{P}_{ls}$ (NSCG)	$\mathcal{P}_{ln}$	$\mathcal{P}_{cn}$	$\mathcal{P}_{conn}$	$\mathcal{P}_{ln}$ (NSCG)
AUG3DC	11	21	9	10	16	33	16	16
AUG3DCQP	11	21	9	9	16	33	16	16
CONT-050	86	170	85	86	18	34	17	17
CONT-100	377	752	376	417	40	59	39	44
CONT-101	419	837	418	536	26	48	25	*
CONT-200	*	*	*	*	57	43	23	23
CVXQP3_S	9	14	7	7	6	33	5	5
DTOC3	9	12	6	7	5	10	5	4
GOULDQP3	7	13	6	6	7	27	6	6
HUES-MOD	2	2	1	1	7	9	7	7
HUESTIS	2	2	1	1	7	10	7	*
LASER	12	20	10	10	2	3	1	1
LISWET1	6	9	4	6	2	4	1	1
MOSARQP1	26	51	25	25	7	22	7	7
MOSARQP2	28	53	26	26	7	19	6	6
PRIMAL1	5	8	4	4	13	25	12	12
QPCSTAIR	11	20	10	10	20	40	19	20
STCQP2	16	26	13	13	21	22	20	20
YAO	5	9	4	4	2	5	1	1

TABLE 6

Drop tolerances for the incomplete Cholesky preconditioners used.

Matrix	N_0		S_0	
	IC1	IC2	IC1	IC2
AUG3DC	10^{-3}	10^{-4}	10^{-1}	10^{-2}
MOSARQP1	10^{-1}	10^{-2}	10^{-3}	10^{-5}
STCQP2	10^{-1}	10^{-2}	10^{-4}	10^{-5}

and COND-200.

4.3. Analysis of computational costs for optimization and interior point method matrices. Next we examine in greater detail the computational costs involved in constructing and applying the null-space and Schur complement preconditioners for the CUTEst matrices AUG3DC, MOSARQP1, and STCQP2 from section 4.2. These matrices were chosen because they exhibit the full range of behaviors observed in the larger test set. The drop tolerances for the incomplete Cholesky preconditioners for these problems are given in Table 6.

Table 7 shows the times required to compute the preconditioners N_0 and S_0 and the number of nonzeros in the resulting factors. The cost of forming the full matrices N and S depends on the cost of solving systems with B_1 and A, respectively. For STCPQ2, $n - m \approx m$, and in this case N is quicker to compute than S; this is likely because $N = A_{22}$ is much sparser than S.[1] However, m is significantly smaller than $n - m$ for the other matrices, and so it is unsurprising that N is more expensive to form. The cost of the incomplete Cholesky factorizations, and the number of nonzeros in the factors, additionally depends on the drop tolerance and the nonzero pattern of N and S, as discussed at the start of this section.

In Table 8 we examine the time to solve the linear system. We see that for both STCQP2 and MOSARQP1 the timings are generally better for the null-space precon-

[1] We do not need to compute $N = A_{22}$, but we do so here for comparison with other problems.

TABLE 7

Time to compute N_0 or S_0, including the time to compute N or S when incomplete Cholesky factorizations are used, and the number of nonzeros in N_0 and S_0. For incomplete Cholesky preconditioners we report the number of nonzeros in one factor.

	Matrix	N_0			S_0		
		N	IC1	IC2	S	IC1	IC2
Time	AUG3DC	0.0025	0.013	0.036	0.0008	0.00042	0.00059
	MOSARQP1	0.39	0.39	0.39	0.013	0.015	0.016
	STCQP2	0.00062	0.00076	0.0014	1.2	1.2	1.3
nnz	AUG3DC	7.8E4	1.1E5	2.7E5	6.3E3	1.8E3	6.9E3
	MOSARQP1	5E3	1.8E3	2.2E3	8.6E3	1.4E4	3.1E4
	STCQP2	1.4E4	2.5E3	1E4	6.1E5	1.2E5	2.4E5

TABLE 8

Time to compute the solution using preconditioned GMRES or preconditioned nonstandard CG.

Matrix	N_0/S_0	$\mathcal{P}_{ls}$	$\mathcal{P}_{cs}$	$\mathcal{P}_{cons}$	$\mathcal{P}_{ls}$ (NSCG)	$\mathcal{P}_{ln}$	$\mathcal{P}_{cn}$	$\mathcal{P}_{conn}$	$\mathcal{P}_{ln}$ (NSCG)
AUG3DC	N	0.030	0.032	0.023	0.0078	0.046	0.27	0.034	0.024
	I	0.047	0.099	0.057	0.018	0.12	0.65	0.23	0.066
	IC1	0.070	0.10	0.096	0.020	0.076	0.18	0.066	0.14
	IC2	0.026	0.046	0.039	0.010	0.073	0.21	0.053	0.22
MOSARQP1	N	0.029	0.032	0.027	0.0076	0.025	0.076	0.044	0.012
	I	2.6	8.6	2.7	0.33	0.050	0.090	0.090	0.031
	IC1	0.069	0.10	0.096	0.029	0.054	0.11	0.084	0.030
	IC2	0.026	0.033	0.030	0.020	0.032	0.079	0.062	0.021
STCQP2	N	0.26	0.27	0.18	0.17	0.028	0.031	0.027	0.013
	I	3.3	7.1	6.5	3.8	0.44	0.17	0.36	0.11
	IC1	0.17	0.27	0.29	0.23	0.098	0.083	0.14	0.056
	IC2	0.12	0.16	0.15	0.27	0.064	0.094	0.075	0.047

ditioners, while Schur complement preconditioners perform better for AUG3DC. For AUG3DC and STCQP2, this correlates with the number of iterations. However, for STCQP2, fewer iterations are generally needed when Schur complement based preconditioners are applied, which shows that the cost of applying the preconditioners is important to consider. For STCQP2, N_0 is much sparser than S, which explains the difference between iteration counts and timings.

Finally, we remark that $\sigma = \boldsymbol{y}^T A_{22}\boldsymbol{y}/\boldsymbol{y}^T N\boldsymbol{y}$ in Theorem 2.1 appears to influence the number of iterations of $\mathcal{P}_{cn}$ significantly. When σ is near 1, as for STCQP2, few iterations are required. However, for AUG3DC and MOSARQP1, for which $\sigma_{\min} \ll 1$ (see Table 1), the number of iterations is quite a bit higher.

4.4. $\mathcal{F}$-matrices. Let $\mathcal{A}$ be a saddle point matrix of the form (1.1) where A is symmetric positive definite and B is a gradient matrix, i.e., B has at most two entries per row, and if there are two entries, they sum to zero; we call such a matrix $\mathcal{A}$ an $\mathcal{F}$-matrix [52]. Such matrices arise naturally in, e.g., discretizations of fluid-flow [3] or in electrical networks [50].

Due to the special structure of B, it is possible to find an invertible subblock B_1 without performing any arithmetic—see, e.g., [44, 50]. This property makes $\mathcal{F}$-matrices an ideal candidate for null-space preconditioning. We test our preconditioners for a number of $\mathcal{F}$-matrices,[2] listed in Table 9. We set $maxit = 1000$.

When we use the cheap, but inaccurate, approximations $S_0 = I$ and $N_0 = I$ (see Table 10), the iteration counts can be quite high for all preconditioners. However, the null-space based preconditioners consistently give lower iteration counts; this can partly be explained by the dimensions of the problems, since in general $n - m$ is significantly smaller than m. As in the previous example, nonstandard CG with $\mathcal{P}_{ls}$ or $\mathcal{P}_{ln}$ seems to be less robust than right-preconditioned GMRES. Similarly to other

[2]We would like to thank Miroslav Tůma for providing these test matrices.

examples in this section, the central-Schur and central-null preconditioners tend to take twice as many iterations as the other preconditioners. We see from Table 11 that when $\widetilde{N}$ and S are replaced by incomplete Cholesky preconditioners, the iteration counts drop for all preconditioners, but the same trends are evident. In particular, the null-space preconditioners are particularly well suited to these $\mathcal{F}$-matrices.

TABLE 9
Problem sizes, $\mathcal{F}$-matrices in Example 4.4.

Matrix	n	m	Matrix	n	m
DORT	13360	9607	M3P	2160	1584
DORT2	7515	5477	S3P	270	207
L3P	17280	12384	dan2	63750	46661

TABLE 10
Iteration counts for the Schur complement preconditioners with $S_0 = I_n$ and the null-space preconditioners with $N_0 = I_{n-m}$ for the $\mathcal{F}$-matrices in Example 4.4. * *stands for "did not converge after* 1000 *iterations."*

Matrix	$\mathcal{P}_{ls}$	$\mathcal{P}_{cs}$	$\mathcal{P}_{cons}$	$\mathcal{P}_{ls}$ (NSCG)	$\mathcal{P}_{ln}$	$\mathcal{P}_{cn}$	$\mathcal{P}_{conn}$	$\mathcal{P}_{ln}$ (NSCG)
DORT	*	*	*	*	763	*	750	*
DORT2	*	*	*	*	481	946	473	*
L3P	360	717	373	375	223	441	217	288
M3P	205	407	224	207	91	177	89	108
S3P	114	225	126	115	36	65	36	34
dan2	*	*	*	*	*	*	*	*

TABLE 11
Iteration counts for the Schur complement preconditioners and the null-space preconditioners, with incomplete Cholesky preconditioners for S_0 and N_0, for the $\mathcal{F}$-matrices in Example 4.4.

Matrix	$\mathcal{P}_{ls}$	$\mathcal{P}_{cs}$	$\mathcal{P}_{cons}$	$\mathcal{P}_{ls}$ (NSCG)	$\mathcal{P}_{ln}$	$\mathcal{P}_{cn}$	$\mathcal{P}_{conn}$	$\mathcal{P}_{ln}$ (NSCG)
DORT	121	237	120	125	14	32	12	12
DORT2	117	233	116	120	10	30	8	8
L3P	44	87	43	43	18	36	15	15
M3P	24	47	23	23	15	31	11	11
S3P	12	23	11	11	12	28	9	9
dan2	7	13	6	6	11	48	8	8

5. Conclusion. We have presented novel preconditioners based on a null-space factorization. By dropping, or approximating, different terms in the null-space factorization, in a similar manner to standard Schur complement preconditioners, we arrived at four different null-space preconditioners.

We have given eigenvalue bounds for these preconditioners and have shown that the eigenvalues of the upper-null, lower-null, and constraint preconditioners are clustered when a good approximation to the null-space matrix can be found. Additionally, two of the preconditioners, although indefinite and nonsymmetric, can be applied with a Krylov method with a short term recurrence.

Finally, we investigated the effectiveness of these preconditioners at reducing the number of iterations of Krylov subspace methods. We found that the preconditioners were more robust than equivalent Schur complement based preconditioners and were more effective when a reasonable approximation to the null-space matrix was available or when the dimension of $n - m$ was small.

Acknowledgments. The authors extend their thanks to Jennifer Scott and Nick Gould for reading an earlier version of this manuscript, and for their valuable comments and suggestions. They also thank two anonymous referees for their careful reading and constructive comments.

REFERENCES

[1] G. Al-Jeiroudi and J. Gondzio, *Convergence analysis of the inexact infeasible interior-point method for linear optimization*, J. Optim. Theory Appl., 141 (2009), pp. 231–247.

[2] G. Al-Jeiroudi, J. Gondzio, and J. Hall, *Preconditioning indefinite systems in interior point methods for large scale linear optimisation*, Optim. Methods Softw., 23 (2008), pp. 345–363.

[3] M. Arioli and G. Manzini, *A null space algorithm for mixed finite-element approximations of Darcy's equation*, Comm. Numer. Methods Engrg., 18 (2002), pp. 645–657.

[4] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[5] M. Benzi and M. A. Olshanskii, *An augmented Lagrangian-based approach to the Oseen problem*, SIAM J. Sci. Comput., 28 (2006), pp. 2095–2113, doi:10.1137/050646421.

[6] M. Benzi and V. Simoncini, *On the eigenvalues of a class of saddle point matrices*, Numer. Math., 103 (2006), pp. 173–196.

[7] L. Bergamaschi, J. Gondzio, and G. Zilli, *Preconditioning indefinite systems in interior point methods for optimization*, Comput. Optim. Appl., 28 (2004), pp. 149–171.

[8] L. T. Biegler, J. Nocedal, and C. Schmid, *A reduced Hessian method for large-scale constrained optimization*, SIAM J. Optim., 5 (1995), pp. 314–347, doi:10.1137/0805017.

[9] D. Braess, *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*, 3rd ed., Cambridge University Press, Cambridge, UK, 2007.

[10] J. H. Bramble and J. E. Pasciak, *A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems*, Math. Comp., 50 (1988), pp. 1–17.

[11] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam, *Algorithm 887: Cholmod, supernodal sparse Cholesky factorization and update/downdate*, ACM Trans. Math. Software, 35 (2008), 22.

[12] T. F. Coleman and A. R. Conn, *On the local convergence of a quasi-Newton method for the nonlinear programming problem*, SIAM J. Numer. Anal., 21 (1984), pp. 755–769, doi:10.1137/0721051.

[13] M. D'Apuzzo, V. De Simone, and D. di Serafino, *On mutual impact of numerical linear algebra and large-scale optimization with focus on interior point methods*, Comput. Optim. Appl., 45 (2010), pp. 283–310.

[14] C. R. Dohrmann and R. B. Lehoucq, *A primal-based penalty preconditioner for elliptic saddle point systems*, SIAM J. Numer. Anal., 44 (2006), pp. 270–282, doi:10.1137/040619016.

[15] H. S. Dollar, N. I. M. Gould, M. Stoll, and A. J. Wathen, *Preconditioning saddle-point systems with applications in optimization*, SIAM J. Sci. Comput., 32 (2010), pp. 249–270, doi:10.1137/080727129.

[16] H. Elman, D. Silvester, and A. Wathen, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*, Numerical Mathematics and Scientific Computation, Oxford University Press, New York, 2005.

[17] R. Estrin and C. Greif, *On nonsingular saddle-point systems with a maximally rank deficient leading block*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 367–384, doi:10.1137/140989996.

[18] B. Fischer, A. Ramage, D. J. Silvester, and A. J. Wathen, *Minimum residual methods for augmented systems*, BIT, 38 (1998), pp. 527–543.

[19] P. E. Gill and M. W. Leonard, *Limited-memory reduced-Hessian methods for large-scale unconstrained optimization*, SIAM J. Optim., 14 (2003), pp. 380–401, doi:10.1137/S1052623497319973.

[20] N. I. M. Gould, M. E. Hribar, and J. Nocedal, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395, doi:10.1137/S1064827598345667.

[21] N. I. M. Gould, D. Orban, and T. Rees, *Projected Krylov methods for saddle-point systems*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 1329–1343, doi:10.1137/130916394.

[22] N. I. M. Gould, D. Orban, and P. L. Toint, GALAHAD*, a library of thread-safe Fortran* 90 *packages for large-scale nonlinear optimization*, ACM Trans. Math. Software, 29 (2003), pp. 353–372.

[23] N. I. M. Gould, D. Orban, and P. L. Toint, CUTEst*: A constrained and unconstrained testing environment with safe threads for mathematical optimization*, Comput. Optim. Appl., 60 (2015), pp. 545–557.
[24] A. Greenbaum, V. Pták, and Z. Strakoš, *Any nonincreasing convergence curve is possible for GMRES*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 465–469, doi:10.1137/S0895479894275030.
[25] C. Greif, E. Moulding, and D. Orban, *Bounds on eigenvalues of matrices arising from interior-point methods*, SIAM J. Optim., 24 (2014), pp. 49–83, doi:10.1137/120890600.
[26] C. Greif and D. Schötzau, *Preconditioners for the discretized time-harmonic Maxwell equations in mixed form*, Numer. Linear Algebra Appl., 14 (2007), pp. 281–297.
[27] C. A. Hall and X. Ye, *Construction of null bases for the divergence operator associated with incompressible Navier-Stokes equations*, Linear Algebra Appl., 171 (1992), pp. 9–52.
[28] M. T. Heath, R. J. Plemmons, and R. C. Ward, *Sparse orthogonal schemes for structural optimization using the force method*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 514–532, doi:10.1137/0905038.
[29] C. Keller, N. I. M. Gould, and A. J. Wathen, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1300–1317, doi:10.1137/S0895479899351805.
[30] P. Krzyżanowski, *On block preconditioners for saddle point problems with singular or indefinite* $(1, 1)$ *block*, Numer. Linear Algebra Appl., 18 (2011), pp. 123–140.
[31] S. Le Borne, *Preconditioned nullspace method for the two-dimensional Oseen problem*, SIAM J. Sci. Comput., 31 (2009), pp. 2494–2509, doi:10.1137/070691577.
[32] D. Li, C. Greif, and D. Schötzau, *Parallel numerical solution of the time-harmonic Maxwell equations in mixed form*, Numer. Linear Algebra Appl., 19 (2012), pp. 525–539.
[33] J. Liesen and B. N. Parlett, *On nonsymmetric saddle point matrices that allow conjugate gradient iterations*, Numer. Math., 108 (2008), pp. 605–624.
[34] K.-A. Mardal and R. Winther, *Preconditioning discretizations of systems of partial differential equations*, Numer. Linear Algebra Appl., 18 (2011), pp. 1–40.
[35] M. F. Murphy, G. H. Golub, and A. J. Wathen, *A note on preconditioning for indefinite linear systems*, SIAM J. Sci. Comput., 21 (2000), pp. 1969–1972, doi:10.1137/S1064827599355153.
[36] J. Nocedal and M. L. Overton, *Projected Hessian updating algorithms for nonlinearly constrained optimization*, SIAM J. Numer. Anal., 22 (1985), pp. 821–850, doi:10.1137/0722050.
[37] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, New York, 1999.
[38] Y. Notay, *A new analysis of block preconditioners for saddle point problems*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 143–173, doi:10.1137/130911962.
[39] J. Pestana, *On the eigenvalues and eigenvectors of block triangular preconditioned block matrices*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 517–525, doi:10.1137/130920897.
[40] J. Pestana and A. J. Wathen, *On the choice of preconditioner for minimum residual methods for non-Hermitian matrices*, J. Comput. Appl. Math., 249 (2013), pp. 57–68.
[41] B. Peters and F. J. Herrmann, *A sparse reduced Hessian approximation for multi-parameter wavefield reconstruction inversion*, in SEG Technical Program Expanded Abstracts 2014, Society of Exploration Geophysicists, Tulsa, OK, 2014, pp. 1206–1210.
[42] M. Porcelli, V. Simoncini, and M. Tani, *Preconditioning of Active-Set Newton Methods for PDE-Constrained Optimal Control Problems*, preprint, arXiv:1407.1144 [math.NA], 2015.
[43] T. Rees, H. S. Dollar, and A. J. Wathen, *Optimal solvers for PDE-constrained optimization*, SIAM J. Sci. Comput., 32 (2010), pp. 271–298, doi:10.1137/080727154.
[44] T. Rees and J. A. Scott, *The Null-Space Method and Its Relationship with Matrix Factorizations for Sparse Saddle Point Systems*, Tech. Report RAL-TR-2014-016, STFC Rutherford Appleton Laboratory, Oxfordshire, UK, 2014.
[45] T. Rees and M. Stoll, *Block triangular preconditioners for PDE-constrained optimization*, Numer. Linear Algebra Appl., 17 (2010), pp. 977–996.
[46] J. Rommes and W. H. A. Schilders, *Efficient methods for large resistor networks*, IEEE Trans. Computer-Aided Design Integrated Circuits Syst., 29 (2010), pp. 28–39.
[47] M. Rozložník and V. Simoncini, *Krylov subspace methods for saddle point problems with indefinite preconditioning*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 368–391, doi:10.1137/S0895479800375540.
[48] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869, doi:10.1137/0907058.
[49] M. Saunders, *LUSOL: A Basis Package for Constrained Optimization*, SOL, Stanford University, Stanford, CA, http://web.stanford.edu/group/SOL/software/lusol/, 2013.

[50] W. H. Schilders, *Solution of indefinite linear systems using an LQ decomposition for the linear constraints*, Linear Algebra Appl., 431 (2009), pp. 381–395.
[51] J. Schöberl and W. Zulehner, *Symmetric indefinite preconditioners for saddle point problems with applications to PDE-constrained optimization problems*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 752–773, doi:10.1137/060660977.
[52] M. Tůma, *A note on the LDL^T decomposition of matrices from saddle-point problems*, SIAM J. Matrix Anal. Appl., 23 (2002), pp. 903–915, doi:10.1137/S0895479897321088.
[53] H. van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644, doi:10.1137/0913035.
[54] S. J. Wright, *Primal-Dual Interior-Point Methods*, SIAM, Philadelphia, 1997.