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, preconditioner, symmetric positive definite, Schur complement

AMS subject classifications. 65F08, 65F50, 65F55

DOI. 10.1137/21M139548X

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

\[ Ax = b, \]

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, nonlinear 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

*Submitted to the journal’s Methods and Algorithms for Scientific Computing section January 29, 2021; accepted for publication (in revised form) July 19, 2021; published electronically November 11, 2021.

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

‡STFC Rutherford Appleton Laboratory, Harwell Campus, Didcot, Oxfordshire, OX11 0QX, UK, and School of Mathematical, Physical and Computational Sciences, University of Reading, Reading, RG6 6AQ, UK (jennifer.scott@stfc.ac.uk).
where $x_\star$ is the exact solution, $x_0$ is the initial guess, $\| \cdot \|_A$ is the $A$-norm, and $\kappa(A) = \lambda_{\text{max}}/\lambda_{\text{min}}$ is the spectral condition number ($\lambda_{\text{max}}$ and $\lambda_{\text{min}}$ denote the largest and 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 $P$ to (1.1), where $P \in \mathbb{R}^{n \times n}$ is chosen so that the spectral condition number of $PA$ is small and applying $P$ 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, [35]. These are referred to as one-level or traditional preconditioners [7, 42]. In general, algebraic preconditioners bound the largest eigenvalues of $PA$ 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, 32] and has been discussed and used by a number of researchers [2, 3, 10, 14, 22, 23, 27, 31, 39, 40, 44, 45]. 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, 29, 42, 43]. Recently, a two-level Schur complement preconditioner based on the power series approximation was proposed in [49].

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, 30] 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 $2 \times 2$ 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 $P$, 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, 33, 46, 47], 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 analyzing the eigenvalue problems that must be solved. We show that these difficulties are mainly associated with the clustering of eigenvalues near 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 SPD 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 pseudoinverse of a matrix $C$ is denoted by $C^+$, 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. $P$ (with or without a subscript) denotes a (deflation) preconditioner. $M$ (with or without a subscript) denotes a two-level (deflation) preconditioner. Matrices with an upper symbol, such as $\tilde{Z}$, $\hat{Z}$, and $\check{Z}$, denote approximations of the matrix $Z$. Euler’s constant is denoted by $e$.

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, and $n_\Gamma$ is the order of the Schur complement \cite{211}. SSMC refers to SuiteSparse Matrix Collection \cite{5}. FF++ refers to FreeFem++ \cite{17}.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>$n$</th>
<th>$nnz(A)$</th>
<th>$\kappa(A)$</th>
<th>$n_\Gamma$</th>
<th>2D/3D</th>
<th>Application</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcsstk38</td>
<td>8,032</td>
<td>355,460</td>
<td>5.5e+16</td>
<td>2,589</td>
<td>2D</td>
<td>Structural problem</td>
<td>SSMC</td>
</tr>
<tr>
<td>elad2d</td>
<td>45,602</td>
<td>543,600</td>
<td>1.5e+8</td>
<td>4,288</td>
<td>2D</td>
<td>Elasticity problem</td>
<td>FF++</td>
</tr>
<tr>
<td>elad3d</td>
<td>9,438</td>
<td>312,372</td>
<td>4.5e+5</td>
<td>4,658</td>
<td>3D</td>
<td>Elasticity problem</td>
<td>FF++</td>
</tr>
<tr>
<td>mnc10848</td>
<td>10,848</td>
<td>1,229,776</td>
<td>1.0e+10</td>
<td>4,440</td>
<td>3D</td>
<td>Structural problem</td>
<td>SSMC</td>
</tr>
<tr>
<td>ndlk</td>
<td>9,000</td>
<td>3,279,690</td>
<td>1.6e+7</td>
<td>1,785</td>
<td>3D</td>
<td>Not available</td>
<td>SSMC</td>
</tr>
<tr>
<td>sbrnt3zm3</td>
<td>5,357</td>
<td>207,123</td>
<td>2.4e+10</td>
<td>2,058</td>
<td>2D</td>
<td>Structural problem</td>
<td>SSMC</td>
</tr>
</tbody>
</table>
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 symmetric positive semidefinite (SPSD) matrix $B$ is defined to be

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

We observe that there are a large number of variants based on different choices of $G$ (for example, [16, 28, 30]). For $q \geq 0$, the $q$-power iteration Nyström method is obtained by choosing

$$G = B^q \Omega$$ (2.2)

for a given (random) starting matrix $\Omega$. 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. Here and elsewhere EVD denotes eigenvalue decomposition. Nyström’s method 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, 30]. 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.

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 $k$th eigenvalues. If the ratio is approximately equal to one, $q$ must be large to obtain a good approximation [36].

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

$$\mathcal{P}Ax = \mathcal{P}b, \quad \mathcal{P} \in \mathbb{R}^{n \times n}.$$ (2.3)

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 $k > 0$, and an oversampling parameter $p \geq 0$ such that $k, p \ll n$, and a threshold $\epsilon$.

**Output:** $\tilde{B}_k = \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^\top \approx B$, where $\tilde{U}_k$ is orthonormal and $\tilde{\Sigma}_k$ is diagonal with nonnegative 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 $\epsilon$.
6: Set $T = RV_1 D_1^{-1} (RV_1)^\top$.
7: Compute the EVD $T = WEW^\top$.
8: Set $\tilde{U} = QW$, $\tilde{U}_k = \tilde{U}(;1:k)$, $\tilde{\Sigma} = E(1:k,1:k)$, and $\tilde{B}_k = \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^\top$. 

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.
Given a projection subspace matrix $Z \in \mathbb{R}^{n \times k}$ of full rank and $k \ll n$, define the nonsingular matrix $E = Z^T AZ \in \mathbb{R}^{k \times k}$ and the matrix $Q = Z E^{-1} Z^T \in \mathbb{R}^{n \times n}$. The deflation preconditioner $P_{\text{def}} \in \mathbb{R}^{n \times n}$ is defined to be [10]

$$
(2.4) \quad P_{\text{def}} = I - AQ.
$$

It is straightforward to show that $P_{\text{def}}$ is a projection matrix and $P_{\text{def}} A$ has $k$ zero eigenvalues (see [43] for basic properties of $P_{\text{def}}$). To solve (1.1), we write

$$
(2.5) \quad P_{\text{def}} Ay = P_{\text{def}} b;
$$

then (due to the identity $AP_{\text{def}}^T = P_{\text{def}} A$) we have that $P_{\text{def}}^T y = P_{\text{def}}^T x$. We therefore obtain the unique solution $x = Q b + P_{\text{def}}^T x$. 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. [43] illustrate that rounding errors can result in the erratic and slow convergence of CG using $P_{\text{def}}$. They thus also consider an adapted deflation preconditioner,

$$
(2.6) \quad P_{\text{a-def}} = I - QA + Q,
$$

that combines $P_{\text{def}}^T$ with $Q$. In exact arithmetic, both $P_{\text{def}}$ and $P_{\text{a-def}}$ used with CG generate the same iterates. However, numerical experiments [43] show that the latter is more robust and leads to better numerical behavior of CG.\footnote{In [43], $P_{\text{def}}$ and $P_{\text{a-def}}$ are termed $P_{\text{def}1}$ and $P_{\text{a-def}2}$, respectively.}

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, $P_{\text{ideal}}$, the deflation subspace is the invariant subspace spanned by the eigenvectors associated with the smallest eigenvalues. To demonstrate how $P_{\text{ideal}}$ 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 $E = Z^T AZ$ is equal to $\Lambda_k = \text{diag}(\lambda_1, \ldots, \lambda_k)$ and the preconditioned matrix is given by

$$
P_{\text{ideal}} A = A - Z_k \Lambda_k Z_k^T.
$$

Since $Z_k$ is orthonormal and its columns span an invariant subspace, the spectrum of $P_{\text{ideal}} A$ is $\{\lambda_n, \ldots, \lambda_{k+1}, 0\}$. Starting with $x_0$ such that $Z_k^T r_0 = 0$ ($r_0$ is the initial residual), for $l \geq 0$, $Z_k^T (P_{\text{ideal}} A)^l r_0 = 0$ and $Z_k^T A^l r_0 = 0$. 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
\[ \kappa_{\text{eff}}(P_{\text{ideal}}A) = \frac{\lambda_n}{\lambda_{k+1}}. \]

Using similar computations, the ideal adapted deflated system is given by
\[ P_{A-\text{ideal}} = A - Z_k\Lambda_k^{-1}Z_k^\top + Z_kZ_k^\top. \tag{2.7} \]
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
\[ \kappa_{\text{eff}}(P_{A-\text{ideal}}A) = \max\left\{1, \frac{\lambda_n}{\lambda_{k+1}}\right\}. \]

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 $Z_k \approx Z_k$ and present an upper bound on the corresponding effective spectral condition number of the deflated matrix $\kappa(PA)$. Their bound [25, Proposition 4.3] (which depends on $\kappa(A)$, $\kappa_{\text{eff}}(P_{\text{ideal}}A)$, and the largest principal angle $\theta$ between $Z_k$ and $Z_k$) is given by
\[ \kappa(PA) \leq \left( \sqrt{\kappa(A)} \sin \theta + \sqrt{\kappa_{\text{eff}}(P_{\text{ideal}}A)} \right)^2, \tag{2.8} \]
where $\sin \theta = \|Z_kZ_k^\top - Z_kZ_k^\top\|_2$.

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, Nataf, and Yousef [14] and, independently, Li, Xi, and Saad [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]), $A$ can be symmetrically permuted to the $2 \times 2$ doubly bordered block diagonal form
\[ P^\top AP = \begin{pmatrix} A_I & A_{II} \\ A_{II}^\top & A_{\Gamma} \end{pmatrix}, \tag{2.9} \]
where $A_I \in \mathbb{R}^{n_I \times n_I}$ is a block diagonal matrix, $A_{\Gamma} \in \mathbb{R}^{n_{\Gamma} \times n_{\Gamma}}$, $A_{II} \in \mathbb{R}^{n_{II} \times n_{II}}$, and $A_{II} = A_{II}^\top$. 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:
\[ A = \begin{pmatrix} I & A_{II} \\ A_{II}A_I^{-1} & I \end{pmatrix} \begin{pmatrix} A_I & A_{II} \\ A_{II} & S_\Gamma \end{pmatrix} \begin{pmatrix} I & \cdot \cdot \cdot \\ A_{II}^{-1} & I \end{pmatrix}, \tag{2.10} \]
where
\[ S_\Gamma = A_{\Gamma} - A_{II}A_I^{-1}A_{II} \tag{2.11} \]
is the Schur complement of $A$ with respect to $A_{Γ}$. Provided the blocks within $A_{Γ}$ are small, they can be factorized cheaply in parallel using a direct algorithm (see, for example, [37]), and thus we assume that solving linear systems with $A_{Γ}$ 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 $A_{Γ}$), 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

$$
M^{-1} = \begin{pmatrix} I & -A_{Γ}^{-1}A_{ΓΓ}^{-1} \\ A_{Γ}^{-1}I & \tilde{S}^{-1} \end{pmatrix} \begin{pmatrix} I \\ -A_{ΓΓ}^{-1}I \end{pmatrix},
$$

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

$$
M^{-1}A = \begin{pmatrix} I & A_{Γ}^{-1}A_{ΓΓ}(I - \tilde{S}^{-1}S_{Γ}) \end{pmatrix},
$$

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

$$
S_{Γ}w = f.
$$

Consider the first-level preconditioner obtained by setting

$$
\tilde{S}_{Γ}^{-1} := A_{Γ}^{-1}.
$$

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 [48]). Let the eigenvalues of the generalized eigenvalue problem

$$
S_{Γ}z = λS_{Γ}z
$$

be $λ_{n_Γ} ≥ · · · ≥ λ_{1} > 0$. From (2.11), $λ_{n_Γ} ≤ 1$, and so

$$
κ(\tilde{S}_{Γ}^{-1}S_{Γ}) = \frac{λ_{n_Γ}}{λ_{1}} ≤ \frac{1}{λ_{1}}.
$$

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 $\tilde{S}_{Γ}^{-1}S_{Γ}$. Let $Λ_{k} = diag\{λ_{1}, \ldots , λ_{k}\}$, and let $Z_{k} ∈ \mathbb{R}^{n_Γ×k}$ 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

$$
A_{Γ} = R_{Γ}^{T}R_{Γ},
$$

and define

$$
\tilde{S}_{2}^{-1} := A_{Γ}^{-1} + Z_{k}(A_{Γ}^{-1} - I)Z_{k}^{T}.
$$
where the SPSD matrix $\Sigma_{2}^{-1}$ is given by (2.20). Since $I - H = R_{1}^{-1} S_{1}^{-1} R_{1}^{-1}$ is SPD, the eigenvalues $\sigma_1 \geq \cdots \geq \sigma_{n_{H}}$ of $H$ belong to $[0, 1]$. Let the EVD of $H$ be

$$H = U \Sigma U^{T},$$

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

$$S_{1}^{-1} = R_{1}^{-1} (I - U \Sigma U^{T})^{-1} R_{1}^{T}$$

$$= R_{1}^{-1} U (I - \Sigma)^{-1} U^{T} R_{1}^{T}$$

$$= R_{1}^{-1} \left( I + U \left( (I - \Sigma)^{-1} - I \right) U^{T} \right) R_{1}^{T}$$

$$= A_{1}^{-1} + R_{1}^{-1} U \left( (I - \Sigma)^{-1} - I \right) U^{T} R_{1}^{T}.$$
If $H$ has an approximate EVD of the form

$$H \approx U\Sigma U^\top, \quad \Sigma = \text{diag}\{\bar{\sigma}_1, \ldots, \bar{\sigma}_n\},$$

then an approximation of $S_k^{-1}$ is

$$S_k^{-1} = A_k^{-1} + R_k^{-1}U \left( (I - \Sigma)^{-1} - I \right) U^\top R_k^{-\top}. \quad (2.24)$$

The simplest selection of $\Sigma$ is the one that ensures the $k$ largest eigenvalues of $(I - \Sigma)^{-1}$ match the largest eigenvalues of $(I - \Sigma)^{-1}$. Li, Xi, and Saad [27] set $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_k, \theta, \ldots, \theta)$, where $\theta \in [0, 1]$. The resulting preconditioner can be written as

$$S_k^{-1} = \frac{1}{1 - \theta} A_k^{-1} + Z_k \left( (I - \Sigma_k)^{-1} - \frac{1}{1 - \theta} I \right) Z_k^\top, \quad (2.25)$$

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

In the next section, we analyze 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:

Given $\varepsilon > 0$, find all the eigenpairs $(\lambda, z) \in \mathbb{R} \times \mathbb{R}^{nr}$ such that

$$S_k z = \lambda A_k z, \quad \lambda < \varepsilon. \quad (3.1)$$

This can be rewritten as

Given $\varepsilon > 0$, find all the eigenpairs $(\lambda, z) \in \mathbb{R} \times \mathbb{R}^{nr}$ such that

$$(I - H) u = \lambda u, \quad z = R_k^{-1} u, \quad \lambda < \varepsilon, \quad (3.2)$$

where $R_k$ and $H$ are given by (2.16) and (2.20). Consider also the eigenproblem

Given $\varepsilon > 0$, find all the eigenpairs $(\sigma, u) \in \mathbb{R} \times \mathbb{R}^{nr}$ such that

$$Hu = \sigma u, \quad \sigma > 1 - \varepsilon. \quad (3.2)$$

As already observed, each eigenpair $(\lambda, z)$ of (3.1) corresponds to the eigenpair $(1 - \lambda, R_k z)$ 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 $[V_j, v_{j+1}]$ is the output of the Lanczos basis of the Krylov subspace; then the subspace relations that hold at iteration $j$ are

$$(I - H)V_j = V_jT_j + v_{j+1}h_j^\top,$$

$$HV_j = V_j(I - T_j) - v_{j+1}h_j^\top,$$
where $T_j \in \mathbb{R}^{j \times j}$ is a symmetric tridiagonal matrix and $h_j \in \mathbb{R}^j$. The eigenpair $(\lambda, z)$ (respectively, $(\sigma, u)$) corresponding to the smallest (respectively, largest) eigenvalue in (3.1) (respectively, (3.2)) is approximated by the eigenpair $(\lambda, R_{T_j}^{-1} \nu_j \tilde{u})$ (respectively, $(\sigma, \nu_j \tilde{u})$) 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 nonconvergent part of the Krylov subspace [41]. 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 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 $\tilde{S}_1$ was found to perform poorly, which is generally the case when $\kappa(A)$ is large. Table 2 reports the Lanczos iteration counts ($it_{\text{Lan}}$) 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 ($it_{\text{PCG}}$) for solving the linear system (2.13) using the first-level preconditioner $\tilde{S}_1 = A_{T_j}^{-1}$ and the two-level preconditioner $\tilde{S}_2$ 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 $\varepsilon_{\text{Lan}}$ in the Lanczos method can reduce the total iteration count for $\tilde{S}_2$.

Table 3 illustrates the effect of varying $\varepsilon_{\text{Lan}}$ for problem e3d (results for the other test problems are consistent). Although $it_{\text{Lan}}$ decreases as $\varepsilon_{\text{Lan}}$ increases, $it_{\text{PCG}}$ increases, and the total count still exceeds the 175 PCG iterations required by the first-level preconditioner $\tilde{S}_1$.

As already observed, in [48] a recursive (multilevel) scheme is proposed to help mitigate the computational costs of building and applying the preconditioner. Nevertheless, the Lanczos method is still used, albeit with reduced costs for applying the operator matrices.

![Figure 1](image-url)
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 EVD. Because $H$ is SPSD, Nyström’s method can be used. 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 $S_2$ 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 $A_r$, 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 behavior 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.

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 $S_T$. Applying the Sherman–Morrison–Woodbury identity [13, section 2.1.3], the inverse of the Schur complement $S_T$ (2.11) can be written as

$$
S_T^{-1} = A_T^{-1} + A_T^{-1}A_I(A_I - A_{II}A_T^{-1}A_I)^{-1}A_{II}A_T^{-1} = A_T^{-1} + A_T^{-1}A_I S_I^{-1} A_{II} A_T^{-1},
$$

where

$$
S_I = A_I - A_{II} A_T^{-1} A_I
$$

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

$$
R_T S_T^{-1} R_T^T = I + R_{II}^{-1} A_{II} S_I^{-1} A_{II} R_T^{-1}.
$$

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

$$
R_T^{-1} A_{II} S_I^{-1} A_{II} R_T^{-1} \approx \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^T,
$$

where $\tilde{U}_k \in \mathbb{R}^{n_T \times k}$ is orthonormal and $\tilde{\Sigma}_k \in \mathbb{R}^{k \times k}$ is diagonal. Combining (4.3) and (4.4), we can define a preconditioner for $R_T^{-1} S_T R_T^{-1}$ to be

$$
P_1 = I + \tilde{U}_k \tilde{\Sigma}_k \tilde{U}_k^T.
$$

The preconditioned matrix $P_1 R_T S_T R_T^T$ is spectrally equivalent to $R_T^{-1} P_1 R_T^T S_T$. Therefore, the preconditioned system can be written as

$$
M_1 S_T = R_T^{-1} P_1 R_T^T S_T = \left( A_T^{-1} + \tilde{Z}_k \tilde{\Sigma}_k \tilde{Z}_k^T \right) S_T,
$$

where $\tilde{Z}_k \in \mathbb{R}^{n_T \times k}$ is orthonormal.
where $\tilde{Z}_k = R_\Gamma^{-1} \tilde{U}_k$. If (4.4) is obtained using a truncated EVD denoted by $U_k \Sigma_k U_k^\top$, then $\tilde{U}_k = U_k$ 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 $R_\Gamma^{-1} S_\Gamma R_\Gamma^\top$. 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 $R_\Gamma^{-1} S_\Gamma R_\Gamma^\top U_k = U_k (I + \Sigma_k)^{-1}$, and the preconditioned matrix is

$$\mathcal{P}_{\text{1-A-DEF}} R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} = R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} + U_k \Sigma_k (I + \Sigma_k)^{-1} U_k^\top$$

$$= R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} + U_k ((I + \Sigma_k) - I) (I + \Sigma_k)^{-1} U_k^\top$$

$$= R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} - U_k (I + \Sigma_k)^{-1} U_k^\top + U_k U_k^\top,$$

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

Note that given the matrix $\tilde{U}_k$ 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 $E_k = \tilde{U}_k R_\Gamma^{-\top} S_\Gamma R_\Gamma^{-1} \tilde{U}_k$ and $Q = \tilde{U}_k E^{-1} \tilde{U}_k^\top$, the deflation preconditioner is

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

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

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

is a two-level preconditioner for $S_\Gamma$.

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 $R_\Gamma^{-\top} A_\Gamma S_\Gamma^{-1} A_\Gamma R_\Gamma^{-1}$ to a set of $k + p$ vectors, where $k > 0$ is the required rank and $p \geq 0$. Because explicitly computing the SPD matrix $S_\Gamma = A_\Gamma - A_\Gamma A_\Gamma^{-1} A_\Gamma$ and factorizing it is prohibitively expensive, applying $S_\Gamma^{-1}$ 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 $R_\Gamma$, then for $S_\Gamma$, and then for $R_\Gamma$. However, the Nyström method requires the solution of only one linear system with $k + p$ right-hand sides for $R_\Gamma$, then for $S_\Gamma$, and then for $R_\Gamma$. 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 [34], for solving the system with $S_\Gamma$ 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 $R_\Gamma$. For large-scale problems, computing the Cholesky factorization $A_\Gamma = R_\Gamma^\top R_\Gamma$ is prohibitive, so we would like to avoid computations with $R_\Gamma$. We can achieve this by using an iterative solver to solve linear systems with $A_\Gamma$. Note that this is possible when solving the generalized eigenvalue problem (2.15). Because $A_\Gamma$ is typically well conditioned, so too is $R_\Gamma$. Thus, we can reduce the cost of computing the Nyström–Schur preconditioner by approximating the SPSD matrix $A_\Gamma S_\Gamma^{-1} A_\Gamma$ (or even by approximating $S_\Gamma^{-1}$). Of course, this needs
to be done without seriously adversely affecting the preconditioner quality. Using an approximate factorization

\[(4.9)\]

\[A_{\Gamma}S_{\Gamma}^{-1}A_{\Gamma} \approx \hat{W}_k\hat{\Sigma}_k \hat{W}_k^\top,\]

an alternative deflation preconditioner is

\[(4.12)\]

\[
P_2 = I + R_{\Gamma}^{-\top} \hat{W}_k \hat{\Sigma}_k \hat{W}_k^\top R_{\Gamma}^{-1}
= R_{\Gamma}^{-\top} \left( A_{\Gamma} + \hat{W}_k \hat{\Sigma}_k \hat{W}_k^\top \right) R_{\Gamma}^{-1}.
\]

The preconditioned Schur complement \(P_2 R_{\Gamma}^{-\top} S_{\Gamma} R_{\Gamma}^{-1}\) is spectrally similar to \(R_{\Gamma}^{-1} P_2 R_{\Gamma}^{-\top} S_{\Gamma}\), and, setting \(\tilde{Z}_k = A_{\Gamma}^{-1} \hat{W}_k\), we have

\[(4.10)\]

\[M_2 S_{\Gamma} = R_{\Gamma}^{-1} P_2 R_{\Gamma}^{-\top} S_{\Gamma} = (A_{\Gamma}^{-1} + \hat{Z}_k \hat{\Sigma}_k \hat{Z}_k^\top) S_{\Gamma}.
\]

Thus \(M_2 = A_{\Gamma}^{-1} + \hat{Z}_k \hat{\Sigma}_k \hat{Z}_k^\top\) is a variant of the Nyström–Schur preconditioner for \(S_{\Gamma}\) that avoids computing \(R_{\Gamma}\).

Alternatively, assuming we have an approximate factorization

\[(4.11)\]

\[S_{\Gamma}^{-1} \approx \hat{V}_k \hat{\Sigma}_k \hat{V}_k^\top\]

yields

\[
P_3 = I + R_{\Gamma}^{-\top} A_{\Gamma} \hat{V}_k \hat{\Sigma}_k \hat{V}_k^\top A_{\Gamma} R_{\Gamma}^{-1}.
\]

Again, \(P_3 R_{\Gamma}^{-\top} S_{\Gamma} R_{\Gamma}^{-1}\) is spectrally similar to \(R_{\Gamma}^{-1} P_3 R_{\Gamma}^{-\top} S_{\Gamma}\), and, setting \(\tilde{Z}_k = A_{\Gamma}^{-1} A_{\Gamma} \hat{V}_k\), we have

\[(4.12)\]

\[M_3 S_{\Gamma} = R_{\Gamma}^{-1} P_3 R_{\Gamma}^{-\top} S_{\Gamma} = (A_{\Gamma}^{-1} + \hat{Z}_k \hat{\Sigma}_k \hat{Z}_k^\top) S_{\Gamma},\]

which gives another variant of the Nyström–Schur preconditioner. In a similar way to defining \(M_{2-A-DEF}\) (4.7), we can define \(M_{3-A-DEF}\) and \(M_{3-A-DEF}\). Note that \(M_{2-A-DEF}\) and \(M_{3-A-DEF}\) also avoid computations with \(R_{\Gamma}\).

### 4.4. Nyström–Schur preconditioner

Algorithm 4.1 presents the construction of the Nyström–Schur preconditioner \(M_2\); an analogous derivation yields the variant \(M_3\). Step 3 is the most expensive step, that is, solving the \(n_I \times n_I\) SPD linear system

\[(4.13)\]

\[S_{\Gamma}X = F,\]

where \(F \in \mathbb{R}^{n_I \times (k+p)}\) and \(S_{\Gamma} = A_{\Gamma} - A_{\Gamma} A_{\Gamma}^{-1} A_{\Gamma}^\top\). Using an iterative solver requires a linear system solve with \(A_{\Gamma}\) 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 \(M_2\) to a vector requires the solution of a linear system with \(A_{\Gamma}\) and a low rank correction; see step 12 of Algorithm 4.1.

### 4.5. Estimation of the spectral condition number

In this section, we provide an expectation of the spectral condition number of \(S_{\Gamma}\) preconditioned by the Nyström–Schur preconditioner. Saibaba [36] 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
Algorithm 4.1. Construction of the Nyström–Schur preconditioner (4.10)

Input: $A$ in block form (2.9), $k > 0$ and $p \geq 0$ ($k, p \ll n_r$) and $\varepsilon > 0$.

Output: Two-level preconditioner for the Nyström method. Let $\Pi$ denote the orthogonal projector on the space spanned by the columns of the matrix $M$.

1. Draw a random matrix $G \in \mathbb{R}^{n_r \times (k+p)}$.
2. Compute $F = A蓥 G$.
3. Solve $S蓥 X = F$.
5. Compute $Y = QR$.
6. Set $C = G^\top Y$.
7. Compute the EVD $C = V_1 D_1^\top + V_2 D_2^\top$, where $D_1$ contains all the eigenvalues that are at least $\varepsilon$.
8. Set $T = R蓥 V_1 D_1^\top R^\top$.
9. Compute the EVD $T = WEW^\top$.
10. Set $\bar{U} = YW(:, 1 : k), \Sigma = E(1 : k, 1 : k)$.
11. Solve $A蓥 Z = \bar{U}$.
12. Define the preconditioner $M_2 = A蓥^{-1} + Z\Sigma Z^\top$.

Output: Two-level preconditioner for the Nyström method. Let $\Pi$ denote the orthogonal projector on the space spanned by the columns of the matrix $M$.

1. Draw a random matrix $G \in \mathbb{R}^{n_r \times (k+p)}$.
2. Compute $F = A蓥 G$.
3. Solve $S蓥 X = F$.
5. Compute $Y = QR$.
6. Set $C = G^\top Y$.
7. Compute the EVD $C = V_1 D_1^\top + V_2 D_2^\top$, where $D_1$ contains all the eigenvalues that are at least $\varepsilon$.
8. Set $T = R蓥 V_1 D_1^\top R^\top$.
9. Compute the EVD $T = WEW^\top$.
10. Set $\bar{U} = YW(:, 1 : k), \Sigma = E(1 : k, 1 : k)$.
11. Solve $A蓥 Z = \bar{U}$.
12. Define the preconditioner $M_2 = A蓥^{-1} + Z\Sigma Z^\top$.

Proposition 4.1. Let the EVD of the SPD matrix $I - H = R蓥^{-1} S蓥 R蓥^{-1}$ be

$$
[U \perp \ U_k] \begin{bmatrix} \Lambda \perp & \Lambda_k \end{bmatrix} \begin{bmatrix} U \perp^\top \\ U_k^\top \end{bmatrix},
$$

where $\Lambda \perp \in \mathbb{R}^{(n_r-k) \times (n_r-k)}$ and $\Lambda_k \in \mathbb{R}^{k \times k}$ are diagonal matrices with the eigenvalues $(\lambda_j)_{k \geq 1}$ (and $\lambda_i, \lambda_i_{n_r \geq 1}$ $\geq 2k+1$, respectively, in decreasing order. Furthermore, assume

$$
\sin \angle (u_j, \bar{U}) = \|u_j - \Pi蓥 u_j\|_2 \leq \gamma_{j,k}^{q+1} c,
$$

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

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

and $c$ has the expected value

$$
\mathbb{E}(c) = \sqrt{\frac{k}{p-1}} + \frac{\varepsilon \sqrt{(k+p)(n_r-k)}}{p},
$$

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

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

Note that if $\lambda_j \leq 1/2$, then $\gamma_{j,k} \leq 2\lambda_j / \lambda_{k+1} (j = 1, \ldots, k)$.
that \( \lambda_k \leq 1/2 \). Let the columns of \( \tilde{U} \in \mathbb{R}^{n \times (k+p)} \) be the approximate eigenvectors of \( I - H \) computed using the Nyström method, and let

\[
\mathcal{P} = I - (I - H)\tilde{U}E^{-1}\tilde{U}^\top \quad \text{with} \quad E = \tilde{U}^\top(I - H)\tilde{U}
\]

be the associated deflation preconditioner. Then, the effective condition number of the two-level preconditioner \( \mathcal{P}(I - H) = \mathcal{P}R_I^{-T}S_I\mathcal{P}^{-1} \) satisfies

\[
\mathbb{E} \left( \sqrt{\kappa_{\text{eff}}(\mathcal{P}(I - H))} \right) \leq c_1 \sqrt{\frac{\lambda_{nr}}{\lambda_{k+1}}},
\]

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

\[
\|x - \Pi_{\tilde{U}} x\|_2^2 \leq c_K \frac{\|x\|_{I - H}^2}{\|I - H\|_2},
\]

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

\[
\kappa_{\text{eff}}(\mathcal{P}(I - H)) \leq c_K.
\]

Let \( t \leq k \), and consider

\[
\|x - \Pi_{\tilde{U}} x\|_2^2 = \left\| \sum_{i=1}^{nr} \alpha_i u_i - \Pi_{\tilde{U}} \sum_{i=1}^{nr} \alpha_i u_i \right\|_2^2 \\
\leq \left\| \sum_{i=t+1}^{nr} (I - \Pi_{\tilde{U}}) \alpha_i u_i \right\|_2^2 + \sum_{i=1}^{t} |\alpha_i| \|u_i - \Pi_{\tilde{U}} u_i\|_2^2 \\
\leq \left\| \sum_{i=t+1}^{nr} \alpha_i u_i \right\|_2^2 + \sum_{i=1}^{t} |\alpha_i| \|u_i - \Pi_{\tilde{U}} u_i\|_2.
\]

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

\[
\left\| \sum_{i=t+1}^{nr} \alpha_i u_i \right\|_2^2 \leq \frac{1}{\sqrt{\lambda_{t+1}}} \left\| \sum_{i=t+1}^{nr} \sqrt{\lambda_i} \alpha_i u_i \right\|_2^2 \leq \frac{1}{\sqrt{\lambda_{t+1}}} \left\| \sum_{i=t+1}^{nr} \sqrt{\lambda_i} \alpha_i u_i \right\|_2 \\
\leq \frac{1}{\sqrt{\lambda_{t+1}}} \sum_{i=t+1}^{nr} \lambda_i \alpha_i^2 = \frac{1}{\sqrt{\lambda_{t+1}}} \|x - \Pi_{U_t} x\|_{I - H} = \frac{\lambda_{nr}}{\sqrt{\lambda_{t+1}}} \frac{\|x - \Pi_{U_t} x\|_{I - H}}{\sqrt{\|I - H\|_2}}.
\]

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

\[
\sum_{i=1}^{t} |\alpha_i| \|u_i - \Pi_{\tilde{U}} u_i\|_2 \leq \sum_{i=1}^{t} |\alpha_i| \gamma_{i,k}^{q+1} \leq c_{\gamma_{t,k}} \sum_{i=1}^{t} |\alpha_i| \sqrt{\gamma_{i,k}} \\
= c_{\gamma_{t,k}} \sqrt{\lambda_{k+1}^{-1} - \sum_{i=1}^{t} |\alpha_i| \frac{1}{\sqrt{\lambda_i^{-1}}}} \\
\leq c_{\gamma_{t,k}} \sqrt{\lambda_{k+1}^{-1} - \sum_{i=1}^{t} |\alpha_i| \frac{1}{\sqrt{\lambda_i^{-1}}}}.
\]
Thus, the entries of $\Lambda$ are close to the origin. This is reflected in a rapid and exponential decay of the values $\lambda_i$. Note that, in practice, when the problem is challenging, a few eigenvalues of $\Lambda$ are close to the origin. Since $\lambda_k \geq \lambda_t$, we have

$$\sum_{i=1}^{t} |\alpha_i| \| u_i - \Pi_{I} u_i \|_2 \leq \sqrt{2} c_{t,k} \frac{1}{\sqrt{\lambda_{k+1}}} \sum_{i=1}^{t} |\alpha_i| \frac{1}{\sqrt{\lambda_i}} \leq \sqrt{2} c_{t,k} \frac{1}{\sqrt{\lambda_{k+1}}} \sum_{i=1}^{t} |\alpha_i| \sqrt{\lambda_i}.$$ 

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

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

Since $\lambda_k \geq \lambda_t$, we have

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

It follows that

$$\| x - \Pi_{I} x \|_2 \leq \sqrt{\frac{\lambda_{nr}}{\lambda_{t+1}}} \frac{\| x - \Pi_{U_t} x \|_{I-H}}{\sqrt{\| I-H \|_2}} + c \sqrt{2 t_{t,k} q^2} \frac{\lambda_{nr}}{\lambda_{t+1}} \frac{\| \Pi_{U_t} x \|_{I-H}}{\sqrt{\| I-H \|_2}} \leq \sqrt{2} \max \left( c \sqrt{2 t_{t,k} q^2}, 1 \right) \sqrt{\frac{\lambda_{nr}}{\lambda_{t+1}}} \frac{\| x \|_{I-H}}{\sqrt{\| I-H \|_2}}.$$ 

Hence (4.19) is satisfied, and we have

$$\kappa_{\text{eff}} (P(I-H)) \leq 2 \max \left( 2 e^2 t_{t,k} q^2, 1 \right) \frac{\lambda_{nr}}{\lambda_{t+1}}.$$ 

Thus,

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

Since $t$ is chosen arbitrarily between 1 and $k$, we have

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

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

Note that, in practice, when the problem is challenging, a few eigenvalues of $R_T^{-1} S_T R_T^{-1}$ 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 s3rtm3m3.
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 \( S_1 \) and \( S_2 \) 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 \( S_2 \) and the reduction in iterations it gives compared to \( S_1 \). 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 \( M_2 \) (4.10) (recall that its construction does not require the Cholesky factors of \( A_I \)). The relative convergence tolerance used when solving the SPD system (4.13) is \( \varepsilon_{S_1} = 0.1 \). This system (4.13) is preconditioned by the block diagonal matrix \( A_I \). We denote by \( it_{S_1} \) the number of block PCG iterations required to solve (4.13) during the construction of the Nyström–Schur preconditioners (it is zero for \( S_1 \) and \( S_2 \)) and by \( it_{PCG} \) the PCG iteration count for solving (2.13). The total number of iterations is \( it_{total} = it_{S_1} + it_{PCG} \). 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 behavior 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, 34]. For PCG, \( iters \) is the largest PCG

![Fig. 2. Problem s3m3: values of the bound (4.20) on \( \mathbb{E} \left( \sqrt{k} \varepsilon_{co} \left( P(I - H) \right) \right) \) for a range of values of \( k \) and \( q \).]
iteration count over the \(k + p\) right-hand sides. For the block method, \(\text{iters} = it_{S_i}\) 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 \(\varepsilon_{S_i}\)

We now study the impact of the convergence tolerance \(\varepsilon_{S_i}\) 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 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 \(S_2\). The use of a large \(\varepsilon_{S_i}\) 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 \(M_i\) and \(M_{i,A,DEF}\) (\(i = 1, 2, 3\)) 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 \(S_1\).

### 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
Table 6
The effects of the convergence tolerance $\varepsilon_{S_i}$ on the quality of the Nyström–Schur preconditioner.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>$\varepsilon_{S_1}$</th>
<th>$\varepsilon_{S_2}$</th>
<th>$\tilde{M}_2$</th>
<th>$\tilde{S}_1$</th>
<th>$\tilde{S}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>el2d</td>
<td>0.8</td>
<td>1</td>
<td>500+</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>68</td>
<td>228</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>70</td>
<td>228</td>
<td>914</td>
<td>231</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>72</td>
<td>228</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>78</td>
<td>228</td>
<td></td>
<td></td>
</tr>
<tr>
<td>el3d</td>
<td>0.8</td>
<td>1</td>
<td>173</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>2</td>
<td>171</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>22</td>
<td>52</td>
<td>174</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>24</td>
<td>52</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>27</td>
<td>52</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nd3k</td>
<td>0.8</td>
<td>32</td>
<td>178</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>32</td>
<td>178</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>32</td>
<td>178</td>
<td>603</td>
<td>143</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>32</td>
<td>178</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>33</td>
<td>178</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7
Comparison of $it_{\text{total}}$ for the variants of the Nyström–Schur preconditioner and $\tilde{S}_1$ and $\tilde{S}_2$. $\varepsilon_{S_i} = 0.1$.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>$\tilde{M}_1$</th>
<th>$\tilde{M}_{1,\text{A-DEF}}$</th>
<th>$\tilde{M}_2$</th>
<th>$\tilde{M}_{2,\text{A-DEF}}$</th>
<th>$\tilde{M}_3$</th>
<th>$\tilde{M}_{3,\text{A-DEF}}$</th>
<th>$\tilde{S}_1$</th>
<th>$\tilde{S}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcsstk38</td>
<td>218</td>
<td>218</td>
<td>219</td>
<td>219</td>
<td>360</td>
<td>360</td>
<td>313</td>
<td>584</td>
</tr>
<tr>
<td>el2d</td>
<td>266</td>
<td>267</td>
<td>300</td>
<td>300</td>
<td>282</td>
<td>282</td>
<td>282</td>
<td>282</td>
</tr>
<tr>
<td>el3d</td>
<td>73</td>
<td>72</td>
<td>76</td>
<td>75</td>
<td>78</td>
<td>78</td>
<td>76</td>
<td>174</td>
</tr>
<tr>
<td>msc10848</td>
<td>206</td>
<td>205</td>
<td>213</td>
<td>211</td>
<td>216</td>
<td>216</td>
<td>222</td>
<td>612</td>
</tr>
<tr>
<td>nd3k</td>
<td>205</td>
<td>205</td>
<td>210</td>
<td>210</td>
<td>211</td>
<td>211</td>
<td>603</td>
<td>143</td>
</tr>
<tr>
<td>s3rmt3m3</td>
<td>127</td>
<td>127</td>
<td>135</td>
<td>134</td>
<td>161</td>
<td>161</td>
<td>153</td>
<td>441</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>$k$</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>40</th>
<th>80</th>
<th>160</th>
<th>320</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tilde{M}_2$</td>
<td>$it_{\tilde{S}_1}$</td>
<td>97</td>
<td>57</td>
<td>37</td>
<td>23</td>
<td>16</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>$it_{\text{PCG}}$</td>
<td>244</td>
<td>203</td>
<td>98</td>
<td>53</td>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>$\tilde{S}_2$</td>
<td>$it_{\text{PCG}}$</td>
<td>212</td>
<td>153</td>
<td>70</td>
<td>37</td>
<td>22</td>
<td>13</td>
</tr>
</tbody>
</table>

parameter $p$. For problem s3rmt3m3, Table 8 compares the iteration counts for $\tilde{M}_2$ with that of the ideal two-level preconditioner $\tilde{S}_2$ for $k$ ranging from 5 to 320. For $\tilde{S}_1$, 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 $S_I$ 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, $it_{\text{PCG}}$ reduces toward the lower bound of 70 PCG iterations required by $\tilde{S}_2$. 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 the 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` [38] 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 parallelized). Table 10 presents the PCG iteration counts for the Nyström–Schur preconditioner $M_2$ (with $k = 20$) and the IC preconditioners `HSL_MI28` and `ichol`. 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 $M_2$ decreases monotonically, while for the IC preconditioners we observe oscillatory behavior.

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.

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 a 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 is 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 $A_F$ (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 [48]. 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 $S_1$, $M_2$, $S_2$, and HSL MI28 preconditioners. In all the tests, we use 64 subdomains. For $M_2$, 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 $M_2$, we sum the counts $it_{S_1}$ and $it_{PCG}$). Note that we do

![Fig. 4. PCG residual norm history for test examples bcstlk38 (top) and ela2d (bottom).](image-url)
Table 11

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

<table>
<thead>
<tr>
<th>Identifier</th>
<th>( S_1 )</th>
<th>( M_2 )</th>
<th>( S_2 )</th>
<th>HSL_MI28</th>
<th>( \kappa(A) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>a01</td>
<td>118</td>
<td>19</td>
<td>45</td>
<td>31</td>
<td>17</td>
</tr>
<tr>
<td>apache1</td>
<td>667</td>
<td>122</td>
<td>291</td>
<td>192</td>
<td>72</td>
</tr>
<tr>
<td>bcsstk17</td>
<td>349</td>
<td>46</td>
<td>55</td>
<td>48</td>
<td>59</td>
</tr>
<tr>
<td>bcsstk25</td>
<td>136</td>
<td>40</td>
<td>77</td>
<td>45</td>
<td>26</td>
</tr>
<tr>
<td>bcsstk36</td>
<td>451</td>
<td>419</td>
<td>145</td>
<td>31</td>
<td>5e+12</td>
</tr>
<tr>
<td>bcsstk38</td>
<td>584</td>
<td>46</td>
<td>171</td>
<td>129</td>
<td>5e+04</td>
</tr>
<tr>
<td>canne</td>
<td>( \top )</td>
<td>57</td>
<td>228</td>
<td>396</td>
<td>9e+10</td>
</tr>
<tr>
<td>cf1</td>
<td>209</td>
<td>30</td>
<td>72</td>
<td>50</td>
<td>274</td>
</tr>
<tr>
<td>conpgh</td>
<td>185</td>
<td>47</td>
<td>177</td>
<td>136</td>
<td>50</td>
</tr>
<tr>
<td>gridgema</td>
<td>426</td>
<td>90</td>
<td>377</td>
<td>298</td>
<td>6e+05</td>
</tr>
<tr>
<td>gyro</td>
<td>( \top )</td>
<td>55</td>
<td>346</td>
<td>518</td>
<td>4e+09</td>
</tr>
<tr>
<td>gyro_k</td>
<td>( \top )</td>
<td>55</td>
<td>346</td>
<td>518</td>
<td>3e+09</td>
</tr>
<tr>
<td>gyro_km</td>
<td>165</td>
<td>14</td>
<td>24</td>
<td>22</td>
<td>1e+07</td>
</tr>
<tr>
<td>m291</td>
<td>867</td>
<td>85</td>
<td>247</td>
<td>187</td>
<td>3e+11</td>
</tr>
<tr>
<td>m292</td>
<td>15</td>
<td>3</td>
<td>5</td>
<td>10</td>
<td>5e+01</td>
</tr>
<tr>
<td>m293</td>
<td>612</td>
<td>47</td>
<td>268</td>
<td>116</td>
<td>145</td>
</tr>
<tr>
<td>m294</td>
<td>479</td>
<td>69</td>
<td>220</td>
<td>175</td>
<td>1e+12</td>
</tr>
<tr>
<td>m295</td>
<td>1279</td>
<td>135</td>
<td>406</td>
<td>421</td>
<td>1e+09</td>
</tr>
<tr>
<td>m296</td>
<td>1091</td>
<td>56</td>
<td>301</td>
<td>230</td>
<td>102</td>
</tr>
<tr>
<td>m297</td>
<td>1184</td>
<td>108</td>
<td>325</td>
<td>248</td>
<td>1e+07</td>
</tr>
<tr>
<td>m298</td>
<td>1428</td>
<td>69</td>
<td>489</td>
<td>757</td>
<td>557</td>
</tr>
<tr>
<td>m299</td>
<td>869</td>
<td>89</td>
<td>138</td>
<td>73</td>
<td>1e+02</td>
</tr>
<tr>
<td>m300</td>
<td>287</td>
<td>108</td>
<td>325</td>
<td>248</td>
<td>1e+07</td>
</tr>
<tr>
<td>m301</td>
<td>1296</td>
<td>232</td>
<td>281</td>
<td>1e+14</td>
<td></td>
</tr>
<tr>
<td>m302</td>
<td>1929</td>
<td>155</td>
<td>517</td>
<td>265</td>
<td>111</td>
</tr>
<tr>
<td>m303</td>
<td>1093</td>
<td>165</td>
<td>386</td>
<td>268</td>
<td>120</td>
</tr>
<tr>
<td>m304</td>
<td>154</td>
<td>19</td>
<td>59</td>
<td>34</td>
<td>3e+06</td>
</tr>
<tr>
<td>m305</td>
<td>192</td>
<td>24</td>
<td>59</td>
<td>39</td>
<td>3e+08</td>
</tr>
<tr>
<td>m306</td>
<td>231</td>
<td>28</td>
<td>54</td>
<td>41</td>
<td>39</td>
</tr>
<tr>
<td>m307</td>
<td>260</td>
<td>11</td>
<td>44</td>
<td>45</td>
<td>3e+11</td>
</tr>
<tr>
<td>m308</td>
<td>( \top )</td>
<td>148</td>
<td>339</td>
<td>236</td>
<td>610</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Identifier</th>
<th>( S_1 )</th>
<th>( M_2 )</th>
<th>( S_2 )</th>
<th>HSL_MI28</th>
<th>( \kappa(A) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>s3dkt3m2</td>
<td>164</td>
<td>318</td>
<td>270</td>
<td>1107</td>
<td>3e+10</td>
</tr>
<tr>
<td>s3rmq4m1</td>
<td>356</td>
<td>31</td>
<td>80</td>
<td>36</td>
<td>472</td>
</tr>
<tr>
<td>s3rmq4m2</td>
<td>434</td>
<td>36</td>
<td>101</td>
<td>64</td>
<td>413</td>
</tr>
<tr>
<td>s3rmq4m3</td>
<td>441</td>
<td>27</td>
<td>101</td>
<td>70</td>
<td>630</td>
</tr>
<tr>
<td>s3rmt3m1</td>
<td>1453</td>
<td>387</td>
<td>600</td>
<td>168</td>
<td>1177</td>
</tr>
<tr>
<td>s3rmt3m2</td>
<td>399</td>
<td>59</td>
<td>112</td>
<td>72</td>
<td>95</td>
</tr>
<tr>
<td>someq1</td>
<td>169</td>
<td>30</td>
<td>62</td>
<td>47</td>
<td>30</td>
</tr>
<tr>
<td>PresPoisson</td>
<td>92</td>
<td>13</td>
<td>29</td>
<td>19</td>
<td>32</td>
</tr>
<tr>
<td>cranker</td>
<td>92</td>
<td>16</td>
<td>49</td>
<td>31</td>
<td>34</td>
</tr>
<tr>
<td>crus</td>
<td>89</td>
<td>17</td>
<td>47</td>
<td>32</td>
<td>38</td>
</tr>
<tr>
<td>kru</td>
<td>81</td>
<td>16</td>
<td>44</td>
<td>31</td>
<td>10</td>
</tr>
<tr>
<td>bodyy5</td>
<td>72</td>
<td>19</td>
<td>67</td>
<td>57</td>
<td>4</td>
</tr>
<tr>
<td>Dubcov02</td>
<td>61</td>
<td>31</td>
<td>32</td>
<td>21</td>
<td>14</td>
</tr>
<tr>
<td>dmbk</td>
<td>55</td>
<td>9</td>
<td>51</td>
<td>30</td>
<td>47</td>
</tr>
<tr>
<td>f7</td>
<td>50</td>
<td>12</td>
<td>31</td>
<td>21</td>
<td>8</td>
</tr>
<tr>
<td>Dubcov03</td>
<td>39</td>
<td>8</td>
<td>24</td>
<td>15</td>
<td>7</td>
</tr>
<tr>
<td>bodyt</td>
<td>38</td>
<td>8</td>
<td>29</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>julian01</td>
<td>22</td>
<td>4</td>
<td>21</td>
<td>19</td>
<td>4</td>
</tr>
<tr>
<td>bundle1</td>
<td>13</td>
<td>3</td>
<td>8</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>blkedge</td>
<td>12</td>
<td>3</td>
<td>12</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>abelc</td>
<td>12</td>
<td>3</td>
<td>12</td>
<td>12</td>
<td>3</td>
</tr>
<tr>
<td>tors01</td>
<td>12</td>
<td>3</td>
<td>12</td>
<td>12</td>
<td>3</td>
</tr>
<tr>
<td>waveh001</td>
<td>12</td>
<td>3</td>
<td>12</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>waveh010</td>
<td>12</td>
<td>3</td>
<td>12</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>fcu001</td>
<td>7</td>
<td>2</td>
<td>7</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>fcu01</td>
<td>7</td>
<td>2</td>
<td>7</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>shallow_water2</td>
<td>7</td>
<td>40</td>
<td>7</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>shallow_water1</td>
<td>5</td>
<td>20</td>
<td>5</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>mm0</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>qa8fm</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>crystm02</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>crystm03</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>finan512</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>ted</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>ted_B</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Trefethen_20000</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Trefethen_20000b</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

Fig. 5. Iteration count performance profile for the large test set. On the left, all 71 problems are included. On the right, only the 40 problems for which the \( S_1 \) (one-level) iteration count exceeded 100 are included.

not include \( S_2 \) in the performance profiles because it is an ideal but impractical two-level preconditioner and, as such, it always outperforms \( M_2 \). The performance profile shows that on the problems where \( S_1 \) struggles, there is little to choose between the overall quality of \( M_2 \) and HSL_MI28.

REFERENCES


Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.
A3860  HUSSAM AL DAAS, TYRONE REES, AND JENNIFER SCOTT


