Two-level Nystrom-Schur preconditioner for sparse symmetric positive definite matrices

Article

Accepted Version

TWO-LEVEL NYSTRÖM–SCHUR PRECONDITIONER FOR SPARSE SYMMETRIC POSITIVE DEFINITE MATRICES*

HUSSAM AL DAAS†, TYRONE REES†, AND JENNIFER SCOTT†‡

Abstract. Randomized methods are becoming increasingly popular in numerical linear algebra. However, few attempts have been made to use them in developing preconditioners. Our interest lies in solving large-scale sparse symmetric positive definite linear systems of equations where the system matrix is preordered to doubly bordered block diagonal form (for example, using a nested dissection ordering). We investigate the use of randomized methods to construct high quality preconditioners. In particular, we propose a new and efficient approach that employs Nyström's method for computing low rank approximations to develop robust algebraic two-level preconditioners. Construction of the new preconditioners involves iteratively solving a smaller but denser symmetric positive definite Schur complement system with multiple right-hand sides. Numerical experiments on problems coming from a range of application areas demonstrate that this inner system can be solved cheaply using block conjugate gradients and that using a large convergence tolerance to limit the cost does not adversely affect the quality of the resulting Nyström–Schur two-level preconditioner.

Key words. Randomized methods, Nyström's method, Low rank, Schur complement, Deflation, Sparse symmetric positive definite systems, Doubly bordered block diagonal form, Block Conjugate Gradients, Preconditioning.

1. Introduction.

Large scale linear systems of equations arise in a wide range of real-life applications. Since the 1970s, sparse direct methods, such as LU, Cholesky, and LDLT factorizations, have been studied in depth and library quality software is available (see, for example, [9] and the references therein). However, their memory requirements and the difficulties in developing effective parallel implementations can limit their scope for solving extremely large problems, unless they are used in combination with an iterative approach. Iterative methods are attractive because they have low memory requirements and are simpler to parallelize. In this work, our interest is in using the conjugate gradient (CG) method to solve large sparse symmetric positive definite (SPD) systems of the form

$$Ax = b, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is SPD, $b \in \mathbb{R}^n$ is the given right-hand side, and x is the required solution. The solution of SPD systems is ubiquitous in scientific computing, being required in applications as diverse as least-squares problems, non-linear optimization subproblems, Monte-Carlo simulations, finite element analysis, and Kalman filtering. In the following, we assume no additional structure beyond a sparse SPD system.

It is well known that the approximate solution x_k at iteration k of the CG method satisfies

$$\|x_\star - x_k\|_A \leq 2\|x_\star - x_0\|_A \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k, \tag{1.2}$$

where $x_\star$ is the exact solution, x_0 is the initial guess, $\|\cdot\|_A$ is the A-norm, and $\kappa(A) = \lambda_{\max}/\lambda_{\min}$ is the spectral condition number ($\lambda_{\max}$ and $\lambda_{\min}$ denote the largest and

*Submitted to the editors January 28, 2021.

†STFC Rutherford Appleton Laboratory, Harwell Campus, Didcot, Oxfordshire, OX11 0QX, UK (hussam.al-daas@stfc.ac.uk, tyrone.rees@stfc.ac.uk, jennifer.scott@stfc.ac.uk).

‡School of Mathematical, Physical and Computational Sciences, University of Reading, Reading RG6 6AQ, UK

smallest eigenvalues of A). The rate of convergence also depends on the distribution of the eigenvalues (as well as on b and x_0): eigenvalues clustered away from the origin lead to rapid convergence. If $\kappa(A)$ is large and the eigenvalues of A are evenly distributed, the system needs to be preconditioned to enhance convergence. This can be done by applying a linear operator $\mathcal{P}$ to (1.1), where $\mathcal{P} \in \mathbb{R}^{n\times n}$ is chosen so that the spectral condition number of $\mathcal{P}A$ is small and applying $\mathcal{P}$ is inexpensive. In some applications, knowledge of the provenance of A can help in building an efficient preconditioner. Algebraic preconditioners do not assume such knowledge, and include incomplete Cholesky factorizations, block Jacobi, Gauss–Seidel, and additive Schwarz; see, for example, [36]. These are referred to as *one-level* or *traditional* preconditioners [7, 43]. In general, algebraic preconditioners bound the largest eigenvalues of $\mathcal{P}A$ but encounter difficulties in controlling the smallest eigenvalues, which can lie close to the origin, hindering convergence.

Deflation strategies have been proposed to overcome the issues related to small eigenvalues. As explained in [25], the basic idea behind deflation is to "hide" certain parts of the spectrum of the matrix from the CG method, such that the CG iteration "sees" a system that has a much smaller condition number than the original matrix. The part of the spectrum that is hidden from CG is determined by the deflation subspace and the improvement in the convergence rate of the deflated CG method is dependent on the choice of this subspace. In the ideal case, the deflation subspace is the invariant subspace spanned by the eigenvectors associated with the smallest eigenvalues of A and the convergence rate is then governed by the "effective" spectral condition number associated with the remaining eigenvalues (that is, the ratio of the largest eigenvalue to the smallest remaining eigenvalue). The idea was first introduced in the late 1980s [8, 33], and has been discussed and used by a number of researchers [2, 3, 10, 14, 22, 23, 27, 32, 40, 41, 45, 46]. However, in most of these references, the deflation subspaces rely on the underlying partial differential equation and its discretization, and cannot be applied to more general systems or used as "black box" preconditioners. Algebraic two-level preconditioners have been proposed in [4, 11, 15, 30, 43, 44]. Recently, a two-level Schur complement preconditioner based on the power series approximation was proposed in [50].

In recent years, the study of randomized methods has become an active and promising research area in the field of numerical linear algebra (see, for example, [16, 31] and the references therein). The use of randomized methods to build preconditioners has been proposed in a number of papers, including [14, 18]. The approach in [14] starts by reordering the system matrix A to a 2×2 doubly bordered block diagonal form, which can be achieved using a nested dissection ordering. The Schur complement system must then be solved. Starting from a first-level preconditioner $\mathcal{P}$, a deflation subspace is constructed via a low rank approximation. Although deflation can be seen as a low rank correction, using randomized methods to estimate the low rank term is not straightforward because the deflation subspace is more likely to be associated with the invariant subspace corresponding to the smallest eigenvalues of the preconditioned matrix, and not to its dominant subspace. In section 2, we review the ingredients involved in building our two-level preconditioner. This includes Nyström's method for computing a low rank approximation of a matrix [12, 16, 34, 47, 48], basic ideas behind deflation preconditioners, and the two-level Schur complement preconditioners presented in [14, 27]. In section 3, we illustrate the difficulties in constructing these two-level preconditioners by analysing the eigenvalue problems that must be solved. We show that these difficulties are mainly associated with the clustering of eigenvalues near

Identifier	n	$nnz(A)$	$\kappa(A)$	n_Γ	2D/3D	Application	Source
bcsstk38	8,032	355,460	5.5e+16	2,589	2D	Structural problem	SSMC
ela2d	45,602	543,600	1.5e+8	4,288	2D	Elasticity problem	FF++
ela3d	9,438	312,372	4.5e+5	4,658	3D	Elasticity problem	FF++
msc10848	10,848	1,229,776	1.0e+10	4,440	3D	Structural problem	SSMC
nd3k	9,000	3,279,690	1.6e+7	1,785	3D	Not available	SSMC
s3rmt3m3	5,357	207,123	2.4e+10	2,058	2D	Structural problem	SSMC

TABLE 1

Set of test matrices. n and $nnz(A)$ denote the order of A and the number of nonzero entries in A disregarding, $\kappa(A)$ is the spectral condition number, n_Γ is the order of the Schur complement (2.11). SSMC refers to SuiteSparse Matrix Collection [5]. FF++ refers to FreeFem++ [17].

the origin. Motivated by this analysis, in section 4 we propose reformulating the approximation problem.

The new formulation leads to well-separated eigenvalues that lie away from the origin, and this allows randomized methods to be used to compute a deflation subspace. Our approach guarantees a user-defined upper bound on the expected value of the spectral condition number of the preconditioned matrix. Numerical results for our new preconditioner and comparisons with other approaches are given in section 5. Concluding remarks are made in section 6.

Our main contributions are:

- an analysis of the eigenvalue problems and solvers presented in [14, 27];
- a reformulation of the eigenvalue problem so that it be efficiently solving using randomized methods;
- a new two-level preconditioner for symmetric positive definite systems that we refer to as a two-level Nyström–Schur preconditioner;
- theoretical bounds on the expected value of the spectral condition number of the preconditioned system.

Test environment. In this study, to demonstrate our theoretical and practical findings, we report on numerical experiments using the test matrices given in Table 1. This set was chosen to include 2D and 3D problems having a range of densities and with relatively large spectral condition numbers. In the Appendix, results are given for a much larger set of matrices. For each test, the entries of the right-hand side vector f are taken to be random numbers in the interval $[0, 1]$. All experiments are performed using Matlab 2020b.

Notation. Throughout this article, matrices are denoted using uppercase letters; scalars and vectors are lowercase. The pseudo inverse of a matrix C is denoted by $C^\dagger$ and its transpose is given by $C^\top$. $\Lambda(M)$ denotes the spectrum of the matrix M and $\kappa(M)$ denotes its condition number. $\Lambda_k = diag(\lambda_1, \ldots, \lambda_k)$ denotes a $k \times k$ diagonal matrix with entries on the diagonal equal to $\lambda_1, \ldots, \lambda_k$. $\widetilde{S}$ (with or without a subscript or superscript) is used as an approximation to a Schur complement matrix. $\mathcal{P}$ (with or without a subscript) denotes a (deflation) preconditioner. $\mathcal{M}$ (with or without a subscript) denotes a two-level (deflation) preconditioner. Matrices with an upper symbol such as $\widetilde{Z}$, $\widehat{Z}$, and $\check{Z}$ denote approximations of the matrix Z. Euler's constant is denoted by e.

2. Background. We start by presenting a brief review of Nyström's method for computing a low rank approximation to a matrix and then recalling key ideas behind two-level preconditioners; both are required in later sections.

2.1. Nyström's method. Given a matrix G, the Nyström approximation of a SPSD matrix B is defined to be

$$BG(G^\top BG)^\dagger (BG)^\top. \tag{2.1}$$

We observe that there are a large number of variants based on different choices of G (for example, [16, 28, 31]). For $q \geq 0$, the q-power iteration Nyström method is obtained by choosing

$$G = B^q \Omega, \tag{2.2}$$

for a given (random) starting matrix Ω. Note that, in practice, for stability it is normally necessary to orthonormalize the columns between applications of B.

The variant of Nyström's method we employ is outlined in Algorithm 2.1. It gives a near-optimal low rank approximation to B and is particularly effective when the eigenvalues of B decay rapidly after the k-th eigenvalue [16, 31]. It requires only one matrix-matrix product with B (or $q+1$ products if (2.2) is used). The rank of the resulting approximation is $\min(r, k)$, where r is the rank of D_1, see Step 5.

Algorithm 2.1 Nyström's method for computing a low rank approximation to a SPSD matrix.

Input: A SPSD matrix $B \in \mathbb{R}^{n\times n}$, the required rank $k > 0$, an oversampling parameter $p \geq 0$ such that $k, p \ll n$, and a threshold ε.

Output: $\widetilde{B}_k = \widetilde{U}_k \widetilde{\Sigma}_k \widetilde{U}_k^\top \approx B$ where $\widetilde{U}_k$ is orthonormal $\widetilde{\Sigma}_k$ is diagonal with non negative entries.

1: Draw a random matrix $G \in \mathbb{R}^{n\times(k+p)}$.
2: Compute $F = BG$.
3: Compute the QR factorization $F = QR$.
4: Set $C = G^\top F$.
5: Compute the EVD $C = V_1 D_1 V_1^\top + V_2 D_2 V_2^\top$, where D_1 contains all the eigenvalues that are at least ε.
6: Set $T = RV_1 D_1^{-1} (RV_1)^\top$.
7: Compute the EVD $T = WEW^\top$.
8: Set $\widetilde{U} = QW$, $\widetilde{U}_k = \widetilde{U}(:, 1:k)$, $\widetilde{\Sigma} = E(1:k, 1:k)$, and $\widetilde{B}_k = \widetilde{U}_k \widetilde{\Sigma}_k \widetilde{U}_k^\top$.

Note that, if the eigenvalues are ordered in descending order, the success of Nyström's method is closely related to the ratio of the $(k+1)$th and the kth eigenvalues. If the ratio is approximately equal to one, q must be large to obtain a good approximation [37].

2.2. Introduction to two-level preconditioners. Consider the linear system (1.1). As already noted, deflation techniques are typically used to shift isolated clusters of small eigenvalues to obtain a tighter spectrum and a smaller condition number. Such changes have a positive effect on the convergence of Krylov subspace methods. Consider the general (left) preconditioned system

$$\mathcal{P}Ax = \mathcal{P}b, \qquad \mathcal{P} \in \mathbb{R}^{n\times n}. \tag{2.3}$$

Given a projection subspace matrix $Z \in \mathbb{R}^{n\times k}$ of full rank and $k \ll n$, define the nonsingular matrix $E = Z^\top AZ \in \mathbb{R}^{k\times k}$ and the matrix $Q = ZE^{-1}Z^\top \in \mathbb{R}^{n\times n}$. The deflation preconditioner $\mathcal{P}_{\text{DEF}} \in \mathbb{R}^{n\times n}$ is defined to be [10]

$$\mathcal{P}_{\text{DEF}} = I - AQ. \tag{2.4}$$

It is straightforward to show that $\mathcal{P}_{\text{DEF}}$ is a projection matrix and $\mathcal{P}_{\text{DEF}}A$ has k zero eigenvalues (see [44] for basic properties of $\mathcal{P}_{\text{DEF}}$). To solve (1.1), we write

$$x = (I - \mathcal{P}_{\text{DEF}}^{\top})x + \mathcal{P}_{\text{DEF}}^{\top}x.$$

Since Q is symmetric, $\mathcal{P}_{\text{DEF}}^{\top} = I - QA$, and so

$$x = QAx + \mathcal{P}_{\text{DEF}}^{\top}x = Qb + \mathcal{P}_{\text{DEF}}^{\top}x,$$

and we only need to compute $\mathcal{P}_{\text{DEF}}^{\top}x$. We first find y that satisfies the deflated system

$$\mathcal{P}_{\text{DEF}}Ay = \mathcal{P}_{\text{DEF}}b, \tag{2.5}$$

then (due to the identity $A\mathcal{P}_{\text{DEF}}^{\top} = \mathcal{P}_{\text{DEF}}A$) we have that $\mathcal{P}_{\text{DEF}}^{\top}y = \mathcal{P}_{\text{DEF}}^{\top}x$. We therefore obtain the unique solution $x = Qb + \mathcal{P}_{\text{DEF}}^{\top}y$. The deflated system (2.5) is singular and can only be solved using CG if it is consistent [24], which is the case here since the same projection is applied to both sides of a consistent nonsingular system (1.1). The deflated system can also be solved using a preconditioner, giving a two-level preconditioner for the original system.

Tang *et al.* [44] illustrate that rounding errors can result in erratic and slow convergence of CG using $\mathcal{P}_{\text{DEF}}$. They thus also consider an adapted deflation preconditioner

$$\mathcal{P}_{\text{A-DEF}} = I - QA + Q, \tag{2.6}$$

that combines $\mathcal{P}_{\text{DEF}}^{\top}$ with Q. In exact arithmetic, both $\mathcal{P}_{\text{DEF}}$ and $\mathcal{P}_{\text{A-DEF}}$ used with CG generate the same iterates. However, numerical experiments [44] show that the latter is more robust and leads to better numerical behavior of CG[1].

Let $\lambda_n \geq \cdots \geq \lambda_1 > 0$ be the eigenvalues of A with associated normalized eigenvectors $v_n, \ldots, v_1$. For the ideal deflation preconditioner, $\mathcal{P}_{\text{ideal}}$, the deflation subspace is the invariant subspace spanned by the eigenvectors associated with the smallest eigenvalues. To demonstrate how $\mathcal{P}_{\text{ideal}}$ modifies the spectrum of the deflated matrix, set $Z_k = [v_1, \ldots, v_k]$ to be the $n \times k$ matrix whose columns are the eigenvectors corresponding to the smallest eigenvalues. It follows that $E = Z^{\top}AZ$ is equal to $\Lambda_k = diag(\lambda_1, \ldots, \lambda_k)$ and the preconditioned matrix is given by

$$\mathcal{P}_{\text{ideal}}A = A - Z_k\Lambda_k Z_k^{\top}.$$

Since Z_k is orthonormal and its columns span an invariant subspace, the spectrum of $\mathcal{P}_{\text{ideal}}A$ is $\{\lambda_n, \ldots, \lambda_{k+1}, 0\}$. Starting with x_0 such that $Z_k^{\top}r_0 = 0$ (r_0 is the initial residual), for $l \geq 0$, $Z_k^{\top}(\mathcal{P}_{\text{ideal}}A)^l r_0 = 0$ and $Z_k^{\top}A^l r_0 = 0$. Hence the search subspace generated by the preconditioned CG (PCG) method lies in the invariant subspace spanned by $v_n, \ldots, v_{k+1}$, which is orthogonal to the subspace spanned by the columns of Z_k. Consequently, the effective spectrum of the operator that PCG sees is $\{\lambda_n, \ldots, \lambda_{k+1}\}$ and the associated *effective spectral condition number* is

$$\kappa_{\text{eff}}(\mathcal{P}_{\text{ideal}}A) = \lambda_n/\lambda_{k+1}.$$

Using similar computations, the ideal adapted deflated system is given by:

$$\mathcal{P}_{\text{A-ideal}} = A - Z_k\Lambda_k^{-1}Z_k^{\top} + Z_kZ_k^{\top}. \tag{2.7}$$

[1] In [44], $\mathcal{P}_{\text{DEF}}$ and $\mathcal{P}_{\text{A-DEF}}$ are termed $\mathcal{P}_{\text{DEF1}}$ and $\mathcal{P}_{\text{A-DEF2}}$, respectively

Furthermore, the spectrum of the operator that PCG sees is $\{\lambda_n, \ldots, \lambda_{k+1}, 1, \ldots, 1\}$ and the associated effective spectral condition number is

$$\kappa_{\text{eff}}(\mathcal{P}_{\text{A-ideal}} A) = \max\{1, \lambda_n\} / \min\{1, \lambda_{k+1}\}.$$

In practice, only an approximation of the ideal deflation subspace spanned by the columns of Z_k is available. Kahl and Rittich [25] analyze the deflation preconditioner using $\widetilde{Z}_k \approx Z_k$ and present an upper bound on the corresponding effective spectral condition number of the deflated matrix $\kappa(\mathcal{P}A)$. Their bound [25, Proposition 4.3], which depends on $\kappa(A)$, $\kappa_{\text{eff}}(\mathcal{P}_{\text{ideal}} A)$, and the largest principal angle θ between $\widetilde{Z}_k$ and Z_k, is given by

$$\kappa(\mathcal{P}A) \leq \left(\sqrt{\kappa(A)} \sin\theta + \sqrt{\kappa_{\text{eff}}(\mathcal{P}_{\text{ideal}} A)} \right)^2, \tag{2.8}$$

where $\sin\theta = \| Z_k Z_k^\top - \widetilde{Z}_k \widetilde{Z}_k^\top \|_2$.

2.3. Schur Complement Preconditioners. This section reviews the Schur complement preconditioner with a focus on two-level variants that were introduced in [14, 27].

One-level preconditioners may not provide the required robustness when used with a Krylov subspace method because they typically fail to capture information about the eigenvectors corresponding to the smallest eigenvalues. To try and remedy this, in their (unpublished) report, Grigori *et al.* [14] and, independently, Li *et al.* [27] propose a two-level preconditioner based on using a block factorization and approximating the resulting Schur complement.

Applying graph partitioning techniques (for example, using the METIS package [26, 29]), A can be symmetrically permuted to the 2×2 doubly bordered block diagonal form

$$P^\top A P = \begin{pmatrix} A_I & A_{I\Gamma} \\ A_{\Gamma I} & A_\Gamma \end{pmatrix}, \tag{2.9}$$

where $A_I \in \mathbb{R}^{n_I \times n_I}$ is a block diagonal matrix, $A_\Gamma \in \mathbb{R}^{n_\Gamma \times n_\Gamma}$, $A_{\Gamma I} \in \mathbb{R}^{n_{\Gamma I} \times n_\Gamma}$ and $A_{I\Gamma} = A_{\Gamma I}^\top$. For simplicity of notation, we assume that A is of the form (2.9) (and omit the permutation P from the subsequent discussion).

The block form (2.9) induces a block LDLT factorization

$$A = \begin{pmatrix} I & \\ A_{\Gamma I} A_I^{-1} & I \end{pmatrix} \begin{pmatrix} A_I & \\ & S_\Gamma \end{pmatrix} \begin{pmatrix} I & A_I^{-1} A_{I\Gamma} \\ & I \end{pmatrix}, \tag{2.10}$$

where

$$S_\Gamma = A_\Gamma - A_{\Gamma I} A_I^{-1} A_{I\Gamma} \tag{2.11}$$

is the Schur complement of A with respect to A_Γ. Provided the blocks within A_I are small, they can be factorized cheaply in parallel using a direct algorithm (see, for example, [38]) and thus we assume that solving linear systems with A_I is not computationally expensive. However, the SPD Schur complement S_Γ is typically large and significantly denser than A_Γ (its size increases with the number of blocks in A_I) and, in large-scale practical applications, it may not be possible to explicitly assemble or factorize it.

Preconditioners may be derived by approximating S_Γ^{-1}. An approximate block factorization of A^{-1} is

$$M^{-1} = \begin{pmatrix} I & -A_I^{-1}A_{I\Gamma I} \\ & I \end{pmatrix} \begin{pmatrix} A_I^{-1} & \\ & \widetilde{S}^{-1} \end{pmatrix} \begin{pmatrix} I & \\ -A_{\Gamma I}A_I^{-1} & I \end{pmatrix},$$

where $\widetilde{S}^{-1} \approx S_\Gamma^{-1}$. If M^{-1} is employed as a preconditioner for A then the preconditioned system is given by

$$M^{-1}A = \begin{pmatrix} I & A_I^{-1}A_{I\Gamma}(I - \widetilde{S}^{-1}S_\Gamma) \\ & \widetilde{S}^{-1}S_\Gamma \end{pmatrix}, \tag{2.12}$$

with $\Lambda(M^{-1}A) = \{\lambda \in \Lambda(\widetilde{S}^{-1}S_\Gamma)\} \cup \{1\}$. Thus, to bound the condition number $\kappa(M^{-1}A)$, we need to construct $\widetilde{S}^{-1}$ so that $\kappa(\widetilde{S}^{-1}S_\Gamma)$ is bounded. Moreover, (2.12) shows that applying the preconditioner requires the efficient solution of linear systems with $\widetilde{S}^{-1}S_\Gamma$ and A_I, the latter being relatively inexpensive. We therefore focus on constructing preconditioners $\widetilde{S}^{-1}$ for linear systems of the form

$$S_\Gamma w = f. \tag{2.13}$$

Consider the first-level preconditioner obtained by setting

$$\widetilde{S}_1^{-1} := A_\Gamma^{-1}. \tag{2.14}$$

Assume for now that we can factorize A_Γ, although in practice it may be very large and a recursive construction of the preconditioner may then be needed (see [49]). Let the eigenvalues of the generalized eigenvalue problem

$$S_\Gamma z = \lambda \widetilde{S}_1 z \tag{2.15}$$

be $\lambda_{n_\Gamma} \geq \cdots \geq \lambda_1 > 0$. From (2.11), $\lambda_{n_\Gamma} \leq 1$ and so

$$\kappa(\widetilde{S}_1^{-1}S_\Gamma) = \frac{\lambda_{n_\Gamma}}{\lambda_1} \leq \frac{1}{\lambda_1}.$$

As this is unbounded as λ_1 approaches zero, we seek to add a low rank term to "correct" the approximation and shift the smallest k eigenvalues of $\widetilde{S}_1^{-1}S_\Gamma$. Let $\Lambda_k = diag\{\lambda_1, \ldots, \lambda_k\}$ and let $Z_k \in \mathbb{R}^{n_\Gamma \times k}$ be the matrix whose columns are the corresponding eigenvectors. Without loss of generality, we assume Z_k is A_Γ-orthonormal. Let the Cholesky factorization of A_Γ be

$$A_\Gamma = R_\Gamma^\top R_\Gamma \tag{2.16}$$

and define

$$\widetilde{S}_2^{-1} := A_\Gamma^{-1} + Z_k(\Lambda_k^{-1} - I)Z_k^\top. \tag{2.17}$$

$\widetilde{S}_2^{-1}$ is an additive combination of the first-level preconditioner $\widetilde{S}_1^{-1}$ and an adapted deflation preconditioner associated with the subspace spanned by the columns of $U_k = R_\Gamma Z_k$, which is an invariant subspace of $R_\Gamma^{-1}S_\Gamma R_\Gamma^{-\top}$. Substituting U_k into (2.17) and using (2.16),

$$\widetilde{S}_2^{-1} = R_\Gamma^{-1}(I + U_k(\Lambda_k^{-1} - I)U_k^\top)R_\Gamma^{-\top}. \tag{2.18}$$

Setting $Q = U_k \Lambda_k^{-1} U_k^\top$ in (2.6) gives

$$\mathcal{P}_{\text{A-DEF}} = R_\Gamma \widetilde{S}_2^{-1} R_\Gamma^\top.$$

Now $\widetilde{S}_2^{-1} S_\Gamma = R_\Gamma^{-1} \mathcal{P}_{\text{A-DEF}} R_\Gamma^{-\top} S_\Gamma$ and $\mathcal{P}_{\text{A-DEF}} R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ are spectrally equivalent and $\Lambda(\widetilde{S}_2^{-1} S_\Gamma) = \{\lambda_{n_\Gamma}, \lambda_{n_\Gamma - 1}, ..., \lambda_{k+1}\} \cup \{1\}$. It follows that

$$\kappa(\widetilde{S}_2^{-1} S_\Gamma) = \frac{\lambda_{n_\Gamma}}{\lambda_{k+1}} \leq \frac{1}{\lambda_{k+1}}.$$

Grigori *et al.* [14] note that (2.15) is equivalent to the generalized eigenvalue problem

$$(A_\Gamma - S_\Gamma) z = A_{\Gamma I} A_I^{-1} A_{I\Gamma} z = \sigma A_\Gamma z, \qquad \sigma = 1 - \lambda. \tag{2.19}$$

Setting $u = R_\Gamma z$ and defining

$$H = R_\Gamma^{-\top} A_{\Gamma I} A_I^{-1} A_{I\Gamma} R_\Gamma^{-1}, \tag{2.20}$$

(2.19) becomes

$$Hu = \sigma u. \tag{2.21}$$

Thus, the smallest eigenvalues λ of (2.15) are transformed to the largest eigenvalues σ of problems (2.19) and (2.21). Grigori *et al.* employ a randomized algorithm to compute a low rank eigenvalue decomposition (EVD) of H that approximates its largest eigenvalues and vectors, which are multiplied by R_Γ^{-1} to obtain approximate eigenvectors of $A_\Gamma^{-1} S_\Gamma$.

In [27], Li *et al.* write the inverse of the Schur complement S_Γ as:

$$\begin{aligned} S_\Gamma^{-1} &= \left(A_\Gamma - A_{\Gamma I} A_I^{-1} A_{I\Gamma}\right)^{-1} \\ &= \left(R_\Gamma^\top R_\Gamma - A_{\Gamma I} A_I^{-1} A_{I\Gamma}\right)^{-1} \\ &= R_\Gamma^{-1} \left(I - H\right)^{-1} R_\Gamma^{-\top}, \end{aligned} \tag{2.22}$$

where the symmetric positive semidefinite (SPSD) matrix H is given by (2.20). Since $I - H = R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ is SPD, the eigenvalues $\sigma_1 \geq \ldots \geq \sigma_{n_\Gamma}$ of H belong to $[0, 1]$. Let the EVD of H be

$$H = U \Sigma U^\top,$$

where U is orthonormal and $\Sigma = diag\{\sigma_1, \ldots, \sigma_{n_\Gamma}\}$. It follows that

$$\begin{aligned} S_\Gamma^{-1} &= R_\Gamma^{-1} \left(I - U \Sigma U^\top\right)^{-1} R_\Gamma^{-\top} \\ &= R_\Gamma^{-1} U \left(I - \Sigma\right)^{-1} U^\top R_\Gamma^{-\top} \\ &= R_\Gamma^{-1} \left(I + U \left(\left(I - \Sigma\right)^{-1} - I\right) U^\top\right) R_\Gamma^{-\top} \\ &= A_\Gamma^{-1} + R_\Gamma^{-1} U \left(\left(I - \Sigma\right)^{-1} - I\right) U^\top R_\Gamma^{-\top}. \end{aligned} \tag{2.23}$$

If H has an approximate EVD of the form

$$H \approx U \widetilde{\Sigma} U^\top, \qquad \widetilde{\Sigma} = diag\{\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_{n_\Gamma}\},$$

then an approximation of S_Γ^{-1} is

$$\widetilde{S}^{-1} = A_\Gamma^{-1} + R_\Gamma^{-1} U \left(\left(I - \widetilde{\Sigma} \right)^{-1} - I \right) U^\top R_\Gamma^{-\top}. \tag{2.24}$$

The simplest selection of $\widetilde{\Sigma}$ is the one that ensures the k largest eigenvalues of $(I-\widetilde{\Sigma})^{-1}$ match the largest eigenvalues of $(I-\Sigma)^{-1}$. Li *et al.* set $\widetilde{\Sigma} = diag(\sigma_1, \ldots, \sigma_k, \theta, \ldots, \theta)$, where $\theta \in [0, 1]$. The resulting preconditioner can be written as

$$\widetilde{S}_\theta^{-1} = \frac{1}{1-\theta} A_\Gamma^{-1} + Z_k \left((I - \Sigma_k)^{-1} - \frac{1}{1-\theta} I \right) Z_k^\top, \tag{2.25}$$

where $\Sigma_k = diag(\sigma_1, \ldots, \sigma_k)$ and the columns of $Z_k = R_\Gamma^{-1} U_k$ are the eigenvectors corresponding to the k largest eigenvalues of H. In [27], it is shown that $\kappa(\widetilde{S}_\theta^{-1} S) = (1 - \sigma_{n_\Gamma})/(1-\theta)$, which takes its minimum value for $\theta = \sigma_{k+1}$.

In the next section, we analyse the eigenvalue problems that need to be solved to construct the preconditioners (2.17) and (2.25). In particular, we show that the approaches presented in [14, 27] for tackling these problems are inefficient because of the eigenvalue distribution.

3. Analysis of $Hu = \sigma u$.

3.1. Use of the Lanczos method.

Consider the eigenproblem:

$$\begin{aligned} &\text{Given } \varepsilon > 0, \text{find all the eigenpairs } (\lambda, z) \in \mathbb{R} \times \mathbb{R}^{n_\Gamma} \text{ such that} \\ &S_\Gamma z = \lambda A_\Gamma z, \qquad \lambda < \varepsilon. \end{aligned}$$

This can be rewritten as:

$$\begin{aligned} &\text{Given } \varepsilon > 0, \text{find all the eigenpairs } (\lambda, z) \in \mathbb{R} \times \mathbb{R}^{n_\Gamma} \text{ such that} \\ &(I - H)u = \lambda u, \qquad z = R_\Gamma^{-1} u, \qquad \lambda < \varepsilon, \end{aligned} \tag{3.1}$$

where R_Γ and H are given by (2.16) and (2.20). Consider also the eigenproblem:

$$\begin{aligned} &\text{Given } \varepsilon > 0, \text{find all the eigenpairs } (\sigma, u) \in \mathbb{R} \times \mathbb{R}^{n_\Gamma} \text{ such that} \\ &Hu = \sigma u, \qquad \sigma > 1 - \varepsilon. \end{aligned} \tag{3.2}$$

As already observed, each eigenpair (λ, z) of (3.1) corresponds to the eigenpair $(1 - \lambda, R_\Gamma z)$ of (3.2). Consider using the Lanczos method to solve these eigenproblems. The Krylov subspace at iteration j generated for (3.1) is

$$K_j((I-H), v_1) = \text{span}(v_1, (I-H)v_1, \ldots, (I-H)^{j-1} v_1),$$

while the subspace generated for (3.2) is

$$K_j(H, v_1) = \text{span}(v_1, Hv_1, \ldots, H^{j-1} v_1).$$

It is clear that, provided the same starting vector v_1 is used, $K_j((I-H), v_1)$ and $K_j(H, v_1)$ are identical. Suppose that $[\mathcal{V}_j, v_{j+1}]$ is the output of the Lanczos basis of the Krylov subspace, then the subspace relations that hold at iteration j are

$$(I - H)\mathcal{V}_j = \mathcal{V}_j T_j + v_{j+1} h_j^\top,$$

$$H\mathcal{V}_j = \mathcal{V}_j(I - T_j) - v_{j+1}h_j^\top,$$

where $T_j \in \mathbb{R}^{j\times j}$ is a symmetric tridiagonal matrix and $h_j \in \mathbb{R}^j$. The eigenpair (λ, z) (respectively, (σ, u)) corresponding to the smallest (respectively, largest) eigenvalue in (3.1) (respectively, (3.2)) is approximated by the eigenpair $(\widetilde{\lambda}, R_\Gamma^{-1}\mathcal{V}_j\widetilde{u})$ (respectively, $(\widetilde{\sigma}, \mathcal{V}_j\widetilde{u})$) corresponding to the smallest (respectively, largest) eigenvalue of T_j (respectively, $I - T_j$). To overcome memory constraints, the Lanczos procedure is typically restarted after a chosen number of iterations, at each restart discarding the non convergent part of the Krylov subspace [42]. Hence, starting with the same v_1 and performing the same number of iterations per cycle, in exact arithmetic the accuracy obtained when solving (3.1) and (3.2) is identical.

Having shown that the convergence of Lanczos' method for solving (3.1) and (3.2) is the same, we focus on (3.2). In Figure 1, for each of our test matrices in Table 1

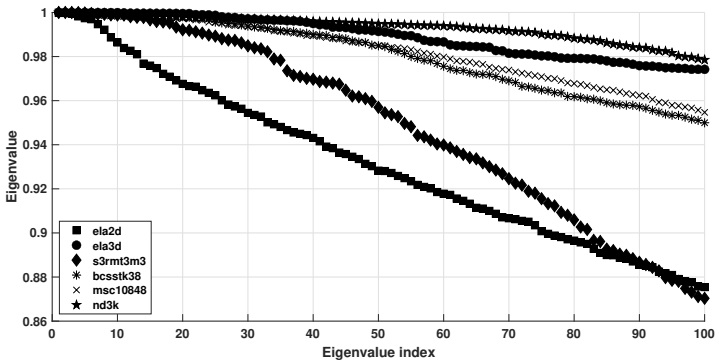

FIG. 1. *Largest 100 eigenvalues of* $H = R_\Gamma^{-\top}A_{\Gamma I}A_I^{-1}A_{I\Gamma}R_\Gamma^{-1}$ *associated with our test matrices computed to an accuracy of* 10^{-8} *using the Krylov-Schur method [42].*

we plot the 100 largest eigenvalues of the matrix H given by (2.20). We see that the largest eigenvalues (which are the ones that we require) are clustered near one and they do not decay rapidly. As there are a significant number of eigenvalues in the cluster, computing the largest k (for $k = O(10)$) and the corresponding eigenvectors with sufficient accuracy using the Lanczos method is challenging. Similar distributions were observed for the larger test set that we report on in the Appendix, particularly for problems for which the one-level preconditioner $\widetilde{S}_1$ was found to perform poorly, which is generally the case when $\kappa(A)$ is large. Table 2 reports the Lanczos iteration counts (it_{Lan}) for computing the $k = 20$ and 40 largest eigenpairs (that is, the number of linear systems that are solved in the Lanczos method). In addition, we present the PCG iteration count (it_{PCG}) for solving the linear system (2.13) using the first-level preconditioner $\widetilde{S}_1 = A_\Gamma^{-1}$ and the two-level preconditioner $\widetilde{S}_2$ given by (2.17). We see that, in terms of the total iteration count, the first-level preconditioner is the more efficient option. It is of interest to consider whether relaxing the convergence tolerance ε_{Lan} in the Lanczos method can reduce the total iteration count for $\widetilde{S}_2$. Table 3 illustrates the effect of varying ε_{Lan} for problem el3d (results for the other test problems are consistent). Although it_{Lan} decreases as ε_{Lan} increases, it_{PCG} increases and the total count still exceeds the 175 PCG iterations required by the first-level preconditioner $\widetilde{S}_1$.

As already observed, in [49] a recursive (multilevel) scheme is proposed to help mitigate the computational costs of building and applying the preconditioner.

	$\widetilde{S}_1$	$\widetilde{S}_2$					
		$k = 20$			$k = 40$		
Identifier	it_{PCG}	it_{Lan}	it_{PCG}	total	it_{Lan}	it_{PCG}	total
bcsstk38	584	797	122	919	730	67	797
el2d	914	1210	231	1441	982	120	1102
el3d	174	311	37	348	389	27	416
msc10848	612	813	116	929	760	63	823
nd3k	603	1796	143	1939	1349	105	1454
s3rmt3m3	441	529	70	599	480	37	517

TABLE 2

The Lanczos iteration count (it_{Lan}) and the iteration count for PCG (it_{PCG}). The convergence tolerance for the Lanczos method and PCG is 10^{-6}. The size of the Krylov subspace per cycle is $2k$.

	$k = 20$			$k = 40$		
$\varepsilon_{\mathrm{Lan}}$	it_{Lan}	it_{PCG}	total	it_{Lan}	it_{PCG}	total
0.1	50	131	181	80	101	181
0.08	50	131	181	100	85	185
0.06	60	121	181	100	85	185
0.04	82	100	182	120	71	191
0.02	127	64	201	207	37	244
0.01	169	41	210	259	32	291
0.005	213	38	251	316	29	345
0.001	247	37	284	372	28	400

TABLE 3

Problem el3d and two-level preconditioner $\widetilde{S}_2$: sensitivity of the number of the Lanczos iteration count (it_{Lan}) and the iteration count for PCG (it_{PCG}) to the convergence tolerance ε_{Lan}. The PCG convergence tolerance is 10^{-6}. The size of the Krylov subspace per cycle is $2k$.

Nevertheless, the Lanczos method is still used, albeit with reduced costs for applying the operator matrices.

3.2. Use of Nyström's method. As suggested in [14], an alternative approach to approximating the dominant subspace of H is to use a randomized method, specifically a randomized eigenvalue decomposition. Because H is SPSD, Nyström's method can be use. Results are presented in Table 4 for problem el3d (results for our other test examples are consistent with these). Here p is the oversampling parameter and q is the power iteration parameter. These show that, as with the Lanczos method, Nyström's method struggles to approximate the dominant eigenpairs of H. Using $k = 20$ (respectively, 40) exact eigenpairs, PCG using $\widetilde{S}_2$ requires 37 (respectively, 28) iterations. To obtain the same iteration counts using vectors computed using Nyström's method requires the oversampling parameter to be greater than 2000, which is clearly prohibitive. Using the power iteration improves the quality of the approximate subspace. However, the large value of q needed to decrease the PCG iteration count means a large number of linear systems must be solved with A_Γ, in addition to the work involved in the orthogonalization that is needed between the power iterations to maintain stability. Indeed, it is sufficient to look at Figure 1 to predict this behaviour for any randomized method applied to H. The lack of success of existing strategies motivates us, in the next section, to reformulate the eigenvalue

problem to one with a spectrum that is easy to approximate.

p	$k=20$	$k=40$
100	171	169
200	170	165
400	165	161
800	155	146
1600	125	111
3200	55	45

q	$k=20$	$k=40$
0	172	171
20	121	87
40	86	48
60	68	34
80	55	30
100	46	29

TABLE 4

PCG iteration counts for problem el3d using the two-level preconditioner $\widetilde{S}_2$ constructed using a rank k approximation of $H = R_\Gamma^{-\top} A_{\Gamma I} A_I^{-1} A_{I\Gamma} R_\Gamma^{-1}$. The PCG convergence tolerance is 10^{-6}. Nyström's method applied to H with the oversampling parameter $p \geq 100$ and the power iteration parameter $q = 0$ (left) and with $p = 0$ and $q \geq 0$ (right).

4. Nyström–Schur two-level preconditioner. In this section, we propose reformulating the eigenvalue problem to obtain a new one such that the desired eigenvectors correspond to the largest eigenvalues and these eigenvalues are well separated from the remaining eigenvalues: this is what is needed for randomized methods to be successful.

4.1. Two-level preconditioner for S_Γ. Applying the Sherman Morrison Woodbury identity [13, 2.1.3], the inverse of the Schur complement S_Γ (2.11) can be written as:

$$\begin{aligned} S_\Gamma^{-1} &= A_\Gamma^{-1} + A_\Gamma^{-1} A_{\Gamma I} (A_I - A_{I\Gamma} A_\Gamma^{-1} A_{\Gamma I})^{-1} A_{I\Gamma} A_\Gamma^{-1} \\ &= A_\Gamma^{-1} + A_\Gamma^{-1} A_{\Gamma I} S_I^{-1} A_{I\Gamma} A_\Gamma^{-1}, \end{aligned} \tag{4.1}$$

where

$$S_I = A_I - A_{I\Gamma} A_\Gamma^{-1} A_{\Gamma I} \tag{4.2}$$

is the Schur complement of A with respect to A_I. Using the Cholesky factorization (2.16), we have

$$R_\Gamma S_\Gamma^{-1} R_\Gamma^\top = I + R_\Gamma^{-\top} A_{\Gamma I} S_I^{-1} A_{I\Gamma} R_\Gamma^{-1}. \tag{4.3}$$

Note that if (λ, u) is an eigenpair of $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$, then $(\frac{1}{\lambda} - 1, u)$ is an eigenpair of $R_\Gamma^{-\top} A_{\Gamma I} S_I^{-1} A_{I\Gamma} R_\Gamma^{-1}$. Therefore, the cluster of eigenvalues of $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ near the origin (which correspond to the cluster of eigenvalues of H near 1) correspond to very large and highly separated eigenvalues of $R_\Gamma^{-\top} A_{\Gamma I} S_I^{-1} A_{I\Gamma} R_\Gamma^{-1}$. Hence, using randomized methods to approximate the dominant subspace of $R_\Gamma^{-\top} A_{\Gamma I} S_I^{-1} A_{I\Gamma} R_\Gamma^{-1}$ can be an efficient way of computing a deflation subspace for $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$. Now assume that we have a low rank approximation

$$R_\Gamma^{-\top} A_{\Gamma I} S_I^{-1} A_{I\Gamma} R_\Gamma^{-1} \approx \breve{U}_k \breve{\Sigma}_k \breve{U}_k^\top, \tag{4.4}$$

where $\breve{U}_k \in \mathbb{R}^{n_\Gamma \times k}$ is orthonormal and $\breve{\Sigma}_k \in \mathbb{R}^{k \times k}$ is diagonal. Combining (4.3) and (4.4), we can define a preconditioner for $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ to be

$$\mathcal{P}_1 = I + \breve{U}_k \breve{\Sigma}_k \breve{U}_k^\top. \tag{4.5}$$

The preconditioned matrix $\mathcal{P}_1 R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ is spectrally equivalent to $R_\Gamma^{-1}\mathcal{P}_1 R_\Gamma^{-\top} S_\Gamma$. Therefore, the preconditioned system can be written as

$$\mathcal{M}_1 S_\Gamma = R_\Gamma^{-1}\mathcal{P}_1 R_\Gamma^{-\top} S_\Gamma = \left(A_\Gamma^{-1} + \breve{Z}_k \breve{\Sigma}_k \breve{Z}_k^\top\right) S_\Gamma, \tag{4.6}$$

where $\breve{Z}_k = R_\Gamma^{-1}\breve{U}_k$. If (4.4) is obtained using a truncated EVD denoted by $U_k\Sigma_k U_k^\top$, then $\breve{U}_k = U_k$ and the subspace spanned by the columns of U_k is an invariant subspace of $R_\Gamma S_\Gamma^{-1} R_\Gamma^\top$ and of its inverse $R_\Gamma^{-1} S_\Gamma R_\Gamma^{-\top}$. Furthermore, using the truncated EVD, (4.5) is an adapted deflation preconditioner for $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$. Indeed, as the columns of U_k are orthonormal eigenvectors, we have from (4.3) that $R_\Gamma S_\Gamma^{-1} R_\Gamma^\top U_k = U_k(I+\Sigma_k)$. Hence $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} U_k = U_k(I+\Sigma_k)^{-1}$ and the preconditioned matrix is

$$\begin{aligned}
\mathcal{P}_{\text{A-DEF}} R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} &= R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} + U_k\Sigma_k(I+\Sigma_k)^{-1}U_k^\top \\
&= R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} + U_k\left((I+\Sigma_k) - I\right)(I+\Sigma_k)^{-1}U_k^\top \\
&= R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} - U_k(I+\Sigma_k)^{-1}U_k^\top + U_k U_k^\top,
\end{aligned}$$

which has the same form as the ideal adapted preconditioned matrix (2.7).

Note that given the matrix $\breve{U}_k$ in the approximation (4.4), then following subsection 2.2, we can define a deflation preconditioner for $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$. Setting $E_k = \breve{U}_k^\top R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}\breve{U}_k$ and $Q = \breve{U}_k E^{-1}\breve{U}_k^\top$, the deflation preconditioner is

$$\mathcal{P}_{\text{1-A-DEF}} = I - Q R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} + Q. \tag{4.7}$$

The preconditioned Schur complement $\mathcal{P}_{\text{1-A-DEF}} R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ is spectrally similar to $R_\Gamma^{-1}\mathcal{P}_{\text{1-A-DEF}} R_\Gamma^{-\top} S_\Gamma$ and thus

$$\mathcal{M}_{\text{1-A-DEF}} = R_\Gamma^{-1}\mathcal{P}_{\text{1-A-DEF}} R_\Gamma^{-\top} \tag{4.8}$$

is a two-level preconditioner for S_Γ.

4.2. Lanczos versus Nyström. The two-level preconditioner (4.8) relies on computing a low-rank approximation (4.4). We now consider the difference between using the Lanczos and Nyström methods for this.

Both methods require the application of $R_\Gamma^{-\top} A_{\Gamma I} S_I^{-1} A_{I\Gamma} R_\Gamma^{-1}$ to a set of $k+p$ vectors, where $k > 0$ is the required rank and $p \geq 0$. Because explicitly computing the SPD matrix $S_I = A_I - A_{I\Gamma}A_\Gamma^{-1}A_{\Gamma I}$ and factorizing it is prohibitively expensive, applying S_I^{-1} must be done using an iterative solver.

The Lanczos method builds a Krylov subspace of dimension $k+p$ in order to compute a low-rank approximation. Therefore, $k+p$ linear systems must be solved, each with one right-hand side, first for R_Γ, then for S_I, and then for $R_\Gamma^\top$. However, the Nyström method requires the solution of only one linear system with $k+p$ right-hand sides for R_Γ, then for S_I, and then for $R_\Gamma^\top$. This allows the use of matrix-matrix operations rather than less efficient matrix-vector operations. Moreover, as we will illustrate in section 5, block Krylov subspace methods, such as block CG [35], for solving the system with S_I yield faster convergence than their classical counterparts. When the Nyström method is used, we call the resulting preconditioner (4.8) the *Nyström–Schur preconditioner*.

4.3. Avoiding computations with R_Γ. For large scale problems, computing the Cholesky factorization $A_\Gamma = R_\Gamma^\top R_\Gamma$ is prohibitive and so we would like to avoid

computations with R_Γ. We can achieve this by using an iterative solver to solve linear systems with A_Γ. Note that this is possible when solving the generalized eigenvalue problem (2.15). Because A_Γ is typically well conditioned, so too is R_Γ. Thus, we can reduce the cost of computing the Nyström–Schur preconditioner by approximating the SPSD matrix $A_{\Gamma I}S_I^{-1}A_{I\Gamma}$ (or even by approximating S_I^{-1}). Of course, this needs to be done without seriously adversely affecting the preconditioner quality. Using an approximate factorization

$$A_{\Gamma I}S_I^{-1}A_{I\Gamma} \approx \widetilde{W}_k\widetilde{\Sigma}_k\widetilde{W}_k^\top, \tag{4.9}$$

an alternative deflation preconditioner is

$$\begin{aligned}\mathcal{P}_2 &= I + R_\Gamma^{-\top}\widetilde{W}_k\widetilde{\Sigma}_k\widetilde{W}_k^\top R_\Gamma^{-1},\\ &= R_\Gamma^{-\top}\left(A_\Gamma + \widetilde{W}_k\widetilde{\Sigma}_k\widetilde{W}_k^\top\right)R_\Gamma^{-1}.\end{aligned}$$

The preconditioned Schur complement $\mathcal{P}_2R_\Gamma^{-\top}S_\Gamma R_\Gamma^{-1}$ is spectrally similar to $R_\Gamma^{-1}\mathcal{P}_2R_\Gamma^{-\top}S_\Gamma$ and, setting $\widetilde{Z}_k = A_\Gamma^{-1}\widetilde{W}_k$, we have

$$\mathcal{M}_2S_\Gamma = R_\Gamma^{-1}\mathcal{P}_2R_\Gamma^{-\top}S_\Gamma = (A_\Gamma^{-1} + \widetilde{Z}_k\widetilde{\Sigma}_k\widetilde{Z}_k^\top)S_\Gamma. \tag{4.10}$$

Thus $\mathcal{M}_2 = A_\Gamma^{-1} + \widetilde{Z}_k\widetilde{\Sigma}_k\widetilde{Z}_k^\top$ is a variant of the Nyström–Schur preconditioner for S_Γ that avoids computing R_Γ.

Alternatively, assuming we have an approximate factorization

$$S_I^{-1} \approx \widehat{V}_k\widehat{\Sigma}_k\widehat{V}_k^\top, \tag{4.11}$$

yields

$$\mathcal{P}_3 = I + R_\Gamma^{-\top}A_{\Gamma I}\widehat{V}_k\widehat{\Sigma}_k\widehat{V}_k^\top A_{I\Gamma}R_\Gamma^{-1}.$$

Again, $\mathcal{P}_3R_\Gamma^{-\top}S_\Gamma R_\Gamma^{-1}$ is spectrally similar to $R_\Gamma^{-1}\mathcal{P}_3R_\Gamma^{-\top}S_\Gamma$ and, setting $\widehat{Z}_k = A_\Gamma^{-1}A_{\Gamma I}\widehat{V}_k$, we have

$$\mathcal{M}_3S_\Gamma = R_\Gamma^{-1}\mathcal{P}_3R_\Gamma^{-\top}S_\Gamma = (A_\Gamma^{-1} + \widehat{Z}_k\widehat{\Sigma}_k\widehat{Z}_k^\top)S_\Gamma, \tag{4.12}$$

which gives another variant of the Nyström–Schur preconditioner. In a similar way to defining $\mathcal{M}_{\text{1-A-DEF}}$ (4.7), we can define $\mathcal{M}_{\text{2-A-DEF}}$ and $\mathcal{M}_{\text{3-A-DEF}}$. Note that $\mathcal{M}_{\text{2-A-DEF}}$ and $\mathcal{M}_{\text{3-A-DEF}}$ also avoid computations with R_Γ.

4.4. Nyström–Schur preconditioner. Algorithm 4.1 presents the construction of the Nyström–Schur preconditioner $\mathcal{M}_2$; an analogous derivation yields the variant $\mathcal{M}_3$. Step 3 is the most expensive step, that is, solving the $n_I \times n_I$ SPD linear system

$$S_IX = F, \tag{4.13}$$

where $F \in \mathbb{R}^{n_I\times(k+p)}$ and $S_I = A_I - A_{I\Gamma}A_\Gamma^{-1}A_{\Gamma I}$. Using an iterative solver requires a linear system solve with A_Γ on each iteration. Importantly for efficiency, the number of iterations can be limited by employing a large relative tolerance when solving (4.13) without adversely affecting the performance of the resulting preconditioner. Numerical experiments in section 5 illustrate this robustness.

Observe that applying $\mathcal{M}_2$ to a vector requires the solution of a linear system with A_Γ and a low rank correction; see Step 12.

Algorithm 4.1 Construction of the Nyström–Schur preconditioner (4.10)

Input: A in block form (2.9), $k > 0$ and $p \geq 0$ ($k, p \ll n_\Gamma$) and $\varepsilon > 0$.
Output: Two-level preconditioner for the $n_\Gamma \times n_\Gamma$ Schur complement S_Γ.

1: Draw a random matrix $G \in \mathbb{R}^{n_\Gamma \times (k+p)}$.
2: Compute $F = A_{I\Gamma} G$.
3: Solve $S_I X = F$.
4: Compute $Y = A_{\Gamma I} X$.
5: Compute $Y = QR$.
6: Set $C = G^\top Y$.
7: Compute the EVD $C = V_1 D_1 V_1^\top + V_2 D_2 V_2^\top$, where D_1 contains all the eigenvalues that are at least ε.
8: Set $T = R V_1 D_1^{-1} V_1^\top R^\top$.
9: Compute the EVD $T = W E W^\top$.
10: Set $\widetilde{U} = YW(:, 1:k)$, $\Sigma = E(1:k, 1:k)$.
11: Solve $A_\Gamma Z = \widetilde{U}$.
12: Define the preconditioner $\mathcal{M}_2 = A_\Gamma^{-1} + Z \Sigma Z^\top$.

4.5. Estimation of the Spectral Condition Number. In this section, we provide an expectation of the spectral condition number of S_Γ preconditioned by the Nyström–Schur preconditioner. Saibaba [37] derives bounds on the angles between the approximate singular vectors computed using a randomized singular value decomposition and the exact singular vectors of a matrix. It is straightforward to derive the corresponding bounds for the Nyström method. Let Π_M denote the orthogonal projector on the space spanned by the columns of the matrix M. Let (λ_j, u_j), $j = 1, \ldots, k$, be the dominant eigenpairs of $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$. Following the notation in Algorithm 2.1, the angle $\theta_j = \angle(u_j, \widetilde{U})$ between the approximate eigenvectors $\widetilde{U} \in \mathbb{R}^{n_\Gamma \times (k+p)}$ of $R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ and the exact eigenvector $u_j \in \mathbb{R}^{n_\Gamma}$ satisfies

$$\sin \angle(u_j, \widetilde{U}) = \|u_j - \Pi_{\widetilde{U}} u_j\|_2 \leq \gamma_{j,k}^{q+1} c, \tag{4.14}$$

where q is the power iteration count (recall (2.2)), $\gamma_{j,k}$ is the gap between $\lambda_j^{-1} - 1$ and $\lambda_{k+1}^{-1} - 1$ given by

$$\gamma_{j,k} = (\lambda_{k+1}^{-1} - 1)/(\lambda_j^{-1} - 1), \tag{4.15}$$

and c has the expected value

$$\mathbb{E}(c) = \sqrt{\frac{k}{p-1}} + \frac{e\sqrt{(k+p)(n_\Gamma - k)}}{p}, \tag{4.16}$$

where k is the required rank and $p \geq 2$ is the oversampling parameter. Hence,

$$\mathbb{E}\left(\sin \angle(u_j, \widetilde{U})\right) = \mathbb{E}\left(\|u_j - \Pi_{\widetilde{U}} u_j\|_2\right) \leq \gamma_{j,k}^{q+1} \mathbb{E}(c). \tag{4.17}$$

Note that if $\lambda_j \leq 1/2$ then $\gamma_{j,k} \leq 2\lambda_j / \lambda_{k+1}$ ($j = 1, \ldots, k$).

PROPOSITION 4.1. *Let the EVD of the SPD matrix $I - H = R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1}$ be*

$$\begin{bmatrix} U_\perp & U_k \end{bmatrix} \begin{bmatrix} \Lambda_\perp & \\ & \Lambda_k \end{bmatrix} \begin{bmatrix} U_\perp^\top \\ U_k^\top \end{bmatrix},$$

where $\Lambda_\perp \in \mathbb{R}^{(n_\Gamma-k)\times(n_\Gamma-k)}$ *and* $\Lambda_k \in \mathbb{R}^{k\times k}$ *are diagonal matrices with the eigenvalues* $(\lambda_i)_{k\geq i\geq 1}$ *and* $(\lambda_i)_{n_\Gamma\geq i\geq k+1}$*, respectively, in decreasing order. Furthermore, assume that* $\lambda_k \leq 1/2$*. Let the columns of* $\widetilde{U} \in \mathbb{R}^{n_\Gamma\times(k+p)}$ *be the approximate eigenvectors of* $I-H$ *computed using the Nyström method and let*

$$\mathcal{P} = I - (I-H)\widetilde{U}E^{-1}\widetilde{U}^\top \quad \textit{with} \quad E = \widetilde{U}^\top(I-H)\widetilde{U},$$

be the associated deflation preconditioner. Then, the effective condition number of the two-level preconditioner $\mathcal{P}(I-H) = \mathcal{P}R_\Gamma^{-\top}S_\Gamma R_\Gamma^{-1}$ *satisfies*

$$\mathbb{E}\left(\sqrt{\kappa_{\text{eff}}\left(\mathcal{P}(I-H)\right)}\right) \leq c_1\sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{k+1}}}, \tag{4.18}$$

where c_1^2 *is independent of the spectrum of* $I-H$ *and can be bounded by a polynomial of degree 3 in* k*.*

Proof. Let $x \in \mathbb{R}^{n_\Gamma}$. Since $u_1,\dots,u_{n_\Gamma}$ form an orthogonal basis of $\mathbb{R}^{n_\Gamma}$, there exists $\alpha_1,\dots,\alpha_{n_\Gamma} \in \mathbb{R}$ such that $x = \sum_{i=1}^{n_\Gamma}\alpha_i u_i$. In [25, Theorem 3.4], Kahl and Rittich show that, if for some positive constant c_K, $\widetilde{U}$ satisfies

$$\|x - \Pi_{\widetilde{U}}x\|_2^2 \leq c_K\frac{\|x\|_{I-H}^2}{\|I-H\|_2}, \tag{4.19}$$

then the effective condition number of $\mathcal{P}(I-H)$ satisfies

$$\kappa_{\text{eff}}\left(\mathcal{P}(I-H)\right) \leq c_K.$$

Let $t \leq k$ and consider

$$\begin{aligned}
\|x - \Pi_{\widetilde{U}}x\|_2 &= \|\sum_{i=1}^{n_\Gamma}\alpha_i u_i - \Pi_{\widetilde{U}}\sum_{i=1}^{n_\Gamma}\alpha_i u_i\|_2 \\
&\leq \|\sum_{i=t+1}^{n_\Gamma}(I-\Pi_{\widetilde{U}})\alpha_i u_i\|_2 + \sum_{i=1}^{t}|\alpha_i|\|u_i - \Pi_{\widetilde{U}}u_i\|_2 \\
&\leq \|\sum_{i=t+1}^{n_\Gamma}\alpha_i u_i\|_2 + \sum_{i=1}^{t}|\alpha_i|\|u_i - \Pi_{\widetilde{U}}u_i\|_2.
\end{aligned}$$

The last inequality is obtained using the fact that $I-\Pi_{\widetilde{U}}$ is an orthogonal projector. Now bound each term on the right separately. We have

$$\begin{aligned}
\|\sum_{i=t+1}^{n_\Gamma}\alpha_i u_i\|_2 &\leq \frac{1}{\sqrt{\lambda_{t+1}}}\|\sum_{i=t+1}^{n_\Gamma}\sqrt{\lambda_{t+1}}\alpha_i u_i\|_2 \leq \frac{1}{\sqrt{\lambda_{t+1}}}\|\sum_{i=t+1}^{n_\Gamma}\sqrt{\lambda_i}\alpha_i u_i\|_2 \\
&\leq \frac{1}{\sqrt{\lambda_{t+1}}}\sum_{i=t+1}^{n_\Gamma}\lambda_i\alpha_i^2 = \frac{1}{\sqrt{\lambda_{t+1}}}\|x-\Pi_{U_t}x\|_{I-H} = \sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}}\frac{\|x-\Pi_{U_t}x\|_{I-H}}{\sqrt{\|I-H\|_2}}.
\end{aligned}$$

∎

From (4.15), $\gamma_{i,k} \leq 1$ for $i = 1, \ldots, t$, thus,

$$\begin{aligned}\sum_{i=1}^{t} |\alpha_i| \|u_i - \Pi_{\widetilde{U}} u_i\|_2 &\leq \sum_{i=1}^{t} |\alpha_i| \gamma_{i,k}^{q+1} c \leq c \gamma_{t,k}^{q+\frac{1}{2}} \sum_{i=1}^{t} |\alpha_i| \sqrt{\gamma_{i,k}} \\ &= c \gamma_{t,k}^{q+\frac{1}{2}} \sqrt{\lambda_{k+1}^{-1} - 1} \sum_{i=1}^{t} |\alpha_i| \frac{1}{\sqrt{\lambda_i^{-1} - 1}} \\ &\leq c \gamma_{t,k}^{q+\frac{1}{2}} \frac{1}{\sqrt{\lambda_{k+1}}} \sum_{i=1}^{t} |\alpha_i| \frac{1}{\sqrt{\lambda_i^{-1} - 1}}.\end{aligned}$$

Assuming that $\lambda_i \leq 1/2$ for $i = 1, \ldots, t$, we have

$$\begin{aligned}\sum_{i=1}^{t} |\alpha_i| \|u_i - \Pi_{\widetilde{U}} u_i\|_2 &\leq \sqrt{2} c \gamma_{t,k}^{q+\frac{1}{2}} \frac{1}{\sqrt{\lambda_{k+1}}} \sum_{i=1}^{t} |\alpha_i| \frac{1}{\sqrt{\lambda_i^{-1}}} \\ &\leq \sqrt{2} c \gamma_{t,k}^{q+\frac{1}{2}} \frac{1}{\sqrt{\lambda_{k+1}}} \sum_{i=1}^{t} |\alpha_i| \sqrt{\lambda_i}.\end{aligned}$$

Using the fact that the l_1 and l_2 norms are equivalent, we have

$$\begin{aligned}\sum_{i=1}^{t} |\alpha_i| \|u_i - \Pi_{\widetilde{U}} u_i\|_2 &\leq c \sqrt{2t} \gamma_{t,k}^{q+\frac{1}{2}} \frac{1}{\sqrt{\lambda_{k+1}}} \sqrt{\sum_{i=1}^{t} \alpha_i^2 \lambda_i} \\ &= c \sqrt{2t} \gamma_{t,k}^{q+\frac{1}{2}} \frac{1}{\sqrt{\lambda_{k+1}}} \|\Pi_{U_t} x\|_{I-H} \\ &= c \sqrt{2t} \gamma_{t,k}^{q+\frac{1}{2}} \sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{k+1}}} \frac{\|\Pi_{U_t} x\|_{I-H}}{\sqrt{\|I - H\|_2}}.\end{aligned}$$

Since $\lambda_k \geq \lambda_t$ we have

$$\sum_{i=1}^{t} |\alpha_i| \|u_i - \Pi_{\widetilde{U}} u_i\|_2 \leq c \sqrt{2t} \gamma_{t,k}^{q+\frac{1}{2}} \sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}} \frac{\|\Pi_{U_t} x\|_{I-H}}{\sqrt{\|I - H\|_2}}.$$

It follows that

$$\begin{aligned}\|x - \Pi_{\widetilde{U}} x\|_2 &\leq \sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}} \frac{\|x - \Pi_{U_t} x\|_{I-H}}{\sqrt{\|I - H\|_2}} + c \sqrt{2t} \gamma_{t,k}^{q+\frac{1}{2}} \sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}} \frac{\|\Pi_{U_t} x\|_{I-H}}{\sqrt{\|I - H\|_2}} \\ &\leq \sqrt{2} \max(c \sqrt{2t} \gamma_{t,k}^{q+\frac{1}{2}}, 1) \sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}} \frac{\|x\|_{I-H}}{\sqrt{\|I - H\|_2}}.\end{aligned}$$

Hence (4.19) is satisfied and we have

$$\kappa_{\text{eff}}\left(\mathcal{P}(I - H)\right) \leq 2 \max(2c^2 t \gamma_{t,k}^{2q+1}, 1) \frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}.$$

Thus,

$$\mathbb{E}\left(\sqrt{\kappa_{\text{eff}}\left(\mathcal{P}(I - H)\right)}\right) \leq \sqrt{2} \max(\mathbb{E}(c) \sqrt{2t} \gamma_{t,k}^{q+\frac{1}{2}}, 1) \sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}}.$$

Since t is chosen arbitrarily between 1 and k we have

$$\mathbb{E}\left(\sqrt{\kappa_{\text{eff}}\left(\mathcal{P}(I-H)\right)}\right) \leq \sqrt{2}\min_{1\leq t\leq k}\left(\max\left(\mathbb{E}(c)\sqrt{2t}\gamma_{t,k}^{q+\frac{1}{2}},1\right)\sqrt{\frac{\lambda_{n_\Gamma}}{\lambda_{t+1}}}\right). \tag{4.20}$$

Because $\mathbb{E}(c)$ can be bounded by a polynomial of degree 1 in k and $\gamma_{t,k} \leq 1$, $\max(4t\gamma_{t,k}^{2q+1}\left(\mathbb{E}(c)\right)^2, 2)$ can be bounded by a polynomial of degree 3 in k independent of the spectrum of $I-H$. ∎

Note that, in practice, when the problem is challenging, a few eigenvalues of $R_\Gamma^{-\top}S_\Gamma R_\Gamma^{-1}$ are close to the origin. This is reflected in a rapid and exponential decay of the values of the entries of $\Lambda^{-1}-I$. Figure 2 depicts the bound obtained in Proposition 4.1 for different values of k and q for problem s3rmt3m3.

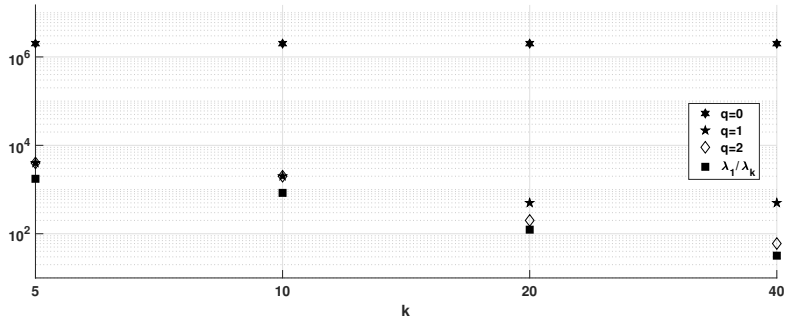

FIG. 2. *Problem s3rmt3m3: Values of the bound (4.20) on* $\left(\mathbb{E}\left(\sqrt{\kappa_{\text{eff}}\left(\mathcal{P}(I-H)\right)}\right)\right)^2$ *for a range of values of* k *and* q.

5. Numerical Experiments. We use 64 subdomains (i.e., A_I is a 64-block diagonal matrix) for each of our test matrices with the exception of one problem. The matrix nd3k is much denser than the others, and we use only two blocks (to reduce the runtime). For comparison purposes, we include results for the Schur complement preconditioners $\widetilde{S}_1$ and $\widetilde{S}_2$ given by (2.14) and (2.17), respectively. As demonstrated in subsection 3.1, the latter is too costly to be practical, however, its performance is the ideal since it guarantees the smallest spectral condition number for a fixed deflation subspace. Therefore, the quality of the Nyström–Schur preconditioner will be measured in terms of how close its performance is to that of $\widetilde{S}_2$ and the reduction in iteration it gives compared to $\widetilde{S}_1$. For a given problem, the right-hand side vector is the same for all the tests: it is generated randomly with entries from the standard normal distribution. The relative convergence tolerance for PCG is 10^{-6}. Unless otherwise specified, the parameters within Nyström's method (Algorithm 2.1) are rank $k = 20$, oversampling $p = 0$, and power iteration $q = 0$. To ensure fair comparisons, the random matrices generated in different runs of the Nyström algorithm use the same seed. We employ the Nyström–Schur variant $\mathcal{M}_2$ (4.10) (recall that its construction does not require the Cholesky factors of A_Γ). The relative convergence tolerance used when solving the SPD system (4.13) is $\varepsilon_{S_I} = 0.1$. This system (4.13) is preconditioned by the block diagonal matrix A_I. We denote by it_{S_I} the number of block PCG iterations required to solve (4.13) during the construction of the Nyström–Schur preconditioners (it is zero for $\widetilde{S}_1$ and $\widetilde{S}_2$), and by it_{PCG} the PCG iteration count

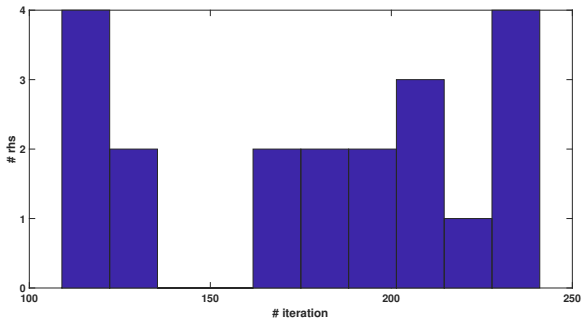

FIG. 3. *Histogram of the PCG iteration counts for (4.13) for problem bcsstk38. The number of right hand sides for which the iteration count is between* $[k, k+10)$, $k = 100, \ldots, 240$, *is given.*

	Classic		Block	
Identifier	*iters*	it_{PCG}	*iters*	it_{PCG}
bcsstk38	238	186	46	173
el2d	549	261	72	228
el3d	95	56	24	52
msc10848	203	194	47	166
nd3k	294	191	32	178
s3rmt3m3	403	157	37	98

TABLE 5

A comparison of the performance of classic and block PCG. iters denotes the iteration count for solving (4.13) (details in the text) and it_{PCG} *is the iteration count for solving (2.13).*

for solving (2.13). The total number of iterations is $it_{\mathrm{total}} = it_{S_I} + it_{\mathrm{PCG}}$. We use the code [1] to generate the numerical experiments.

5.1. Linear system with S_I. We start by considering how to efficiently compute an approximate solution of (4.13).

5.1.1. Block and classic CG. The system (4.13) has $k + p$ right hand sides. The number of iterations required by PCG to solve each right hand side is different and the variation can be large; this is illustrated in Figure 3 for problem bcsstk38. Here we report the number of right hand sides for which the iteration count lies in the interval $[k, k + 10)$, $k = 100, \ldots, 240$. For example, there are 4 right hand sides for which the count is between 110 and 119. Similar behaviour was observed for our other test problems.

Table 5 reports the iteration counts for the classical PCG method and the breakdown-free block PCG method [21, 35]. For PCG, *iters* is the largest PCG iteration count over the $k + p$ right hand sides. For the block method, $iters = it_{S_I}$ is the number of block PCG iterations. As expected from the theory, the block method significantly reduces the (maximum) iteration count. For our examples, it also leads to a modest reduction in the iteration count it_{PCG} for solving (2.13).

5.1.2. Impact of tolerance ε_{S_I}. We now study the impact of the convergence tolerance ε_{S_I} used when solving (4.13) on the quality of the Nyström–Schur preconditioner. In Table 6, we present results for three test problems that illustrate the (slightly) different behaviors we observed. The results demonstrate numerically

		$\mathcal{M}_2$		$\widetilde{S}_1$	$\widetilde{S}_2$
Identifier	ε_{S_I}	it_{S_I}	it_{PCG}		
el2d	0.8	1	500+	914	231
	0.5	68	228		
	0.3	70	228		
	0.1	72	228		
	0.01	78	228		
el3d	0.8	1	173	174	37
	0.5	2	171		
	0.3	22	52		
	0.1	24	52		
	0.01	27	52		
nd3k	0.8	32	178	603	143
	0.5	32	178		
	0.3	32	178		
	0.1	32	178		
	0.01	33	178		

TABLE 6
The effects of the convergence tolerance ε_{S_I} on the quality of the Nyström–Schur preconditioner.

Identifier	$\mathcal{M}_1$	$\mathcal{M}_{1\text{-A-DEF}}$	$\mathcal{M}_2$	$\mathcal{M}_{2\text{-A-DEF}}$	$\mathcal{M}_3$	$\mathcal{M}_{3\text{-A-DEF}}$	$\widetilde{S}_1$	$\widetilde{S}_2$
bcsstk38	218	218	219	219	360	313	584	122
el2d	266	267	300	300	282	282	914	231
el3d	73	72	76	75	78	76	174	37
msc10848	206	205	213	211	216	222	612	116
nd3k	205	205	210	210	211	211	603	143
s3rmt3m3	127	127	135	134	161	153	441	70

TABLE 7
Comparison of it_{total} for the variants of the Nyström–Schur preconditioner and $\widetilde{S}_1$ and $\widetilde{S}_2$. $\varepsilon_{S_I} = 0.1$.

that a large tolerance can be used without affecting the quality of the preconditioner. Indeed, using $\varepsilon_{S_I} = 0.3$ leads to a preconditioner whose efficiency is close to that of the ideal (but impractical) two-level preconditioner $\widetilde{S}_2$. The use of a large ε_{S_I} to limit it_{S_I} is crucial in ensuring low construction costs for the Nyström–Schur preconditioners.

5.2. Type of preconditioner. We next compare the performances of the variants $\mathcal{M}_i$ and $\mathcal{M}_{\text{i-A-DEF}}$ $(i = 1, 2, 3)$ of the Nyström–Schur preconditioner presented in section 4. In Table 7, we report the total iteration count it_{total}. All the variants have similar behaviors and have a significantly smaller count than the one-level preconditioner $\widetilde{S}_1$.

5.3. Varying the rank and the oversampling parameter. We now look at varying the rank k within the Nyström algorithm and demonstrate numerically that the efficiency of the preconditioner is robust with respect to the oversampling parameter p. For problem s3rmt3m3, Table 8 compares the iteration counts for $\mathcal{M}_2$ with that of the ideal two-level preconditioner $\widetilde{S}_2$ for k ranging from 5 to 320. For $\widetilde{S}_1$, the iteration count is 441. This demonstrates the effectiveness of the Nyström–Schur preconditioner in reducing the iteration count. Increasing the size of the deflation subspace (the rank k) steadily reduces the iteration count required to solve the S_I

	k	5	10	20	40	80	160	320
$\mathcal{M}_2$	it_{S_I}	97	57	37	23	16	11	8
	$it_{\rm PCG}$	244	203	98	53	30	20	14
$\widetilde{S}_2$	$it_{\rm PCG}$	212	153	70	37	22	13	9

TABLE 8
Problem s3rmt3m3: Impact of the rank k on the iteration counts ($p = 0$).

p	0	5	10	20	40
it_{S_I}	37	31	28	23	20
$it_{\rm PCG}$	98	86	79	77	74

TABLE 9
Problem s3rmt3m3: Impact of the oversampling parameter p on the iteration counts ($k = 20$).

system (4.13). For the same test example, Table 9 presents the iteration counts for a range of values of the oversampling parameter p (here $k = 20$). We observe that the counts are relatively insensitive to p but, as p increases, $it_{\rm PCG}$ reduces towards the lower bound of 70 PCG iterations required by $\widetilde{S}_2$. Similar behavior was noticed for our other test examples. Although increasing k and p improves the efficiency of the Nyström–Schur preconditioner, this comes with extra costs during both the construction of the preconditioner and its application. Nevertheless, the savings from the reduction in the iteration count and the efficiency in solving block linear systems of equations for moderate block sizes (for example, $k = 40$) typically outweigh the increase in construction costs.

5.4. Comparisons with incomplete Cholesky factorization preconditioners. Finally, we compare the Nyström–Schur preconditioner with two incomplete Cholesky factorization preconditioners applied to original system. The first is the Matlab variant `ichol` with the global diagonal shift set to 0.1 and default values for other parameters and the second is the Matlab interface to the incomplete Cholesky (IC) factorization preconditioner `HSL_MI28` [39] from the HSL library [20] using the default parameter settings. IC preconditioners are widely used but their construction is often serial, potentially limiting their suitability for very large problems (see [19] for an IC preconditioner that can be parallelised). In terms of iteration counts, the Nyström–Schur and the `HSL_MI28` preconditioners are clearly superior to the simple `ichol` preconditioner, with neither consistently offering the best performance. Figure 4 presents the residual norm history for PCG. This is confirmed by the results in the Appendix for our large test set. The residual norm for $\mathcal{M}_2$ decreases monotonically while for the IC preconditioners we observe oscillatory behaviour.

Because our implementation of the Nyström–Schur preconditioner is in Matlab, we are not able to provide performance comparisons in terms of computation times. Having demonstrated the potential of our two-level Nyström–Schur preconditioner, one of our objectives for the future is to develop an efficient (parallel) implementation in Fortran that will be included within the HSL library. This will allow users to test out the preconditioner and to assess the performance of both constructing and applying the preconditioner. Our preliminary work on this is encouraging.

Identifier	$\mathcal{M}_2$		HSL_MI28	ichol
	it_{S_I}	it_{PCG}		
bcsstk38	46	173	593	2786
ela2d	72	228	108	2319
ela3d	24	52	36	170
msc10848	47	166	145	784
nd3k	32	178	102	1231
s3rmt3m3	37	98	610	2281

TABLE 10

PCG iteration counts for the Nyström–Schur preconditioner $\mathcal{M}_2$ (with $k = 20$) and the IC preconditioners `HSL_MI28` *and* `ichol`.

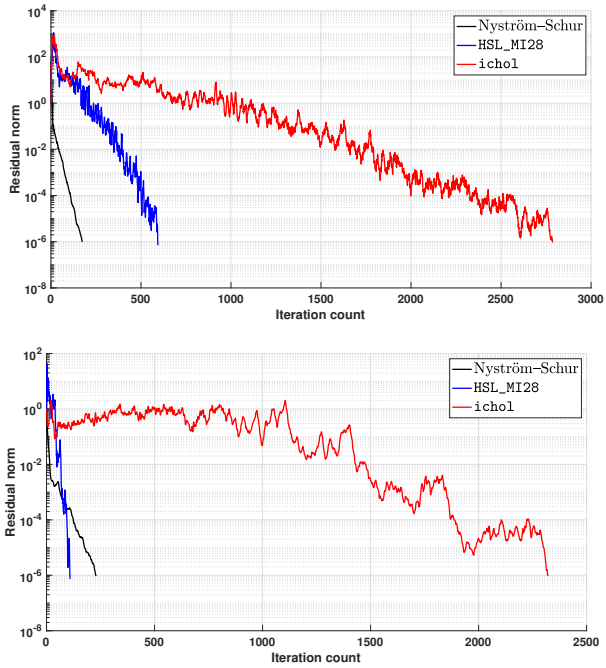

FIG. 4. *PCG residual norm history for test examples bcsstk38 (top) and ela2d (bottom).*

6. Concluding comments. In this paper, we have investigated using randomized methods to construct efficient and robust preconditioners for use with CG to solve large-scale SPD linear systems. The approach requires an initial ordering to doubly bordered block diagonal form and then uses a Schur complement approximation. We have demonstrated that by carefully posing the approximation problem we can apply randomized methods to construct high quality preconditioners, which gives an improvement over previously proposed methods that use low rank approximation strategies. We have presented a number of variants of our new Nyström–Schur preconditioner. During the preconditioner construction, we must solve a smaller linear system with multiple right-hand sides. Our numerical experiments have shown that a small number of iterations of block CG are needed to obtain an approximate solution that is sufficient to construct an effective preconditioner.

Currently, the construction and application of our Nyström–Schur preconditioners requires the solution of linear systems with the block matrix A_Γ (2.9). Given the promising results presented in this paper, in the future we plan to investigate employing a recursive approach, following ideas given in [49]. This will only require the solution of systems involving a much smaller matrix and will lead to a practical approach for very large-scale SPD systems. A parallel implementation of the preconditioner will also be developed.

Appendix A. Extended numerical experiments. Here we present results for a larger test set. The problems are given in Table 11. We selected all the SPD matrices in the SuiteSparse Collection with n lying between 5K and 100K, giving us a set of 71 problems. For each problem, we ran PCG with the $\widetilde{S}_1$, $\mathcal{M}_2$, $\widetilde{S}_2$ and `HSL_MI28` preconditioners. In all the tests, we use 64 subdomains. For $\mathcal{M}_2$, we used $k = 20$ and set $p = q = 0$. Iteration counts are given in the table, whilst performance profiles [6] are presented in Figure 5. In recent years, performance profiles have become a popular and widely used tool for providing objective information when benchmarking algorithms. The performance profile takes into account the number of problems solved by an algorithm as well as the cost to solve it. It scales the cost of solving the problem according to the best solver for that problem. In our case, the performance cost is the iteration count (for $\mathcal{M}_2$, we sum the counts it_{S_I} and it_{PCG}). Note that we do not include $\widetilde{S}_2$ in the performance profiles because it is an ideal but impractical two-level preconditioner and, as such, it always outperforms $\mathcal{M}_2$. The performance profile shows that on the problems where $\widetilde{S}_1$ struggles, there is little to choose between the overall quality of $\mathcal{M}_2$ and `HSL_MI28`.

	$\widetilde{S}_1$	$\mathcal{M}_2$		$\widetilde{S}_2$	HSL_MI28	$\kappa(A)$
Identifier		it_{S_I}	it_{PCG}			
aft01	118	19	45	31	17	9e+18
apache1	667	122	291	192	72	3e+06
bcsstk17	349	46	55	48	59	1e+10
bcsstk18	136	40	77	45	26	6e+11
bcsstk25	†	92	660	453	254	1e+13
bcsstk36	451	64	214	169	†	1e+12
bcsstk38	584	46	171	122	593	6e+16
bodyy6	182	53	163	129	5	9e+04
cant	†	57	228	396	933	5e+10
cfd1	209	30	72	50	274	1e+06
consph	185	47	177	136	50	3e+07
gridgena	426	90	377	298	66	6e+05
gyro	†	55	346	518	319	4e+09
gyro_k	†	55	346	518	319	3e+09
gyro_m	165	16	34	22	17	1e+07
m_t1	867	85	247	187	‡	3e+11
minsurfo	15	3	15	13	3	8e+01
msc10848	612	47	168	116	145	3e+10
msc23052	479	69	220	175	‡	1e+12
nasasrb	1279	135	496	421	†	1e+09
nd3k	1091	56	301	230	102	5e+07
nd6k	1184	108	325	248	116	6e+07
oilpan	647	67	122	72	507	4e+09
olafu	1428	69	489	757	557	2e+12
pdb1HYS	869	89	83	274	483	2e+12
vanbody	†	287	1106	769	‡	4e+03
ct20stif	1296	90	232	281	†	2e+14
nd12k	1039	155	337	265	111	2e+08
nd24k	1093	165	386	268	120	2e+08
s1rmq4m1	154	19	50	32	33	5e+06
s1rmt3m1	192	24	59	39	18	3e+08
s2rmq4m1	231	28	54	41	39	4e+08
s2rmt3m1	260	31	64	45	33	3e+11
s3dkq4m2	†	148	339	236	610	6e+11
s3dkt3m2	†	164	338	270	1107	3e+10
s3rmq4m1	356	31	80	58	472	4e+10
s3rmt3m1	434	36	101	64	413	4e+10
s3rmt3m3	441	37	101	70	610	3e+00
ship_001	1453	367	600	368	1177	6e+09
smt	399	59	112	72	95	1e+09
thermal1	169	30	62	47	30	4e+01
Pres_Poisson	92	13	29	19	32	3e+06
crankseg_1	92	16	49	33	34	9e+18
crankseg_2	89	17	47	32	38	8e+06
Kuu	81	16	44	31	10	3e+04
bodyy5	72	19	67	57	4	9e+03
Dubcova2	62	11	32	23	14	1e+04
cbuckle	55	9	51	39	47	7e+07
fv3	50	12	31	21	8	4e+03
Dubcova1	39	8	24	15	7	2e+03
bodyy4	34	8	29	24	4	1e+03
jnlbrng1	22	4	21	19	4	1e+02
bundle1	13	3	8	5	5	1e+04
t2dah_e	12	3	12	11	3	3e+07
obstclae	12	3	12	12	3	4e+01
torsion1	12	3	12	12	3	8e+03
wathen100	12	3	12	11	3	2e+07
wathen120	12	3	12	11	3	2e+07
fv1	7	2	7	7	3	1e+01
fv2	7	2	7	7	3	1e+01
shallow_water2	7	40	7	7	3	3e+12
shallow_water1	5	20	5	5	2	1e+01
Muu	6	1	6	6	2	1e+02
qa8fm	6	1	6	6	2	1e+02
crystm02	6	1	6	5	2	4e+02
crystm03	6	1	6	5	2	4e+02
finan512	5	1	5	5	3	9e+01
ted_B_unscaled	3	1	3	4	2	4e+05
ted_B	2	1	3	3	2	2e+11
Trefethen_20000b	3	1	2	2	3	1e+05
Trefethen_20000	4	1	2	2	3	2e+05

TABLE 11

PCG iteration counts for SPD matrices from the SuiteSparse Collection with n ranging between 5K and 100K.

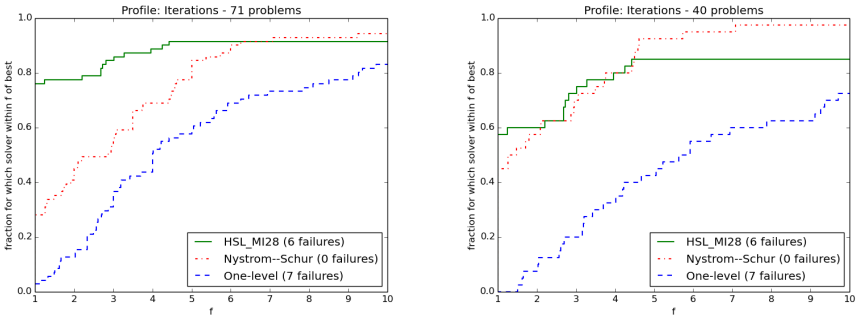

FIG. 5. *Iteration count performance profile for the large test set. The 40 problems used in the right hand plot are the subset for which the $\widetilde{S}_1$ (one-level) iteration count exceeded 100.*

REFERENCES

[1] H. AL DAAS, *haldaas/Nystrom–Schur-Preconditioner: version reproducing paper numerical experiments*, June 2021, https://doi.org/10.5281/zenodo.4957301.

[2] H. AL DAAS AND L. GRIGORI, *A class of efficient locally constructed preconditioners based on coarse spaces*, SIAM Journal on Matrix Analysis and Applications, 40 (2019), pp. 66–91.

[3] H. AL DAAS, L. GRIGORI, P. JOLIVET, AND P.-H. TOURNIER, *A multilevel Schwarz preconditioner based on a hierarchy of robust coarse spaces*, SIAM Journal on Scientific Computing, (2021), pp. A1907–A1928.

[4] H. AL DAAS, P. JOLIVET, AND J. A. SCOTT, *A robust algebraic domain decomposition preconditioner for sparse normal equations*, 2021, https://arxiv.org/abs/2107.09006.

[5] T. A. DAVIS AND Y. HU, *The University of Florida sparse matrix collection*, ACM Transactions on Mathematical Software, 38 (2011), pp. 1–28.

[6] E. D. DOLAN AND J. J. MORÉ, *Benchmarking optimization software with performance profiles*, Mathematical Programming, 91 (2002), pp. 201–213.

[7] V. DOLEAN, P. JOLIVET, AND F. NATAF, *An introduction to domain decomposition methods*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2015. Algorithms, theory, and parallel implementation.

[8] Z. DOSTÁL, *Conjugate gradient method with preconditioning by projector*, International Journal of Computer Mathematics, 23 (1988), pp. 315–323.

[9] I. S. DUFF, A. M. ERISMAN, AND J. K. REID, *Direct Methods for Sparse Matrices, Second Edition*, Oxford University Press, London, 2017.

[10] J. FRANK AND C. VUIK, *On the construction of deflation-based preconditioners*, SIAM Journal on Scientific Computing, 23 (2001), pp. 442–462.

[11] A. GAUL, M. H. GUTKNECHT, J. LIESEN, AND R. NABBEN, *A framework for deflated and augmented Krylov subspace methods*, SIAM Journal on Matrix Analysis and Applications, 34 (2013), pp. 495–518.

[12] A. GITTENS AND M. W. MAHONEY, *Revisiting the Nyström method for improved large-scale machine learning*, J. Mach. Learn. Res., 17 (2016), pp. 3977–4041.

[13] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, third ed., 1996.

[14] L. GRIGORI, F. NATAF, AND S. YOUSEF, *Robust algebraic Schur complement preconditioners based on low rank corrections*, Research Report RR-8557, INRIA, July 2014, https://hal.inria.fr/hal-01017448.

[15] M. H. GUTKNECHT, *Deflated and augmented Krylov subspace methods: A framework for deflated BiCG and related solvers*, SIAM Journal on Matrix Analysis and Applications, 35 (2014), pp. 1444–1466.

[16] N. HALKO, P.-G. MARTINSSON, AND J. A. TROPP, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), pp. 217–288.

[17] F. HECHT, *New development in freefem++*, Journal of Numerical Mathematics, 20 (2012), pp. 251–265.

[18] N. J. HIGHAM AND T. MARY, *A new preconditioner that exploits low-rank approximations to*

factorization error, SIAM Journal on Scientific Computing, 41 (2019), pp. A59–A82.
[19] J. Hook, J. Scott, F. Tisseur, and J. Hogg, *A max-plus apporach to incomplete Cholesky factorization preconditioners*, SIAM Journal on Scientific Computing, 40 (2018), pp. A1987–A2004.
[20] *HSL. A collection of Fortran codes for large-scale scientific computation*, 2018. http://www.hsl.rl.ac.uk.
[21] H. Ji and Y. Li, *A breakdown-free block conjugate gradient method*, BIT Numerical Mathematics, 57 (2017), pp. 379–403.
[22] T. B. Jönsthövel, M. B. van Gijzen, C. Vuik, C. Kasbergen, and A. Scarpas, *Preconditioned conjugate gradient method enhanced by deflation of rigid body modes applied to composite materials*, Computer Modeling in Engineering & Sciences, 47 (2009), pp. 97–118.
[23] T. B. Jönsthövel, M. B. van Gijzen, C. Vuik, and A. Scarpas, *On the use of rigid body modes in the deflated preconditioned conjugate gradient method*, SIAM Journal on Scientific Computing, 35 (2013), pp. B207–B225.
[24] E. F. Kaasschieter, *Preconditioned conjugate gradients for solving singular systems*, Journal of Computational and Applied Mathematics, 24 (1988), pp. 265–275.
[25] K. Kahl and H. Rittich, *The deflated conjugate gradient method: Convergence, perturbation and accuracy*, Linear Algebra and its Applications, 515 (2017), pp. 111–129.
[26] G. Karypis and V. Kumar, *METIS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices*, Technical Report 97-061, University of Minnesota, Department of Computer Science and Army HPC Research Center, 1997.
[27] R. Li, Y. Xi, and Y. Saad, *Schur complement-based domain decomposition preconditioners with low-rank corrections*, Numerical Linear Algebra with Applications, 23 (2016), pp. 706–729.
[28] P.-G. Martinsson and J. A. Tropp, *Randomized numerical linear algebra: Foundations and algorithms*, Acta Numerica, 29 (2020), pp. 403–572.
[29] *METIS - serial graph partitioning and fill-reducing matrix ordering*, 2020. http://glaros.dtc.umn.edu/gkhome/metis/metis/overview.
[30] R. Nabben and C. Vuik, *A comparison of abstract versions of deflation, balancing and additive coarse grid correction preconditioners*, Numerical Linear Algebra with Applications, 15 (2008), pp. 355–372.
[31] Y. Nakatsukasa, *Fast and stable randomized low-rank matrix approximation*, 2020, https://arxiv.org/abs/2009.11392.
[32] F. Nataf, H. Xiang, V. Dolean, and N. Spillane, *A coarse space construction based on local Dirichlet-to-Neumann maps*, SIAM Journal on Scientific Computing, 33 (2011), pp. 1623–1642.
[33] R. A. Nicolaides, *Deflation of conjugate gradients with applications to boundary value problems*, SIAM J. on Numerical Analysis, 24 (1987), pp. 355–365.
[34] E. J. Nyström, *Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben*, Acta Mathematica, 54 (1930), pp. 185–204.
[35] D. P. O'Leary, *The block conjugate gradient algorithm and related methods*, Linear Algebra and its Applications, 29 (1980), pp. 293–322.
[36] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd ed., 2003.
[37] A. K. Saibaba, *Randomized subspace iteration: Analysis of canonical angles and unitarily invariant norms*, SIAM Journal on Matrix Analysis and Applications, 40 (2019), pp. 23–48.
[38] J. A. Scott, *A parallel frontal solver for finite element applications*, International J. of Numerical Methods in Engineering, 50 (2001), pp. 1131–1144.
[39] J. A. Scott and M. Tůma, `HSL_MI28`*: An efficient and robust limited-memory incomplete Cholesky factorization code*, ACM Transactions on Mathematical Software, 40 (2014), pp. 24:1–19.
[40] N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, and R. Scheichl, *Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps*, Numerische Mathematik, 126 (2014), pp. 741–770.
[41] N. Spillane and D. Rixen, *Automatic spectral coarse spaces for robust finite element tearing and interconnecting and balanced domain decomposition algorithms*, International Journal for Numerical Methods in Engineering, 95 (2013), pp. 953–990.
[42] G. W. Stewart, *A Krylov–Schur algorithm for large eigenproblems*, SIAM Journal on Matrix Analysis and Applications, 23 (2002), pp. 601–614.

[43] J. M. Tang, S. P. MacLachlan, R. Nabben, and C. Vuik, *A comparison of two-level preconditioners based on multigrid and deflation*, SIAM Journal on Matrix Analysis and Applications, 31 (2010), pp. 1715–1739.
[44] J. M. Tang, R. Nabben, C. Vuik, and Y. A. Erlangga, *Comparison of two-level preconditioners derived from deflation, domain decomposition and multigrid methods*, Journal of Scientific Computing, 39 (2009), pp. 340–370.
[45] C. Vuik, A. Segal, and J. A. Meijerink, *An efficient preconditioned CG method for the solution of a class of layered problems with extreme contrasts in the coefficients*, Journal of Computational Physics, 152 (1999), pp. 385–403.
[46] C. Vuik, A. Segal, J. A. Meijerink, and G. T. Wijma, *The construction of projection vectors for a deflated ICCG method applied to problems with extreme contrasts in the coefficients*, Journal of Computational Physics, 172 (2001), pp. 426–450.
[47] C. K. I. Williams and M. Seeger, *Using the Nyström method to speed up kernel machines*, in Advances in Neural Information Processing Systems 13, T. K. Leen, T. G. Dietterich, and V. Tresp, eds., MIT Press, 2001, pp. 682–688.
[48] D. Woodruff, *Sketching as a Tool for Numerical Linear Algebra*, Foundations and Trends(r) in Theoretical Computer Science Series, Now Publishers, 2014.
[49] Y. Xi, R. Li, and Y. Saad, *An algebraic multilevel preconditioner with low-rank corrections for sparse symmetric matrices*, SIAM Journal on Matrix Analysis and Applications, 37 (2016), pp. 235–259.
[50] Q. Zheng, Y. Xi, and Y. Saad, *A power Schur complement low-rank correction preconditioner for general sparse linear systems*, SIAM Journal on Matrix Analysis and Applications, 42 (2021), pp. 659–682.