MAKING THE NYSTRÖM METHOD HIGHLY ACCURATE FOR LOW-RANK APPROXIMATIONS*

JIANLIN XIA†

Abstract. The Nyström method is a convenient heuristic method to obtain low-rank approximations to kernel matrices in nearly linear complexity. Existing studies typically use the method to approximate positive semidefinite matrices with low or modest accuracies. In this work, we propose a series of heuristic strategies to make the Nyström method reach high accuracies for nonsymmetric and/or rectangular matrices. The resulting methods (called *high-accuracy Nyström methods*) treat the Nyström method and a skinny rank-revealing factorization as a fast pivoting strategy in a progressive alternating direction refinement process. Two refinement mechanisms are used: alternating the row and column pivoting starting from a small set of randomly chosen columns, and adaptively increasing the number of samples until a desired rank or accuracy is reached. A fast subset update strategy based on the progressive sampling of Schur complements is further proposed to accelerate the refinement process. Efficient randomized accuracy control is also provided. Relevant accuracy and singular value analysis is given to support some of the heuristics. Extensive tests with various kernel functions and data sets show how the methods can quickly reach prespecified high accuracies in practice, sometimes with quality close to SVDs, using only small numbers of progressive sampling steps.

Key words. high-accuracy Nyström method, kernel matrix, low-rank approximation, progressive sampling, alternating direction refinement, error analysis

AMS subject classifications. 15A23, 65F10, 65F30

1. Introduction.

The Nyström method is a very useful technique for data analysis and machine learning. It can be used to quickly produce low-rank approximations to data matrices. The original Nyström method in [37] is designed for symmetric positive definite kernel matrices and it essentially uses uniform sampling to select rows/columns (that correspond to some subsets of data points) to serve as basis matrices in low-rank approximations. It has been empirically shown to work reasonably well in practice. The Nyström method is highly efficient in the sense that it can produce a low-rank approximation in complexity linear in the matrix size n (supposing the target approximation rank r is small).

For problems with high coherence [13, 34], the accuracy of the usual Nyström method with uniform sampling may be very low. There have been lots of efforts to improve the method. See, e.g., [10, 13, 24, 48]. In order to gain good accuracy, significant extra costs are needed to estimate leverage scores or determine sampling probabilities in nonuniform sampling [8, 11, 23].

Due to its modest accuracy, the Nyström method is usually used for data analysis and not much for regular numerical computations. In numerical analysis and scientific computing where controllable high accuracies are desired, often truncated SVDs or more practical variations like rank-revealing factorizations [6, 17] and randomized SVD/sketching methods [19, 35] are used. These methods can produce highly reliable low-rank approximations but usually cost $O(n^2)$ operations.

The purpose of this work is to propose a set of strategies based on the Nyström method to produce high-accuracy low-rank approximations for kernel matrices in about linear complexity. The matrices are allowed to be nonsymmetric and/or rectan-

*Submitted for review.

Funding: The research of Jianlin Xia was supported in part by an NSF grant DMS-2111007.

†Department of Mathematics, Purdue University, West Lafayette, IN 47907, USA (xiaj@purdue.edu).

gular. Examples include *off-diagonal blocks* of larger kernel matrices that frequently arise from numerical solutions of differential and integral equations, structured eigenvalue solutions, N-body simulations, and image processing. There has been a rich history in studying the low-rank structure of these off-diagonal kernel matrices based on ideas from the fast multipole method (FMM) [15] and hierarchical matrix methods [18]. To obtain a low-rank approximation to such a rectangular kernel matrix A with the Nyström method, a basic way is to choose respectively random row and column index sets $\mathcal{I}$ and $\mathcal{J}$ and then get a so-called CUR approximation

$$A \approx A_{:,\mathcal{J}} A^{+}_{\mathcal{I},\mathcal{J}} A_{\mathcal{I},:}, \tag{1.1}$$

where $A_{:,\mathcal{J}}$ and $A_{\mathcal{I},:}$ denote submatrices formed by the columns and rows of A corresponding to the index sets $\mathcal{J}$ and $\mathcal{I}$, respectively, and $A_{\mathcal{I},\mathcal{J}}$ can be understood similarly. However, the accuracy of (1.1) is typically low with random $\mathcal{I}$ and $\mathcal{J}$, unless the so-called volume of $A_{\mathcal{I},\mathcal{J}}$ happens to be sufficiently large [14]. It is well known that finding a submatrix with the maximum volume is NP-hard. See [20] for a comprehensive discussion of various viewpoints to understand the CUR decomposition and also see [21] for some perturbation error studies.

Here, we would like to design adaptive Nyström schemes that can produce controllable errors (including near machine precision) while still retaining nearly linear complexity in practice. We start by treating the combination of the Nyström method and a reliable algebraic rank-revealing factorization as a fast pivoting strategy to select significant rows/columns (called *representative rows/columns* as in [40]). We then provide one way to analyze the resulting low-rank approximation error, which serves as a motivation for the design of our new schemes. Further key strategies include the following.

1. Use selected columns and rows to perform fast *alternating direction row and column pivoting*, respectively, so as to refine selections of representative rows and columns.
2. Adaptively attach a small number of new samples so as to perform *progressive alternating direction pivoting*, which produces new expanded representative rows and columns and *advances the numerical rank* needed to reach high accuracies.
3. Use a fast subset update strategy that successively samples the Schur complements so as to improve the efficiency and accelerate the advancement of the sizes of basis matrix toward target numerical ranks.
4. Adaptively control the accuracy via quick estimation of the approximation errors.

Specifically, in the first strategy above, randomly selected columns are used to quickly perform row pivoting for A and obtain representative rows (which form a *row skeleton* $A_{\mathcal{I},:}$). The row skeleton is further used to quickly perform column pivoting for A to obtain some representative columns (which form a *column skeleton* $A_{:,\mathcal{J}}$). This refines the original choice of representative columns. Related methods include various forms of the adaptive cross approximation (ACA) with row/column pivoting [4, 26], the volume sampling approximation [8], and the iterative cross approximation [25]. In particular, the method in [25] iteratively refines selections of significant submatrices (with volumes as large as possible). However, later we can see that this strategy alone is not enough to reach high accuracy, even if a large number of initial samples is used.

Next in the second strategy, new column samples are attached progressively in small stepsizes so as to repeat the alternating direction pivoting until convergence is reached. Convenient uniform sampling is used since the sampled columns are for the

purpose of pivoting. This eliminates the need of estimating sampling probabilities. The third strategy enables to avoid applying pivoting to row/column skeletons with growing sizes. That is, the row (column) skeleton is expanded by quickly updating the previous skeleton when new columns (rows) are attached. We also give an aggressive subset update method that can quickly reach high accuracies with a small number of progressive sampling steps in practice. With the forth strategy, we can conveniently control the number of sampling steps until a desired accuracy is reached. It avoids the need to perform quadratic cost error estimation.

The combination of these strategies leads to a type of low-rank approximation schemes which we call *high-accuracy Nyström* (HAN) schemes. They are heuristic schemes that are both fast and accurate in practice. Although a fully rigorous justification of the accuracy is lacking, we give different perspectives to motivate and support the ideas. Relevant analysis is provided to understand certain singular value and accuracy behaviors in terms of both deterministic rank-revealing factorizations and statistical error evaluation.

We demonstrate the high accuracy of the HAN schemes through comprehensive numerical tests based on kernel matrices defined from various kernel functions evaluated at different data sets. In particular, an aggressive HAN scheme can produce approximation accuracies close to the quality of truncated SVDs. It is numerically shown to have nearly linear complexity and further usually needs just a surprisingly small number of sampling steps.

Additionally, the design of the HAN schemes does not require analytical information from the kernel functions or geometric information from the data points. They can then serve as fully blackbox fast low-rank approximation methods, as indicated in the tests.

The remaining discussions are organized as follows. We show the pivoting strategy based on the Nyström method and give a way to study the approximation error in Section 2. The detailed design of the HAN schemes together with relevant analysis is given in Section 3. Section 4 presents the numerical tests, followed by some concluding remarks in Section 5. Throughout the paper, we use capital letters like A and K for matrices, calligraphic letters like $\mathcal{I}$ and $\mathcal{J}$ for index sets, and bold letters like $\mathbf{x}$ and $\mathbf{y}$ for point sets.

2. Pivoting based on the Nyström method and an error study. We first consider a low-rank approximation method based on a pivoting strategy consisting of the Nyström method and rank-revealing factorizations of tall and skinny matrices. A way to study the low-rank approximation error will then be given. These will provide motivations for some of our ideas in the HAN schemes.

Consider two sets of real data points in d dimensions: $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$, $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$. Let A be the $m \times n$ kernel matrix

$$A = (\kappa(x_i, y_j))_{x_i \in \mathbf{x}, y_j \in \mathbf{y}}, \tag{2.1}$$

which is sometimes also referred to as the interaction matrix between $\mathbf{x}$ and $\mathbf{y}$. We would like to approximate A by a low-rank form. The strong rank-revealing QR or LU factorizations [17, 28] are reliable ways to find low-rank approximations with high accuracy. They may be used to obtain an approximation (called interpolative decomposition) of the following form:

$$A \approx A_{:,\mathcal{J}} V^T \quad \text{with} \quad V = Q \begin{pmatrix} I \\ F \end{pmatrix}, \tag{2.2}$$

where Q is a permutation matrix, $r \equiv |\mathcal{J}|$ (size or cardinality of $\mathcal{J}$) is the approximate (or numerical) rank, and $\|F\|_{\max} \le c$ with $c \ge 1$. c is a user-specified parameter and may be set to be a constant or a low-degree polynomial of m, n, and r [17].

We suppose r is small. The column skeleton $A_{:,\mathcal{J}}$ corresponds to a subset $\mathbf{t} \subset \mathbf{y}$ which is a subset of *landmark points*. Here we also call $\mathbf{t}$ a representative subset, which can be selected reliably by strong rank-revealing factorizations. A strong rank-revealing factorization may be further applied to $A_{:,\mathcal{J}}^T$ to select a representative subset $\mathbf{s} \subset \mathbf{x}$ corresponding to a row index set $\mathcal{I}$ in $A_{:,\mathcal{J}}$. That is, we can find a *pivot block* $A_{\mathcal{I},\mathcal{J}}$. Without loss of generality, assume $|\mathcal{I}| = |\mathcal{J}| = r$. (If the factorization produces $\mathcal{I}$ with $|\mathcal{I}| < |\mathcal{J}|$, V can be modified so as to replace $\mathcal{J}$ by an appropriate index set with size $|\mathcal{I}|$.) Thus, the resulting decomposition may be written as an equality

$$A_{:,\mathcal{J}} = U A_{\mathcal{I},\mathcal{J}} \quad \text{with} \quad U = P \begin{pmatrix} I \\ E \end{pmatrix}, \tag{2.3}$$

where P is a permutation matrix and, with $1:m$ standing for $1, 2, \ldots, m$,

$$E = A_{\{1:m\}\setminus\mathcal{I},\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1}, \quad \|E\|_{\max} \le c. \tag{2.4}$$

Since $A_{:,\mathcal{J}}$ is a tall and skinny matrix, we refer to (2.3) as a *skinny rank-revealing (SRR) factorization*. (2.2) and (2.3) in turn lead to the approximation

$$A \approx U A_{\mathcal{I},\mathcal{J}} V^T. \tag{2.5}$$

With (2.5), we may further obtain a CUR approximation like in (1.1) (with the pseudoinverse replaced by $A_{\mathcal{I},\mathcal{J}}^{-1}$).

The direct application of strong rank-revealing factorizations to A to obtain (2.2) is expensive and costs $O(rmn)$. To reduce the cost, we can instead follow the Nyström method and randomly sample columns from A to form $A_{:,\mathcal{J}}$. However, the accuracy of the resulting approximation based on the forms (2.2) or (2.5) may be low. On the other hand, we can view the SRR factorization (2.3) as a way to quickly choose the representative subset $\mathbf{s}$ (based on the interaction between $\mathbf{x}$ and $\mathbf{t}$ instead of the interaction between $\mathbf{x}$ and $\mathbf{y}$). In other words, (2.3) is a way to quickly perform *row pivoting* for A so as to select representative rows $A_{\mathcal{I},:}$ from A. Then we can use the following low-rank approximation:

$$A \approx U A_{\mathcal{I},:} \quad \text{with} \quad U = P \begin{pmatrix} I \\ E \end{pmatrix}, \tag{2.6}$$

which may be viewed as a potentially refined form over (2.2) when $\mathcal{J}$ is randomly selected. (Note that P and E depend on $\mathcal{J}$.)

We would like to gain some insights into the accuracy of approximations based on the Nyström method. There are various earlier studies based on (1.1). Those in [5, 48] are relevant to our result below. When A is positive (semi-)definite, the analysis in [48] bounds the errors in terms of the distances between the landmark points and the remaining data points. A similar strategy is also followed in [5, Lemma 3.1] for symmetric A. The resulting bound may be very conservative since it is common for some data points in practical data sets to be far away from the landmark points. In addition, the error bounds in [5, 48] essentially involve a factor $\|A_{\mathcal{I},\mathcal{J}}^{-1}\|_2$ (or $\|A_{\mathcal{I},\mathcal{J}}^{+}\|_2$), which may be too large if high accuracy is desired. This is because the smallest singular value of A_{11} may be just slightly larger than a smaller tolerance.

Here, we provide a way to understand the approximation error based on (2.6). It uses the minimization of a slightly overdetermined problem and does not involve $\|A_{\mathcal{I},\mathcal{J}}^{-1}\|_2$. The following analysis does not aim to precisely quantify the error magnitude (which is hard anyway). Instead, it can serve as a motivation for some strategies in our high-accuracy Nyström methods later.

LEMMA 2.1. *Suppose $\mathcal{J}$ is a given column index set with $|\mathcal{J}| = r$ and (2.3)–(2.4) hold. Then the resulting approximation (2.6) satisfies*

$$\|A - UA_{\mathcal{I},:}\|_{\max} \le 2c\sqrt{r} \max_{1\le i\le m, 1\le j\le n} \min_{v\in\mathbb{R}^r} \|A_{\widetilde{\mathcal{I}}_i,\mathcal{J}} v - A_{\widetilde{\mathcal{I}}_i,j}\|_2, \tag{2.7}$$

where $\widetilde{\mathcal{I}}_i = \mathcal{I} \cup \{i\}$ for each $1 \le i \le m$.

Proof. From (2.3) and (2.6), we have, for any $1 \le i \le m$, $1 \le j \le n$,

$$(A - UA_{\mathcal{I},:})_{ij} = (A - A_{:,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1} A_{\mathcal{I},:})_{ij} = A_{ij} - A_{i,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1} A_{\mathcal{I},j}.$$

It is obvious that $A_{ij} - A_{i,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1} A_{\mathcal{I},j} = 0$ if $i \in \mathcal{I}$. Thus, suppose $i \in \{1:m\}\backslash\mathcal{I}$.

For any $v \in \mathbb{R}^r$,

$$\begin{aligned} |A_{ij} - A_{i,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1} A_{\mathcal{I},j}| &= |(A_{ij} - A_{i,\mathcal{J}} v) + (A_{i,\mathcal{J}} v - A_{i,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1} A_{\mathcal{I},j})| \\ &\le |A_{ij} - A_{i,\mathcal{J}} v| + \|A_{i,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1}\|_2 \|A_{\mathcal{I},\mathcal{J}} v - A_{\mathcal{I},j}\|_2 \\ &\le |A_{ij} - A_{i,\mathcal{J}} v| + c\sqrt{r} \|A_{\mathcal{I},\mathcal{J}} v - A_{\mathcal{I},j}\|_2, \end{aligned}$$

where the last step is because $A_{i,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1}$ is a row of E in (2.4) and its entries have magnitudes bounded by c. With $c \ge 1$, we further have

$$\begin{aligned} |A_{ij} - A_{i,\mathcal{J}} A_{\mathcal{I},\mathcal{J}}^{-1} A_{\mathcal{I},j}| &\le c\sqrt{r}\,(|A_{i,\mathcal{J}} v - A_{ij}| + \|A_{\mathcal{I},\mathcal{J}} v - A_{\mathcal{I},j}\|_2) \\ &\le 2c\sqrt{r} \|A_{\widetilde{\mathcal{I}}_i,\mathcal{J}} v - A_{\widetilde{\mathcal{I}}_i,j}\|_2. \end{aligned}$$

Since this holds for all $v \in \mathbb{R}^r$, take the minimum for v to get the desired result. ∎

The bound in this lemma can be roughly understood as follows. If $A_{\widetilde{\mathcal{I}}_i,j}$ is nearly in the range of $A_{\widetilde{\mathcal{I}}_i,\mathcal{J}}$ for all i, j, the bound in (2.7) would then be very small and we would have found $\mathcal{I}$ and $\mathcal{J}$ that produce an accurate low-rank approximation (2.6). Otherwise, to further improve the accuracy, it would be necessary to refine $\mathcal{I}$ and $\mathcal{J}$ and possibly include additional i and j indices respectively into $\mathcal{I}$ and $\mathcal{J}$. A heuristic strategy is to progressively pick i and j so that $A_{\widetilde{\mathcal{I}}_i,j}$ is as linearly independent from the columns of $A_{\widetilde{\mathcal{I}}_i,\mathcal{J}}$ as possible. Motivated by this, we may use a subset refinement process. First, use randomly picked columns $A_{:,\mathcal{J}}$ to generate a row skeleton and then use the row skeleton to generate a new column skeleton. The new column skeleton suggests which new j should be attached to $\mathcal{J}$. Next, if a desired accuracy is not reached, then randomly pick more columns to attach to the refined set $\mathcal{J}$ and start a new round of refinement. Such a process is called *progressive alternating direction pivoting* (or *subset refinement*) below.

Remark 2.2. It is worth pointing out that there are some interesting recent perspectives on the CUR decomposition and its perturbation error analysis in [20, 21]. For example, the approximation of A by a CUR form $A_{:,\mathcal{J}} X A_{\mathcal{I},:}$ may be based on choices like (1.1) or $X = A_{:,\mathcal{J}}^{+} A A_{\mathcal{I},:}^{+}$. While these two choices are equivalent in the exact rank-r case (when A and $A_{:,\mathcal{J}}$ both have rank r), the latter one is actually optimal in Frobenius norm. Thus, we may potentially obtain improved error results

by analyzing $A_{:,\mathcal{J}}(A^{+}_{:,\mathcal{J}}AA^{+}_{\mathcal{I},:})A_{\mathcal{I},:}$ for the Nyström method. On the other hand, as pointed out in [21], the study in [21] focuses on generic error bounds for CUR approximations with given $\mathcal{I}$ and $\mathcal{J}$ without referring to particular algorithms for selecting them. Our focus here is on an adaptive pivoting and sampling strategy for selecting $\mathcal{I}$ and $\mathcal{J}$. Also, our target approximation involves basis matrices like U in (2.3) and V in (2.2) which are the outcome of SRR factorizations that have quality control like (2.4). Such basis matrices are also very convenient for operations like zeroing out submatrices as needed in structured matrix factorizations in [41, 45]. In addition, an approximation $A_{:,\mathcal{J}}XA_{\mathcal{I},:}$ with $X = A^{+}_{:,\mathcal{J}}AA^{+}_{\mathcal{I},:}$ is not used here since the cost to form this X would be quadratic in the size of A.

3. High-accuracy Nyström schemes. In this section, we show how to use the Nyström method to design the high-accuracy Nyström (HAN) schemes that can produce highly accurate low-rank approximations in practice. We begin with the basic idea of the progressive alternating direction pivoting and then show how to perform fast subset update and how to conveniently control the accuracy.

3.1. Progressive alternating direction pivoting. The direct application of strong rank-revealing factorizations to A has quadratic complexity. One way to save the cost is as follows. Start from some column samples of A like in the usual Nyström method. Use the SRR factorization to select a row skeleton, which can then be used to select a refined column skeleton. The process can be repeated in a recursive way, leading to a fast alternating direction refinement scheme. A similar empirical scheme has been adopted recently in [25, 31]. However, when high accuracies are desired, the effectiveness of this scheme may be limited. That is, just like the usual Nyström method, a brute-force increase of the initial sample size may not necessarily improve the approximation accuracy significantly. A high accuracy may require the initial sample size to be overwhelmingly larger than the target numerical rank, which makes the cost too high.

Here, we instead adaptively or progressively apply the alternating direction refinement based on step-by-step small increases of the sample size. We use one round of alternating row and column pivoting to refine the subset selections. After this, if a target accuracy τ or numerical rank r is not reached, we include a small number of additional samples to repeat the procedure.

Uniform sampling is used in all of our sampling steps since we need a fast sampling method that does not require problem specific information. Here, uniform sampling suffices in practice since our purpose is to use the sampled columns just for row pivoting and they are not directly used in basis matrices. It is possible to improve the accuracy by using nonuniform sampling based on the estimation of leverage scores or sampling probabilities [8, 11, 23]. However, this would add extra costs in operations like column norm estimation, formation of certain intermediate matrices like Schur complements, etc. They might destroy the desired linear complexity. Also, we may not be able to use diagonal entries to decide sampling probabilities since A may not be square.

The basic framework to find a low-rank approximation to A in (2.1) is as follows, where the subset $\mathcal{J}$ is initially an empty set and $b \le r$ is a small integer as the *stepsize* in the progressive column sampling.

1. (*Progressive sampling*) Randomly choose a column index set $\widehat{\mathcal{J}} \subset \{1:n\}\backslash\mathcal{J}$ with $|\widehat{\mathcal{J}}| = b$ and set
$$\widetilde{\mathcal{J}} = \mathcal{J} \cup \widehat{\mathcal{J}}.$$

2. (*Row pivoting*) Apply an SRR factorization to $A_{:,\widetilde{\mathcal{J}}}$ to find a row index set $\mathcal{I}$:

$$A_{:,\widetilde{\mathcal{J}}} \approx U A_{\mathcal{I},\widetilde{\mathcal{J}}}, \tag{3.1}$$

   where U looks like that in (2.3).
3. (*Column pivoting*) Apply an SRR factorization to $A_{\mathcal{I},:}$ to find a refined column index set $\mathcal{J}$:

$$A_{\mathcal{I},:} \approx A_{\mathcal{I},\mathcal{J}} V^T, \tag{3.2}$$

   where V looks like that in (2.2).
4. (*Accuracy check*) If a desired accuracy, maximum sample size, or a target numerical rank is reached or if $\mathcal{I}$ stays the same as in the previous step, return a low-rank approximation to A like the following and exit:

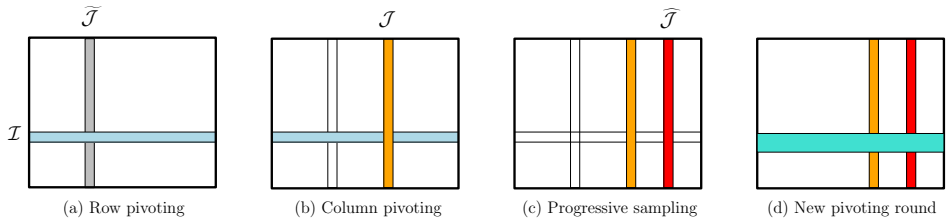
$$\tilde{A} = U A_{\mathcal{I},:}, \quad A_{:,\mathcal{J}} V^T, \quad \text{or} \quad U A_{\mathcal{I},\mathcal{J}} V^T.$$

   Otherwise, repeat from Step 1. (More details on the stopping criteria and fast error estimation will be given in Section 3.3.)

This basic HAN scheme (denoted HAN-B) is illustrated in Figure 3.1, with more details given in Algorithm 3.1. Note that the key outputs of the SRR factorization (2.3) are the index set $\mathcal{I}$ and the matrix E. (The permutation matrix P is just to bring the index set $\mathcal{I}$ to the leading part and does not need to be stored.) For convenience, we denote (2.3) by the following procedure in Algorithm 3.1 (with the parameter c in (2.4) assumed to be fixed):

$$[\mathcal{I}, E] \leftarrow \mathsf{SRR}(A_{:,\mathcal{J}}).$$

The scheme may be understood heuristically as follows. Initially, with $\widetilde{\mathcal{J}}$ a random sample from the column indices, it is known that the expectation of the norm of a row of $A_{:,\widetilde{\mathcal{J}}}$ is a multiple of the norm of the corresponding row in A (see, e.g., [1, 9]). Thus, the relative magnitudes of the row norms of A can be roughly reflected by those of $A_{:,\widetilde{\mathcal{J}}}$. It then makes sense to use $A_{:,\widetilde{\mathcal{J}}}$ for quick row pivoting (by finding $A_{\mathcal{I},\widetilde{\mathcal{J}}}$ with determinant as large as possible). This strategy shares features similar to the randomized pivoting strategies in [27, 46] which are also heuristic and work well in practice, except that the methods in [27, 46] need matrix-vector multiplications with costs $O(mn)$.



(a) Row pivoting (b) Column pivoting (c) Progressive sampling (d) New pivoting round

FIG. 3.1. *Illustration of the basic high-accuracy Nyström (HAN-B) scheme.*

With the resulting row pivot index set $\mathcal{I}$, the scheme further uses the SRR factorization to find a submatrix $A_{\mathcal{I},\mathcal{J}}$ of $A_{\mathcal{I},:}$ with determinant as large as possible, which enables to refine the column selection. It may be possible to further improve the index sets through multiple rounds of such refinements like in [25, 31]. However,

Algorithm 3.1 Basic HAN scheme

procedure HAN-B(A, b, τ (or r))
  $\mathcal{J} \leftarrow \varnothing$
  for $i = 1, 2, \ldots$ **do**
    $\widehat{\mathcal{J}} \leftarrow$ randomly chosen index set from $\{1 : n\} \backslash \mathcal{J}$ with $|\widehat{\mathcal{J}}| = b$
    $\widetilde{\mathcal{J}} \leftarrow \mathcal{J} \cup \widehat{\mathcal{J}}$ ▷ *Progressive sampling*
    $\overline{\mathcal{I}} \leftarrow \mathcal{I}$
    $[\mathcal{I}, E] \leftarrow \mathsf{SRR}(A_{:,\widetilde{\mathcal{J}}})$ ▷ *Row pivoting*
    $\widehat{\mathcal{I}} \leftarrow \mathcal{I} \backslash \overline{\mathcal{I}}$ ▷ *Removal of indices in the previous $\mathcal{I}$ from the new one*
    $[\mathcal{J}, F] \leftarrow \mathsf{SRR}(A^T_{\mathcal{I},:})$ ▷ *Column pivoting*
    if a target accuracy τ or numerical rank r is reached or $\widehat{\mathcal{I}} = \varnothing$ **then**
      ▷ *See Section 3.3 for more discussions*
      Return $A \approx U A_{\mathcal{I},:}$, $A_{:,\mathcal{J}} V^T$, or $U A_{\mathcal{I},\mathcal{J}} V^T$
    end if
  end for
end procedure

the accuracy gain seems limited, even if a large initial sample size is used (as shown in our test later). Thus, we progressively attach additional samples (in small stepsizes) to the refined subset $\mathcal{J}$ and then repeat the previous procedure. In practice, this makes a significant difference in reducing the approximation error.

In this scheme, the sizes of the index sets $\mathcal{I}$ and $\mathcal{J}$ grow with the progressive sampling. Accordingly, the costs of the SRR factorizations (3.1)–(3.2) increase since the SRR factorizations at step i are applied to matrices of sizes $m \times (ib)$ or $(ib) \times n$. With the total number of iterations $N \approx \frac{r}{b}$, the total cost (excluding the cost to check the accuracy) is

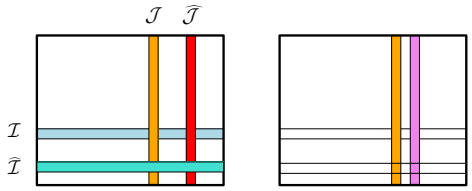
$$\xi_{\mathsf{HAN-B}} = \sum_{i=1}^{N} O\left((ib)^2(m+n)\right) = O\left(\frac{r^3}{b}(m+n)\right). \tag{3.3}$$

With i increases, the iterations advance toward the target numerical rank or accuracy.

3.2. Fast subset update via Schur complement sampling. In the basic scheme HAN-B, the complexity count in (3.3) for the SRR factorizations at step i gets higher with increasing i. To improve the efficiency, we show how to update the index sets so that at step i, the SRR factorization (for the row pivoting step for example) only needs to be applied to a matrix of size $(m - (i-1)b) \times b$ instead of $m \times (ib)$, followed by some quick postprocessing steps.

Suppose we start from a column index set $\widetilde{\mathcal{J}} = \mathcal{J} \cup \widehat{\mathcal{J}}$ as in Step 1 of the basic HAN scheme above. We would like to avoid applying the SRR factorizations to the full columns $A_{:,\widetilde{\mathcal{J}}}$ in Step 2 and the full rows $A_{\mathcal{I},:}$ in Step 3. We seek to directly produce an expanded column index set over $\mathcal{J}$, as illustrated in Figure 3.2. It includes two steps. One is to produce an update $\widehat{\mathcal{I}}$ to the row index set $\mathcal{I}$ (Figure 3.2(a), which replaces Steps (c)–(d) in Figure 3.1) and the other is to produce an update to the column index set (Figure 3.2(b)). Clearly, we just need to show how to perform the first step.

With the row pivoting step like in (3.1), we can obtain a low-rank approximation of the form (2.6). Using the row permutation matrix P in (2.6) (computed in (2.3)),



(a) Row subset update (b) Column subset update

FIG. 3.2. *Illustration of the subset update process.*

we may write A as

$$A = P \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \approx P \begin{pmatrix} I \\ E \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \end{pmatrix} \quad \text{with} \tag{3.4}$$
$$A_{11} = A_{\mathcal{I},\mathcal{J}}, \quad E = A_{21}A_{11}^{-1}.$$

At this point, we have

$$A = P \begin{pmatrix} I & \\ E & I \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ & S \end{pmatrix} \quad \text{with} \quad S = A_{22} - EA_{12}, \tag{3.5}$$

where S is the *Schur complement*.

Remark 3.1. In the usual strong rank-revealing factorizations like the one in [28], the low-rank approximation is obtained also from a decomposition of the form (3.5) with S dropped. Here, our fast pivoting scheme is more efficient. Of course, the strong rank-revealing factorization in [28] guarantees the quality of the low-rank approximation in the sense that, there exist low-degree polynomials $c \geq 1$ and $f \geq 1$ in m, n, and k (size of A_{11}) such that (2.4) holds and, for $1 \leq i \leq k$, $1 \leq j \leq \min\{m,n\} - k$,

$$\sigma_i(A_{11}) \geq \frac{\sigma_i(A)}{f}, \quad \sigma_j(S) \leq \sigma_{k+j}(A)f, \quad \|A_{11}^{-1}A_{12}\|_{\max} \leq c, \tag{3.6}$$

where $\sigma_i(\cdot)$ denotes the i-th largest singular value of a matrix.

Our subset update strategy is via the sampling of the Schur complement S. In fact, when $A_{\mathcal{I},:} = \begin{pmatrix} A_{11} & A_{12} \end{pmatrix}$ is accepted as a reasonable row skeleton, we then continue to find a low-rank approximation to S in (3.5) so it makes sense to sample S. It is worth noting that the full matrix S is not needed. Instead, only its columns corresponding to $A_{:,\widehat{\mathcal{J}}}$ are formed. That is, we form

$$S_{:,\mathcal{L}} = (A_{22})_{:,\mathcal{L}} - E(A_{12})_{:,\mathcal{L}},$$

where $\mathcal{L}$ corresponds to $\widehat{\mathcal{J}}$ and selects entries from $\{1:n\}\backslash\mathcal{J}$ in a two-level composition of the index sets as follows:

$$(\{1:n\}\backslash\mathcal{J}) \circ \mathcal{L} = \widehat{\mathcal{J}}. \tag{3.7}$$

That is, sampling the columns of A with the index set $\widehat{\mathcal{J}}$ is essentially to sample the columns of S with $\mathcal{L}$. For notational convenience, suppose the columns of A have been permuted so that

$$P^T A = \left(\begin{array}{c|c} A_{11} & A_{12} \\ A_{21} & A_{22} \end{array}\right) \quad \text{with} \quad A_{:,\mathcal{J}} = \begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix}.$$

Now, apply an SRR factorization to $S_{:,\mathcal{L}}$ to get

$$S_{:,\mathcal{L}} \approx \hat{P} \begin{pmatrix} I \\ \hat{E} \end{pmatrix} S_{\mathcal{K},\mathcal{L}}. \tag{3.8}$$

Then $S \approx \hat{P} \begin{pmatrix} I \\ \hat{E} \end{pmatrix} S_{\mathcal{K},:}$. Accordingly, we may write S as

$$S = \hat{P} \begin{pmatrix} S_{22} & S_{23} \\ S_{32} & S_{33} \end{pmatrix}, \tag{3.9}$$

where $S_{\mathcal{K},:} = \begin{pmatrix} S_{22} & S_{23} \end{pmatrix}$. From (3.8) and (3.9), S can be further written as

$$S = \hat{P} \begin{pmatrix} I & \\ \hat{E} & \hat{S} \end{pmatrix} \begin{pmatrix} S_{22} & S_{23} \\ & I \end{pmatrix} \quad \text{with} \quad \hat{S} = S_{33} - \hat{E} S_{23}, \tag{3.10}$$

where $\hat{S}$ is a new Schur complement (and is not formed).

At this point, we have the following proposition which shows how to expand the row index set $\mathcal{I}$ by an update $\widehat{\mathcal{I}}$.

PROPOSITION 3.2. *A may be factorized as*

$$A = \tilde{P} \begin{pmatrix} I & \\ \tilde{E} & I \end{pmatrix} \begin{pmatrix} A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}} & A_{\widetilde{\mathcal{I}},\{1:n\}\setminus\widetilde{\mathcal{J}}} \\ & \hat{S} \end{pmatrix}, \tag{3.11}$$

where $\tilde{P}$ is a permutation matrix, $\widetilde{\mathcal{I}} = \mathcal{I} \cup \widehat{\mathcal{I}}$ with

$$\widehat{\mathcal{I}} = (\{1:m\}\setminus\mathcal{I}) \circ \mathcal{K}, \tag{3.12}$$

and $\|\tilde{E}\|_{\max} \le bc^2 + c$ when $\|E\|_{\max} \le c$, $\|\hat{E}\|_{\max} \le c$, and $|\widehat{\mathcal{J}}| = b$.

Proof. (3.5) and (3.10) lead to

$$\begin{aligned} A &= P \begin{pmatrix} I & \\ E & I \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ & \hat{P} \begin{pmatrix} I & \\ \hat{E} & \hat{S} \end{pmatrix} \begin{pmatrix} S_{22} & S_{23} \\ & I \end{pmatrix} \end{pmatrix} \\ &= P \begin{pmatrix} I & \\ E & \hat{P} \begin{pmatrix} I & \\ \hat{E} & \hat{S} \end{pmatrix} \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ & \begin{pmatrix} S_{22} & S_{23} \\ & I \end{pmatrix} \end{pmatrix} \\ &= P \begin{pmatrix} I & \\ & \hat{P} \end{pmatrix} \begin{pmatrix} I & \\ \hat{P}^T E & \begin{pmatrix} I & \\ \hat{E} & \hat{S} \end{pmatrix} \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ & \begin{pmatrix} S_{22} & S_{23} \\ & I \end{pmatrix} \end{pmatrix} \\ &= \tilde{P} \begin{pmatrix} I & & \\ E_1 & I & \\ E_2 & \hat{E} & \hat{S} \end{pmatrix} \begin{pmatrix} A_{11} & \hat{A}_{12} & \hat{A}_{13} \\ & S_{22} & S_{23} \\ & & I \end{pmatrix}, \end{aligned} \tag{3.13}$$

where $\tilde{P} = P \begin{pmatrix} I & \\ & \hat{P} \end{pmatrix}$, $\hat{P}^T E$ is partitioned as $\begin{pmatrix} E_1 \\ E_2 \end{pmatrix}$ conformably with $\begin{pmatrix} I \\ \hat{E} \end{pmatrix}$, and A_{12} is partitioned as $\begin{pmatrix} \hat{A}_{12} & \hat{A}_{13} \end{pmatrix}$ with $\hat{A}_{12} = A_{\mathcal{I},\widehat{\mathcal{J}}}$.

We can now factorize the second factor on the far right-hand side of (3.13) as

$$\begin{pmatrix} I & & \\ E_1 & I & \\ E_2 & \hat{E} & \hat{S} \end{pmatrix} = \begin{pmatrix} I & & \\ & I & \\ \bar{E} & \hat{E} & \hat{S} \end{pmatrix} \begin{pmatrix} I & & \\ E_1 & I & \\ & & I \end{pmatrix},$$

where

$$\bar{E} = E_2 - \hat{E}E_1. \tag{3.14}$$

Then, A may be written as

$$\begin{aligned} A &= \tilde{P} \begin{pmatrix} I & & \\ & I & \\ \bar{E} & \hat{E} & \hat{S} \end{pmatrix} \begin{pmatrix} I & & \\ E_1 & I & \\ & & I \end{pmatrix} \begin{pmatrix} A_{11} & \hat{A}_{12} & \hat{A}_{13} \\ & S_{22} & S_{23} \\ & & I \end{pmatrix} \\ &= \tilde{P} \begin{pmatrix} I & & \\ & I & \\ \bar{E} & \hat{E} & I \end{pmatrix} \begin{pmatrix} A_{11} & \hat{A}_{12} & \hat{A}_{13} \\ \hat{A}_{21} & \hat{A}_{22} & \hat{A}_{23} \\ & & \hat{S} \end{pmatrix} \end{aligned} \tag{3.15}$$

where

$$\begin{pmatrix} \hat{A}_{21} & \hat{A}_{22} & \hat{A}_{23} \end{pmatrix} = \begin{pmatrix} 0 & S_{22} & S_{23} \end{pmatrix} + E_1 \begin{pmatrix} A_{11} & \hat{A}_{12} & \hat{A}_{13} \end{pmatrix}.$$

The block $\begin{pmatrix} \hat{A}_{21} & \hat{A}_{22} & \hat{A}_{23} \end{pmatrix}$ essentially corresponds to the rows of A with index set $\widehat{\mathcal{I}}$ in (3.12). This is because of the special form of the second factor on the right-hand side of (3.15). Then get (3.11) by letting

$$\tilde{E} = \begin{pmatrix} \bar{E} & \hat{E} \end{pmatrix}.$$

Now since $\|E\|_{\max} \le c$, $\|\hat{E}\|_{\max} \le c$, and $\hat{E}$ has column size b, we have $\|\bar{E}\|_{\max} \le bc^2 + c$. Accordingly, $\|\tilde{E}\|_{\max} \le bc^2 + c$. ∎

This proposition shows that we can get a factorization (3.11) similar to (3.5), but with the expanded row skeleton $A_{\widetilde{\mathcal{I}},:}$. Accordingly, we may then obtain a new approximation to A similar to (2.6):

$$A \approx \tilde{U} A_{\widetilde{\mathcal{I}},:} \quad \text{with} \quad \tilde{U} = \tilde{P} \begin{pmatrix} I \\ \tilde{E} \end{pmatrix}, \tag{3.16}$$

To support the reliability of such an approximation, we can use the following way. As mentioned in Remark 3.1, if (3.5) is assumed to be obtained by a strong rank-revealing factorization, then we would have nice singular value bounds in (3.6). Now, if we assume that is the case and (3.10) is also obtained by a strong rank-revealing factorization, then we would like to show (3.11) from the subset update would also satisfy some nice singular value bounds. For this purpose, we need the following lemma.

LEMMA 3.3. *If (3.5) is assumed to satisfy (3.6) with $k = |\mathcal{I}|$ the size of A_{11}, and (3.10) is assumed to satisfy, for $1 \le i \le b$, $1 \le j \le \min\{m, n\} - k - b$,*

$$\sigma_i(S_{22}) \ge \frac{\sigma_i(S)}{\hat{f}}, \quad \sigma_j(\hat{S}) \le \sigma_{b+j}(S)\hat{f}, \tag{3.17}$$

where $\hat{f} \geq 1$, *then* $\mu = \frac{\sigma_k(A_{11})}{\sigma_1(S_{22})}$ *satisfies*

$$\frac{1}{f^2} \leq \mu \leq s\hat{f},$$

where $s = \frac{\sigma_k(A)}{\sigma_{k+1}(A)}$.

Proof. By (3.6) and the interlacing property of singular values,

$$\sigma_k(A_{11}) \geq \frac{\sigma_k(A)}{f}, \quad \sigma_1(S_{22}) \leq \sigma_1(S) \leq \sigma_{k+1}(A)f,$$

which yield

$$\mu \geq \frac{\sigma_k(A)/f}{\sigma_{k+1}(A)f} \geq \frac{1}{f^2}.$$

Similarly, by the interlacing property of singular values and (3.17),

$$\sigma_k(A_{11}) \leq \sigma_k(A), \quad \sigma_1(S_{22}) \geq \frac{\sigma_1(S)}{\hat{f}} \geq \frac{\sigma_{k+1}(A)}{\hat{f}},$$

where the result $\sigma_1(S) \geq \sigma_{k+1}(A)$ directly follows from Weyl's inequality or [22, Theorem 3.3.16]:

$$\begin{aligned}\sigma_{k+1}(A) &= \sigma_{k+1}\left(P\begin{pmatrix} I \\ E \end{pmatrix}\begin{pmatrix} A_{11} & A_{12} \end{pmatrix} + P\begin{pmatrix} 0 & \\ & S \end{pmatrix}\right) \\ &\leq \sigma_{k+1}\left(P\begin{pmatrix} I \\ E \end{pmatrix}\begin{pmatrix} A_{11} & A_{12} \end{pmatrix}\right) + \sigma_1\left(P\begin{pmatrix} 0 & \\ & S \end{pmatrix}\right) = 0 + \sigma_1(S).\end{aligned}$$

Then $\mu \leq \frac{\sigma_k(A)}{\sigma_{k+1}(A)/\hat{f}} = s\hat{f}$. ∎

As a quick note, here $s = \frac{\sigma_k(A)}{\sigma_{k+1}(A)}$ reflects the gap between $\sigma_k(A)$ and $\sigma_{k+1}(A)$. Since we seek to expand the index sets $\mathcal{I}$ and $\mathcal{J}$ (and k hasn't yet reached the target numerical rank r), it is reasonable to regard s as a modest magnitude. Now we are ready to show the singular value bounds.

PROPOSITION 3.4. *With the assumptions and notation in Lemma 3.3, (3.11) satisfies, for* $1 \leq i \leq k+b$, $1 \leq j \leq \min\{m,n\} - k - b$,

$$\sigma_i(A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}}) \geq \frac{\sigma_i(A)}{\tilde{f}}, \quad \sigma_j(\hat{S}) \leq \sigma_{k+b+j}(A)\tilde{f}, \tag{3.18}$$

where $\tilde{f} = (1 + s\hat{f} + s\hat{f}b^2c^2)f^2\hat{f}$.

Proof. According to (3.15), $A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}} = \begin{pmatrix} I & \\ E_1 & I \end{pmatrix}\begin{pmatrix} A_{11} & \hat{A}_{12} \\ & S_{22} \end{pmatrix}$. With a strategy like in [17, 28], rewrite $A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}}$ as

$$A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}} = \begin{pmatrix} I & \\ E_1 & \frac{1}{\sqrt{\mu}}I \end{pmatrix}\begin{pmatrix} A_{11} & \\ & \mu S_{22} \end{pmatrix}\begin{pmatrix} I & A_{11}^{-1}\hat{A}_{12} \\ & \frac{1}{\sqrt{\mu}}I \end{pmatrix},$$

where $\mu = \frac{\sigma_k(A_{11})}{\sigma_1(S_{22})}$. Then

$$\begin{pmatrix} A_{11} & \\ & \mu S_{22} \end{pmatrix} = \begin{pmatrix} I & \\ -\sqrt{\mu}E_1 & \sqrt{\mu}I \end{pmatrix} A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}} \begin{pmatrix} I & -\sqrt{\mu}A_{11}^{-1}\hat{A}_{12} \\ & \sqrt{\mu}I \end{pmatrix}$$

By [22, Theorem 3.3.16],

$$\begin{aligned}\sigma_i\left(\begin{pmatrix} A_{11} & \\ & \mu S_{22}\end{pmatrix}\right) &\le \sigma_i(A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}}) \left\|\begin{pmatrix} I & \\ -\sqrt{\mu}E_1 & \sqrt{\mu}I\end{pmatrix}\right\|_2 \left\|\begin{pmatrix} I & -\sqrt{\mu}A_{11}^{-1}\hat{A}_{12} \\ & \sqrt{\mu}I\end{pmatrix}\right\|_2 \\ &\le \sigma_i(A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}})\sqrt{1+\mu+\mu\|E_1\|_2^2}\sqrt{1+\mu+\mu\|A_{11}^{-1}\hat{A}_{12}\|_2^2} \\ &\le \sigma_i(A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}})(1+s\hat{f}+s\hat{f}b^2c^2),\end{aligned}$$

where the last inequality is from Lemma 3.3 and the fact that E_1 and $A_{11}^{-1}\hat{A}_{12}$ are $b\times b$ matrices with entrywise magnitudes bounded by c. Thus,

$$\sigma_i(A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}}) \ge \frac{1}{1+s\hat{f}+s\hat{f}b^2c^2}\sigma_i\left(\begin{pmatrix} A_{11} & \\ & \mu S_{22}\end{pmatrix}\right). \tag{3.19}$$

Since $\sigma_k(A_{11}) = \sigma_1(\mu S_{22})$, we get

$$\begin{aligned}\sigma_i\left(\begin{pmatrix} A_{11} & \\ & \mu S_{22}\end{pmatrix}\right) &= \sigma_i(A_{11}), \quad 1\le i\le k, \\ \sigma_{k+i}\left(\begin{pmatrix} A_{11} & \\ & \mu S_{22}\end{pmatrix}\right) &= \sigma_i(\mu S_{22}), \quad 1\le i\le b.\end{aligned}$$

By (3.17) and Lemma 3.3,

$$\sigma_i(\mu S_{22}) \ge \mu\frac{\sigma_i(S)}{\hat{f}} \ge \frac{1}{f^2\hat{f}}\sigma_i(S) \ge \frac{1}{f^2\hat{f}}\sigma_{k+i}(A), \quad 1\le i\le b, \tag{3.20}$$

where the result $\sigma_i(S)\ge\sigma_{k+i}(A)$ again follows from Weyl's inequality or [22, Theorem 3.3.16]. Putting (3.20) and the first inequality in (3.6) into (3.19) to get

$$\sigma_i(A_{\widetilde{\mathcal{I}},\widetilde{\mathcal{J}}}) \ge \frac{1}{1+s\hat{f}+s\hat{f}b^2c^2}\frac{1}{f^2\hat{f}}\sigma_i(A), \quad 1\le i\le k+b.$$

Finally, (3.17) and (3.6) yield

$$\sigma_j(\hat{S}) \le \sigma_{b+j}(S)\hat{f} \le \sigma_{k+b+j}(A)f\hat{f}, \quad 1\le j\le \min\{m,n\}-k-b.$$

Then taking $\tilde{f} = (1+s\hat{f}+s\hat{f}b^2c^2)f^2\hat{f}$ to get (3.18). ∎

This proposition indicates that, if (3.6) and (3.17) are assumed to result from strong rank-revealing factorizations, then (3.11) as produced by the subset update method would also enjoy nice singular value properties like in a strong rank-revealing factorization. This supports the effectiveness of performing the subset update. Although here we obtain (3.6) and (3.17) through the much more economic SRR factorizations coupled with the Nyström method, it would be natural to use subset updates to quickly get the expanded index set $\widetilde{\mathcal{I}}$ (from the original index set $\mathcal{I}$). The SRR factorizations are only applied to blocks with column sizes b instead of ib in step i.

In a nutshell, the subset update process starts from a row skeleton $A_{\mathcal{I},:}$, samples the Schur complement S, and produces an expanded row skeleton $A_{\widetilde{\mathcal{I}},:}$ and the basis matrix $\tilde{U}$ in (3.16). The process is outlined in Algorithm 3.2.

Such a subset update strategy can also be applied to expand the column index set $\mathcal{J}$. That is, when $\mathcal{I}$ is expanded into $\mathcal{I}\cup\widehat{\mathcal{I}}$, we can apply the strategy above with $\mathcal{J}$ replaced by $\mathcal{I}$, $\widehat{\mathcal{J}}$ replaced by $\widehat{\mathcal{I}}$, and relevant columns replaced by rows.

We then incorporate the subset update strategy into the basic HAN scheme. There are two ways to do so with different performance (see Algorithm 3.3).

Algorithm 3.2 Subset update

procedure $[\widetilde{\mathcal{I}}, \tilde{E}] = \mathsf{Set\text{-}Upd}(A, \mathcal{I}, E, \widehat{\mathcal{J}})$
  Get $\mathcal{L}$ as in (3.7)
  $S_{:,\mathcal{L}} \leftarrow A_{\{1:m\}\backslash\mathcal{I},\widehat{\mathcal{J}}} - EA_{\mathcal{I},\widehat{\mathcal{J}}}$ ▷ *Sampling the Schur complement*
  $[\mathcal{K}, \hat{E}] \leftarrow \mathsf{SRR}(S_{:,\mathcal{L}})$
  $\begin{pmatrix} E_1 \\ E_2 \end{pmatrix} \leftarrow \begin{pmatrix} E_{\mathcal{K},:} \\ E_{\{1:m-k\}\backslash\mathcal{K},:} \end{pmatrix}$ ▷ $m-k$: *row size of* S *and* E
  $\bar{E} \leftarrow E_2 - \hat{E}E_1$
  $\widehat{\mathcal{I}} \leftarrow (\{1:m\}\backslash\mathcal{I}) \circ \mathcal{K}$
  $\widetilde{\mathcal{I}} \leftarrow \mathcal{I} \cup \widehat{\mathcal{I}}, \quad \tilde{E} \leftarrow \begin{pmatrix} \bar{E} & \hat{E} \end{pmatrix}$
end procedure

- HAN-U: This is an HAN scheme with fast updates for both the row subsets and the column subsets. Thus, both the index sets $\mathcal{I}$ and $\mathcal{J}$ are expanded through updates. In this scheme, $|\mathcal{I}|$ and $|\mathcal{J}|$ are each advanced by stepsize b in every iteration step.
- HAN-A: This is an HAN scheme with aggressive updates where only, say, the column index set $\mathcal{J}$ is updated. The row index set $\mathcal{I}$ is still updated via the usual SRR pivoting applied to $A_{:,\widetilde{\mathcal{J}}}$ (line 4 of Algorithm 3.3). This scheme potentially expands the index sets much more aggressively. The reason is as follows. The SRR factorization of $A_{:,\widetilde{\mathcal{J}}}$ may update $\mathcal{I}$ to a very different set and the set difference $\widehat{\mathcal{I}}$ (line 8 of Algorithm 3.3) may have size comparable to $|\mathcal{J}|$. Then, the column subset update is applied based on $A_{\widehat{\mathcal{I}},:}$ as in line 12 of Algorithm 3.3 and can potentially increase the size of $\mathcal{J}$ by $|\widehat{\mathcal{I}}|$.

Algorithm 3.3 HAN scheme with fast or aggressive subset update

procedure $\tilde{L} =$ HAN-U OR HAN-A(A, b, τ (or r))
  ⋮ ▷ *Keeping lines 2–6 & replacing lines 7–9 of Algorithm 3.1 by the following*
  if $i = 1$ or HAN-A **then** ▷ *Row pivoting in the initial step or in* HAN-A
    $[\mathcal{I}, E] \leftarrow \mathsf{SRR}(A_{:,\widetilde{\mathcal{J}}})$
  else ▷ *Row subset update in* HAN-U
    $[\mathcal{I}, E] \leftarrow \mathsf{Set\text{-}Upd}(A, \mathcal{I}, E, \widehat{\mathcal{J}})$
  end if
  $\widehat{\mathcal{I}} \leftarrow \mathcal{I}\backslash\overline{\mathcal{I}}$ ▷ *Removal of indices in the previous* $\mathcal{I}$ *from the new one*
  if $i = 1$ **then** ▷ *Column pivoting in the initial step*
    $[\mathcal{J}, F] \leftarrow \mathsf{SRR}(A^T_{\mathcal{I},:})$
  else ▷ *Column subset update*
    $[\mathcal{J}, F] \leftarrow \mathsf{Set\text{-}Upd}(A^T, \mathcal{J}, F, \widehat{\mathcal{I}})$
  end if
  ⋮ ▷ *Keeping the remaining lines of Algorithm 3.1*
end procedure

If Algorithm 3.2 is applied at the ith iteration of Algorithm 3.3 as in line 6, the main costs are as follows.

- The formation of $S_{:,\mathcal{L}}$ costs $O\left((m-(i-1)b)(i-1)b^2\right) + O\left((m-(i-1)b)b\right)$.
- The SRR factorization of $S_{:,\mathcal{L}}$ in (3.8) costs $O\left(b^2(m-(i-1)b)\right)$.

- The computation of (3.14) costs $O\left((m-ib)(i-1)b^2\right)+O\left((m-ib)(i-1)b\right)$.

These costs add up to $O(ib(2bm+m+2b^2))$, where some low-order terms are dropped and b is assumed to be a small fixed stepsize. The HAN-U scheme applies Algorithm 3.2 to both the row and the column subset updates. Accordingly, with $N \approx \frac{r}{b}$ iterations, the total cost of the HAN-U scheme is

$$\xi_{\mathsf{HAN-U}} = \sum_{i=1}^{N} O(ib(2b(m+n)+m+n+4b^2)) = O\left(r^2(m+n)\right),$$

which is a significant reduction over the cost in (3.3).

The cost of the HAN-A scheme depends on how many iteration steps are involved and on how aggressive the index sets advance. In the most aggressive case, suppose at each step the updated index set $\mathcal{I}$ (or $\mathcal{J}$) doubles the size from the previous step, then it only needs $\tilde{N} \approx \log_2 \frac{r}{b}$ steps. Accordingly, the cost is

$$\xi_{\mathsf{HAN-A}} = \sum_{i=1}^{\tilde{N}} O\left((2^{i-1}b)^2(m+n)\right) = O(r^2(m+n)),$$

which is comparable to $\xi_{\mathsf{HAN-U}}$. Moreover, in such a case, HAN-A would only need about $b\log_2 \frac{r}{b}$ column samples instead of about r samples, which makes it possible to find a low-rank approximation with a total sample size much smaller than r. This has been observed frequently in numerical tests (see Section 4).

3.3. Stopping criteria and adaptive accuracy control. The HAN schemes output both $\mathcal{I}$ and $\mathcal{J}$ so we may use $UA_{\mathcal{I},:}$, $A_{:,\mathcal{J}}V^T$, or $UA_{\mathcal{I},\mathcal{J}}V^T$ as the output low-rank approximation, where V and U look like those in (2.2) and (2.6), respectively. Based on the differences of the schemes, we use the following choice which works well in practice:

$$\tilde{A} = \begin{cases} A_{:,\mathcal{J}}V^T, & \mathsf{HAN\text{-}B} \text{ or } \mathsf{HAN\text{-}U}, \\ UA_{\mathcal{I},:}, & \mathsf{HAN\text{-}A}. \end{cases} \tag{3.21}$$

The reason is as follows. $A_{:,\mathcal{J}}V^T$ is the output from the end of the iteration and is generally a good choice. On the other hand, since HAN-A obtains U from a full strong rank-revealing factorization step which potentially gives better accuracy, so $UA_{\mathcal{I},:}$ is used for HAN-A.

The following stopping criteria may be used in the iterations.

- The iterations stop when a maximum sample size or a target numerical rank is reached. The numerical rank is reflected by $|\mathcal{I}|$ or $|\mathcal{J}|$, depending on the output low-rank form in (3.21).
- In HAN-B and HAN-A, the iteration stops when $\mathcal{I}$ stays the same as in the previous step.
- Another criterion is when the approximation error is smaller than τ. It is generally expensive to directly evaluate the error. There are various ways to estimate it. For example, in HAN-U and HAN-A, we may use the following bound based on (3.5) and (3.10):

$$\|A-\tilde{A}\|_2 = \|S\|_2 \approx \left\| \begin{pmatrix} I \\ \hat{E} \end{pmatrix} \begin{pmatrix} S_{22} & S_{23} \end{pmatrix} \right\|_2 \le \left\| \begin{pmatrix} I \\ \hat{E} \end{pmatrix} \right\|_2 \left\| \begin{pmatrix} S_{22} & S_{23} \end{pmatrix} \right\|_2.$$

(Note the approximations to A and S are obtained by randomization.) We may also directly estimate the absolute or relative approximation errors without the need to evaluate $\|A-\tilde{A}\|_2$ or $\|A\|_2$, as explained below.

The following lemmas suggest how to estimate the absolute and relative errors.

LEMMA 3.5. *Suppose $k = |\mathcal{J}|$ for the column index set $\mathcal{J}$ of A_{11} in (3.4) and $\widehat{\mathcal{J}}$ is a set of independent uniform random samples from $\{1:n\}\backslash\mathcal{J}$ via the index set $\mathcal{L}$ as in (3.7) with $|\mathcal{L}| = b$. Let $\theta = \frac{n-k}{b}\|S_{:,\mathcal{L}}\|_F^2$ and $\mathcal{E} = A - UA_{\mathcal{I},:}$. Then*

$$\mathrm{E}(\theta) = \|\mathcal{E}\|_F^2. \tag{3.22}$$

If (3.5) is further assumed to satisfy (3.6), then

$$\Pr\left(\left|\theta - \|\mathcal{E}\|_F^2\right| \geq ((n-k)f^2)(\sigma_{k+1}(A))^2\right) \leq 2\exp\left(-2b\right).$$

Proof. From (3.5), $\|\mathcal{E}\|_F = \|S\|_F$. S has size $(m-k)\times(n-k)$. $S_{:,\mathcal{L}}$ essentially results from the uniform sampling of the columns of S with $\mathcal{L}$ in (3.7). Let C be the submatrix formed by the b columns of the order-$(n-k)$ identity matrix corresponding to the column index set $\mathcal{L}$. Then

$$\begin{aligned} S_{:,\mathcal{L}} &= SC, \\ \mathrm{E}(\|S_{:,\mathcal{L}}\|_F^2) &= \mathrm{E}(\mathrm{trace}(S_{:,\mathcal{L}}^T S_{:,\mathcal{L}})) = \mathrm{E}(\mathrm{trace}(C^T S^T SC)) \\ &= \frac{b}{n-k}\,\mathrm{trace}(S^T S) = \frac{b}{n-k}\|S\|_F^2, \end{aligned} \tag{3.23}$$

where the equality from the first line to the second directly comes by the definition of expectations and is a trace estimation result in [1]. This gives (3.22).

If (3.5) is further assumed to satisfy (3.6), then $\|S_{:,j}\|_2 \leq \|S\|_2 \leq f\sigma_{k+1}(A)$. The probability result can be obtained like in [1, Theorem 8.2] by writing $\|S_{:,\mathcal{L}}\|_F^2$ as the sum of b squares $\|S_{:,j}\|_2^2$ and applying Hoeffding's inequality with $\varepsilon > 0$:

$$\begin{aligned} \Pr\left(\left|\theta - \|\mathcal{E}\|_F^2\right| \geq \varepsilon\right) &\leq 2\exp\left(-\frac{2b\varepsilon^2}{(n-k)^2\max_{j\in\mathcal{L}}\|S_{:,j}\|_2^4}\right) \\ &\leq 2\exp\left(-\frac{2b\varepsilon^2}{(n-k)^2\left(f\sigma_{k+1}(A)\right)^4}\right). \end{aligned}$$

Setting $\varepsilon = (n-k)\left(f\sigma_{k+1}(A)\right)^2$ to get the result. ∎

The probability result indicates that, even with small b, θ is a very accurate estimator for $\|\mathcal{E}\|_F^2$ (provided that (3.6) holds). We can further consider the estimation of the relative error.

LEMMA 3.6. *With the assumptions and notation in Lemma 3.5, $H = \frac{n-k}{b}S_{:,\mathcal{L}}S_{:,\mathcal{L}}^T$ satisfies*

$$\frac{\|\mathcal{E}\|_2}{\|A\|_2} \leq \frac{\sqrt{\|\,\mathrm{E}(H)\|_2}}{\|A_{11}\|_2} \leq f^2\frac{\sigma_{k+1}(A)}{\|A\|_2}. \tag{3.24}$$

Proof. With (3.23),

$$\mathrm{E}(S_{:,\mathcal{L}}S_{:,\mathcal{L}}^T) = \mathrm{E}(SCC^TS^T) = S[\mathrm{E}(CC^T)]S^T = \frac{b}{n-k}SS^T,$$

where $\mathrm{E}(CC^T) = \frac{b}{n-k}I$ is simply by the definition of expectations and has been explored in, say, [9]. This leads to

$$\sqrt{\|\,\mathrm{E}(H)\|_2} = \|S\|_2 = \|\mathcal{E}\|_2,$$

which, together with $\|A_{11}\|_2 \le \|A\|_2$, yields the first inequality in (3.24). The second inequality in (3.24) is based on (3.6):

$$\frac{\sqrt{\|\operatorname{E}(H)\|_2}}{\|A_{11}\|_2} = \frac{\|S\|_2}{\|A_{11}\|_2} \le \frac{f\sigma_{k+1}(A)}{\|A\|_2/f} = f^2\frac{\sigma_{k+1}(A)}{\|A\|_2}.$$ ∎

From these lemmas, we can see that the absolute or relative errors in the low-rank approximation may be estimated by using $S_{:,\mathcal{L}}$ and A_{11}. For example, a reasonable estimator for the relative error of the low-rank approximation $\tilde{A}$ is given by

$$\phi = \sqrt{\frac{n-k}{b}}\frac{\|S_{:,\mathcal{L}}\|_2}{\|A_{11}\|_2}(\approx \frac{\|A-\tilde{A}\|_2}{\|A\|_2}). \tag{3.25}$$

This estimator can be quickly evaluated and only costs $O(b(m-k)+b^2+k^2)$. The cost may be further reduced to $O(b^2+k^2)$ by using $\sqrt{\frac{n-k}{b}}\frac{\|S_{\mathcal{K},\mathcal{L}}\|_2}{\|A_{11}\|_2}$ since $S_{\mathcal{K},\mathcal{L}}$ results from a strong rank-revealing factorization applied to $S_{:,\mathcal{L}}$ and there is a low-degree polynomial g in $m-k$ and b such that $\frac{\|S_{:,\mathcal{L}}\|_2}{g} \le \|S_{\mathcal{K},\mathcal{L}}\|_2 \le \|S_{:,\mathcal{L}}\|_2$. To enhance the reliability, we may stop the iteration if the estimators return errors smaller than a threshold consecutively for multiple steps.

4. Numerical tests. We now illustrate the performance of the HAN schemes and compare with some other Nyström-based schemes. The following methods will be tested:

- HAN-B, HAN-U, HAN-A: the HAN schemes as in Algorithms 3.1 and 3.3;
- Nys-B: the traditional Nyström method to produce an approximation like in (1.1), where both the row index set $\mathcal{I}$ and the column index set $\mathcal{J}$ are uniformly and randomly selected;
- Nys-P: the scheme to find an approximation like in (2.6) but with $\mathcal{I}$ obtained by one pivoting step (2.3) applied to uniformly and randomly selected $A_{:,\mathcal{J}}$;
- Nys-R: the scheme that extends Nys-P by applying several steps of alternating direction refinements to improve $\mathcal{I}$ and $\mathcal{J}$ like in lines 7–9 of Algorithm 3.1, which corresponds to the iterative cross-approximation scheme in [25]. (In Nys-R, the accuracy typically stops improving after few steps of refinement, so we fix the number of refinement steps to be 10 in the tests.)

In the HAN schemes HAN-B, HAN-U, and HAN-A, the stepsize b in the progressive column sampling is set to be $b = 5$. The stopping criteria follow the discussions at the beginning of Section 3.3. Specifically, the iteration stops if the randomized relative error estimate in (3.25) is smaller than the threshold $\tau = 10^{-14}$, or if the total sample size S (in all progressive sampling steps) reaches a certain maximum, or if the index refinement no longer updates the row index set $\mathcal{I}$. Since the HAN schemes involve randomized error estimation, it is possible for some iterations to stop earlier or later than necessary. Also, HAN-B does not use the fast subset update strategy in Section 3.2, so an extra step is added to estimate the accuracy with (3.25).

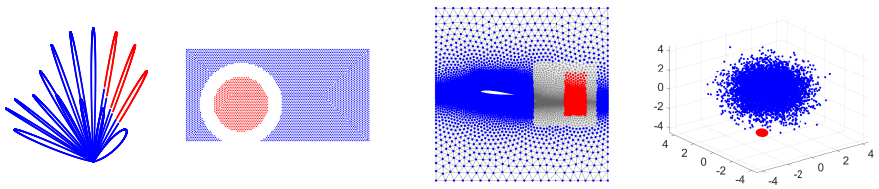
The Nyström-based schemes Nys-B, Nys-P, and Nys-R are directly applied with different given sample sizes S and do not really have a fast accuracy estimation mechanism. In the plots below for the relative approximation errors $\frac{\|A-\tilde{A}\|_2}{\|A\|_2}$, the Nyström and HAN schemes are put together for comparison. However, it is important to distinguish the meanings of the sample sizes S for the two cases along the horizontal axes. For the Nyström schemes, each S is set directly. For the HAN schemes, each S is the total sample size of all sampling steps and is reached progressively through a sequence of steps each of stepsize b.

In the three Nyström schemes, the cardinality $|\mathcal{I}|$ will be reported as the numerical rank. In the HAN schemes, the numerical rank will be either $|\mathcal{I}|$ or $|\mathcal{J}|$, depending on the low-rank form in (3.21).

Since the main applications of the HAN schemes are numerical computations, our tests below focus on two and three dimensional problems, including some discretized meshes and some structured matrix problems. We also include an example related to high-dimensional data sets. The tests are done in Matlab R2019a on a cluster using two 2.60GHz cores and 32GB of memory.

EXAMPLE 1. First consider some kernel matrices generated by the evaluation of various commonly encountered kernel functions evaluated at two well-separated data points $\mathbf{x}$ and $\mathbf{y}$ in two and three dimensions. $\mathbf{x}$ and $\mathbf{y}$ are taken from the following four data sets (see Figure 4.1).

(a) `Flower`: a flower shape curve, where the $\mathbf{x}$ set is located at a corner and $|\mathbf{x}| = 1018$, $|\mathbf{y}| = 13965$.
(b) `FEM`: a 2D finite element mesh extracted from the package MESHPART [12], where the $\mathbf{x}$ set is surrounded by the points in $\mathbf{y}$ with $|\mathbf{x}| = 821$, $|\mathbf{y}| = 4125$. The mesh is from an example in [47] that shows the usual Nyström method fails to reach high accuracies for some kernel matrices even with the number of samples near the numerical rank.
(c) `Airfoil`: an unstructured 2D mesh (airfoil) from the SuiteSparse matrix collection (http://sparse.tamu.edu), where the $\mathbf{x}$ and $\mathbf{y}$ sets are extracted so that $\mathbf{x}$ has a roughly rectangular shape and $|\mathbf{x}| = 617$, $|\mathbf{y}| = 11078$.
(d) `Set3D`: A set of 3D data points extract from the package DistMesh [32] but with the $\mathbf{y}$ points randomly perturbed with $|\mathbf{x}| = 717$, $|\mathbf{y}| = 6650$.



(a) `Flower` (b) `FEM` (c) `Airfoil` (d) `Set3D`

FIG. 4.1. *Example 1. Data sets under consideration with the* $\mathbf{x}$ *and* $\mathbf{y}$ *sets marked in red and blue, respectively.*

The points in the data sets are nonuniformly distributed in general, except in the case `FEM` where the points are more uniform. The data points in two dimensions are treated as complex numbers. The setup of the $\mathbf{x}$ and $\mathbf{y}$ sets has the size of $\mathbf{x}$ just several times larger than the target numerical rank. This is often the case in the FMM and structured solvers where the corresponding matrix blocks are short and wide off-diagonal blocks that need to be compressed in the hierarchical approximation of a global kernel matrix (see, e.g., [15, 33, 38, 41]). We consider several types of kernels as follows:

$$\kappa(x,y) = \frac{1}{x-y}, \quad \frac{1}{(x-y)^2}, \quad \frac{1}{|x-y|}, \quad \sqrt{|x-y|+1}, \quad \frac{1}{\sqrt{|x-y|^2+1}},$$
$$e^{-|x-y|}, \quad e^{-\alpha|x-y|^2}, \quad \log|x-y|, \quad \tan(x\cdot y+1),$$

where α is a parameter. Such kernels are frequently used in the FMM and in struc-

tured matrix computations like Toeplitz solutions [7] and some structured eigenvalue solvers [16, 30, 36]. For data points in three dimensions, $|x-y|$ represents the distance between x and y.

For each data set, we apply the methods above to the kernel matrices A as in (2.1) formed by evaluating some $\kappa(x,y)$ at $\mathbf{x}$ and $\mathbf{y}$. Most of the kernel matrices have modest numerical ranks. The schemes Nys-B, Nys-P, and Nys-R use sample sizes S up to 400 in almost all the tests. The HAN schemes use much smaller sample sizes. HAN-B and HAN-U use sample sizes $S \le 200$ for most tests, and HAN-A uses sample sizes $S \le 50$ for all the cases.

For some kernels evaluated at the set `Flower`, the relative errors $\frac{\|A-\tilde{A}\|_2}{\|A\|_2}$ in one test run are reported in Figure 4.2. With larger S, the error typically gets smaller. However, Nys-B is only able to reach modest accuracies even if S is quite large. (The error curve nearly stagnate in the first row of Figure 4.2 with increasing S.) The accuracy gets better with Nys-P for some cases. Nys-R can further improve the accuracy. However, they still cannot get accuracy close to $\tau = 10^{-14}$ and their error curves in the second row of Figure 4.2 get stuck around some small rank sizes insufficient to reach high accuracies.

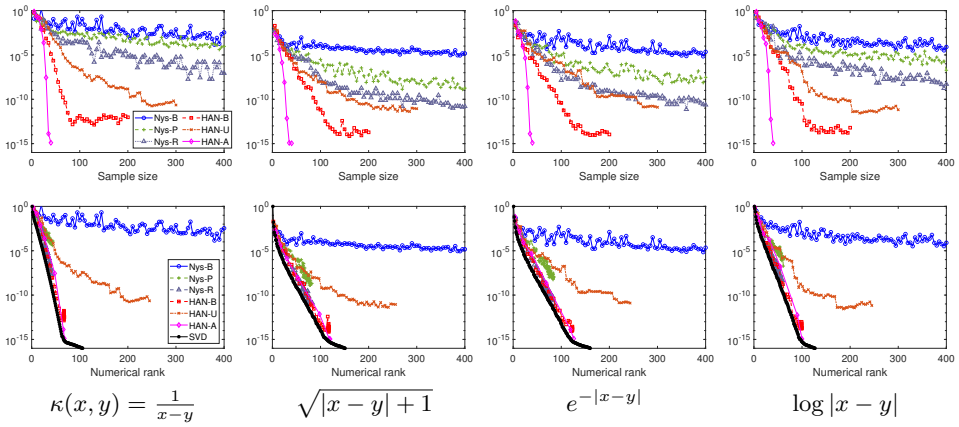


FIG. 4.2. *Example 1 (data set `Flower`): Low-rank approximation errors* $\frac{\|A-\tilde{A}\|_2}{\|A\|_2}$ *as the sample size S increases in one test, where the second row shows the errors with respect to the resulting numerical ranks corresponding to the first row and the SVD line shows the scaled singular values.*

In comparison, the HAN schemes usually yield much better accuracies, especially with HAN-B and HAN-A. HAN-U is often less accurate than HAN-B but is more efficient because of the fast subset update. The most remarkable result is from HAN-A, which quickly reaches accuracies around 10^{-15} after few sampling steps (with small overall sample sizes). The second row of Figure 4.2 also includes the scaled singular values $\frac{\sigma_i(A)}{\sigma_1(A)}$. We can observe that HAN-B and particularly HAN-A produce approximation errors with decay patterns very close to that of SVD.

To further confirm the accuracies, we run each scheme 100 times and report the results in Figure 4.3. In general, we observe that the HAN schemes are more accurate, especially HAN-B and HAN-A. The direct outcome from HAN-U is not as accurate, but this is likely due to the quality of the basis matrices in (3.21). Originally in HAN-U, the columns are collected from previous iteration steps to assemble $A_{:,\mathcal{J}}$. It is possible the quality of some earlier columns is not extremely high. On the other hand, $A_{:,\mathcal{J}}$ can

work sufficiently well as a candidate for row pivoting to find representative rows and a new approximation $UA_{\mathcal{I},:}$ like in (3.21) where $A_{\mathcal{I},\mathcal{J}}$ has the quality guarantee from the SRR factorization. Accordingly, the representative rows $A_{\mathcal{I},:}$ are also expected to have high quality. (In fact, HAN-B and HAN-A end the iteration with one row or column pivoting step by an SRR factorization.) With this additional row pivoting step, the resulting errors of HAN-U (called *effective* errors in Figure 4.3) are observed to be close to those of HAN-B.

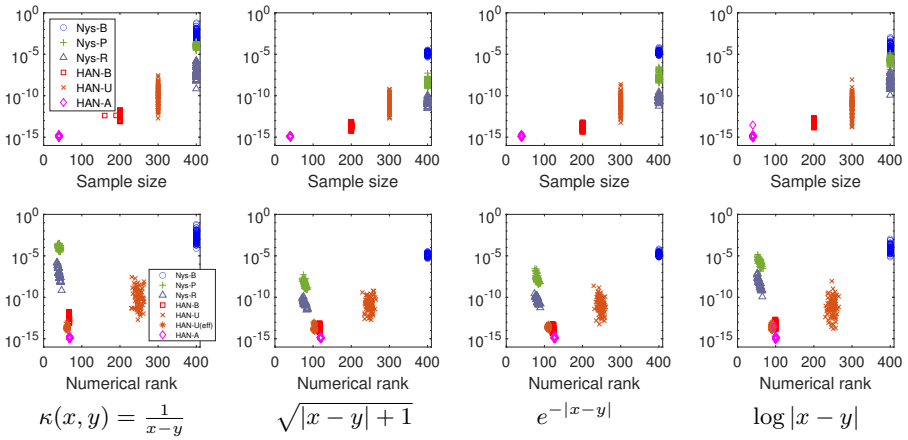


FIG. 4.3. *Example 1 (data set* `Flower`*): Relative errors from running the methods for* 100 *times, where HAN-U(eff) is for the effective errors of HAN-U.*

Similarly, for the other data sets and various different kernel functions, we have test results as given in Figures 4.4–4.7. (Sample convergence curves for the sets `FEM` and `Airfoil` are not included since they are similar to those in Figures 4.2 and 4.6.) The results can be interpreted similarly. For some cases, Nys-B, Nys-P, and even Nys-R may be quite inaccurate. One example is for $\kappa(x,y)=e^{-16|x-y|^2}$ in Figures 4.6 and 4.7, where even Nys-R becomes quite unreliable and demonstrates oscillatory errors for different S and different tests.

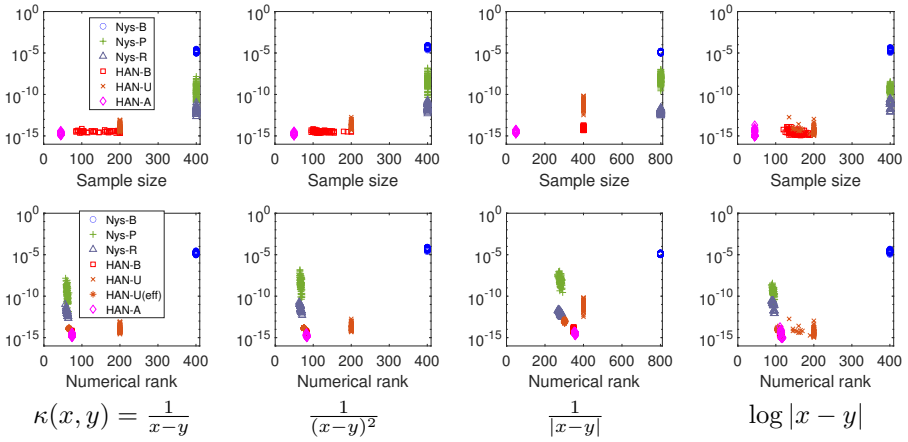


FIG. 4.4. *Example 1 (data set* `FEM`*): Relative errors from running the methods for* 100 *times, where HAN-U(eff) is for the effective errors of HAN-U.*

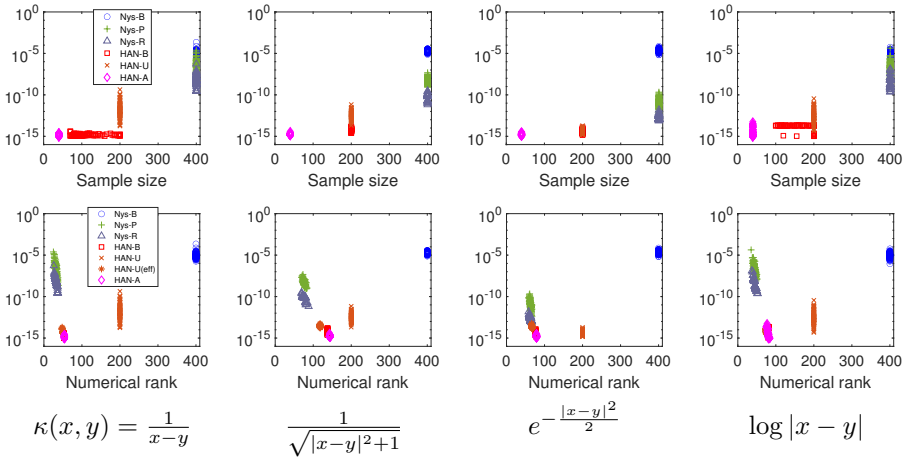


FIG. 4.5. *Example 1 (data set* `Airfoil`*): Relative errors from running the methods for* 100 *times, where* HAN-U(eff) *is for the effective errors of* HAN-U.

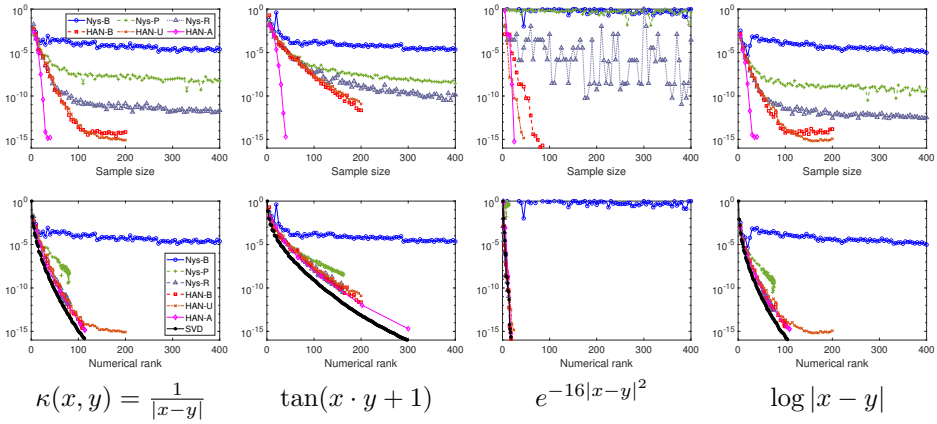


FIG. 4.6. *Example 1 (data set* `Set3D`*): Low-rank approximation errors* $\frac{\|A-\tilde{A}\|_2}{\|A\|_2}$ *as the sample size* S *increases in one test, where the second row shows the errors with respect to the resulting numerical ranks corresponding to the first row and the SVD line shows the scaled singular values.*

For each data set, the average timing of the methods from 100 runs is shown in Table 4.1. Not surprisingly, the simple inaccurate method Nys-B is very fast. We note that the HAN schemes include adaptive error control steps that are not in the Nyström schemes, which gives the Nyström schemes advantages in the timing comparison. Nevertheless, when we compare the most accurate algorithms Nys-R and HAN-A from the two classes, the aggressive rank advancement makes HAN-A very efficient. HAN-A is generally faster than Nys-R by multiple times.

EXAMPLE 2. Next, we consider two types of matrices related to some applications.

(a) The first one is a type of implicitly defined kernel matrices. Suppose C is a circulant matrix with eigenvalues being discretized values of a function $f(t)$ at some points in an interval. For example, $f(t)$ may be a coefficient function in a differential equation. Such matrices frequently appear in discretizations of ODEs and PDEs [2, 3], spectral methods [33], as well as image processing

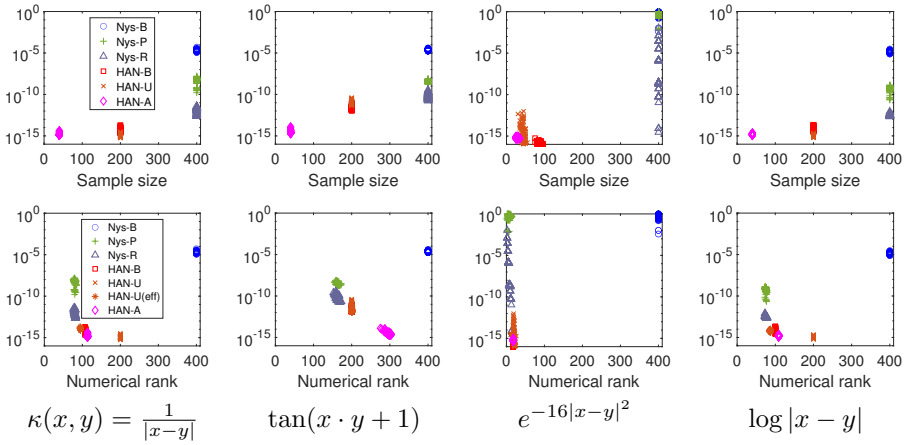


FIG. 4.7. *Example 1 (data set* `Set3D`*): Relative errors from running the methods for* 100 *times, where* HAN-U(eff) *is for the effective errors of* HAN-U.

TABLE 4.1
Example 1. Average timing (in seconds) of the methods from 100 *runs.*

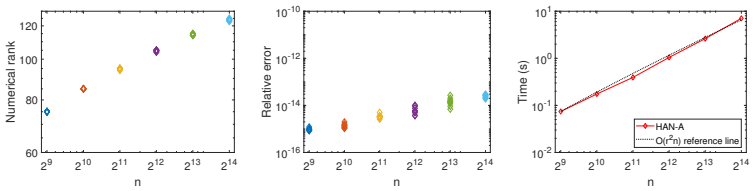
	$\kappa(x,y)$	$\frac{1}{x-y}$	$\sqrt{\lvert x-y\rvert+1}$	$e^{-\lvert x-y\rvert}$	$\log\lvert x-y\rvert$
`Flower`	Nys-B	0.056	0.020	0.019	0.019
	Nys-P	0.150	0.047	0.046	0.046
	Nys-R	**1.751**	**1.285**	**1.313**	**0.851**
	HAN-B	10.947	6.332	6.237	5.281
	HAN-U	4.786	2.062	1.987	2.061
	HAN-A	**0.802**	**0.727**	**0.690**	**0.512**
	$\kappa(x,y)$	$\frac{1}{x-y}$	$\frac{1}{(x-y)^2}$	$\frac{1}{\lvert x-y\rvert}$	$\log\lvert x-y\rvert$
`FEM`	Nys-B	0.059	0.058	0.186	0.019
	Nys-P	0.114	0.114	0.208	0.037
	Nys-R	**0.868**	**0.962**	**2.495**	**0.469**
	HAN-B	2.738	1.705	13.454	1.057
	HAN-U	0.702	0.702	1.334	0.342
	HAN-A	**0.191**	**0.193**	**0.607**	**0.108**
	$\kappa(x,y)$	$\frac{1}{x-y}$	$\frac{1}{\sqrt{\lvert x-y\rvert^2+1}}$	$e^{-\frac{\lvert x-y\rvert^2}{2}}$	$\log\lvert x-y\rvert$
`Airfoil`	Nys-B	0.071	0.020	0.020	0.018
	Nys-P	0.119	0.033	0.032	0.029
	Nys-R	**1.155**	**0.937**	**0.734**	**0.457**
	HAN-B	3.486	5.813	3.361	2.468
	HAN-U	1.944	0.946	0.940	0.898
	HAN-A	**0.322**	**0.451**	**0.289**	**0.158**
	$\kappa(x,y)$	$\frac{1}{\lvert x-y\rvert}$	$\tan(x\cdot y+1)$	$e^{-16\lvert x-y\rvert^2}$	$\log\lvert x-y\rvert$
`Set3D`	Nys-B	0.023	0.024	0.030	0.023
	Nys-P	0.042	0.046	0.082	0.042
	Nys-R	**0.723**	**1.849**	**0.121**	**0.680**
	HAN-B	3.299	4.732	0.139	3.095
	HAN-U	0.645	0.630	0.035	0.651
	HAN-A	**0.355**	**0.756**	**0.035**	**0.333**

[29]. They are usually multiplied or added to some other matrices so that the circulant structure is destroyed. However, it is shown in [33, 44] that they have small off-diagonal numerical ranks with some $f(t)$. Such rank structures are preserved under various matrix operations and can then be used to design fast direct solvers. The matrix A we consider here is the $n \times n$ upper right off-diagonal block of each C (with half of the size of C) with $f(t)$ a piecewise linear function with discontinuity. It is shown in [44] that A is the evaluation of an implicit kernel function over certain data points.

(b) The second type of matrices results from a 2D discretized linear elasticity equation as in [42, 43]. Under some conditions, the direct factorization of the sparse discretized matrix yields Schur complements S with small off-diagonal numerical ranks. Fast structured sparse direct solvers like those in [38, 39] reply on the low-rank approximations of the off-diagonal blocks. Here, each S is the final Schur complement in the factorization based on nested dissection ordering of the sparse matrix. We take the matrix A to be the $n \times n$ upper right off-diagonal block of each S. The size n increases with the discretization mesh size. S is not the direct evaluation of a kernel at given data points but can be related to a certain kernel function.

We consider A with varying sizes so as to demonstrate that HAN-A can reach high accuracies with nearly linear complexity. For each n, we run HAN-A for 10 times and report the outcome. For the first type of matrices, Figure 4.8(a) shows the numerical ranks r from HAN-A, which slowly increase with n. This is consistent with the result in [44] where it is shown that the numerical ranks grow as a low-degree power of $\log n$. The low-rank approximation errors are given in Figure 4.8(b) and the average time from the 10 runs for each n is given in Figure 4.8(c). When n doubles, the runtimes roughly follow the $O(r^2 n)$ pattern, as explained in Section 3.2.

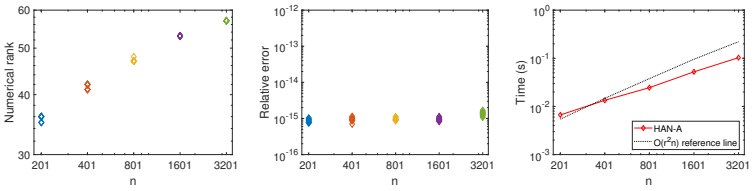
For the second type of matrices, the results are shown in Figure 4.9. We can similarly obtain high accuracies with nearly linear complexity.



(a) Numerical rank r (b) $\frac{\|A-\tilde{A}\|_2}{\|A\|_2}$ (c) Average timing

FIG. 4.8. *Example 2: Results from HAN-A for a type of implicitly defined kernel matrices arising in spectral methods.*

EXAMPLE 3. Finally for completeness, we would like to show that the HAN schemes also work for high-dimensional data sets. (We remark that practical data analysis may not necessarily need very high accuracies. However, the HAN schemes can serve as a fast way to convert such data matrices into some rank structured forms that allow quick matrix operations.) We consider kernel matrices resulting from the evaluation of some kernel functions at two data sets `Abalone` and `DryBean` from the UCI Machine Learning Repository (https://archive.ics.uci.edu). The two data sets have 4177 and 13611 points in 8 and 16 dimensions, respectively. Here, each data set is standardized to have mean 0 and variance 1. We take the submatrix of each



(a) Numerical rank r (b) $\frac{\|A-\tilde{A}\|_2}{\|A\|_2}$ (c) Average timing

FIG. 4.9. *Example 2: Results from* HAN-A *for dense off-diagonal blocks of Schur complements within factorizations of a 2D discretized elasticity equation discretized on different mesh sizes.*

resulting kernel matrix formed by the first 1000 rows so as to make it rectangular and nonsymmetric.

A set of test results is given in Figure 4.10. Nys-B can only reach modest accuracies around 10^{-5}. Nys-R can indeed gets quite good accuracies. Nevertheless, HAN-A still reaches high accuracies with a small number of sampling steps. Similar results are observed with multiple runs. The timing comparison is also similar to that in Example 1. For example, for the test matrix with $\kappa(x,y) = e^{-\frac{|x-y|^2}{\sigma^2}}$ and the data set `Abalone`, the average times of 100 runs of Nys-B, Nys-P, Nys-R, and HAN-A used in this example are 0.018, 0.043, 0.914, and 0.247 seconds, respectively.

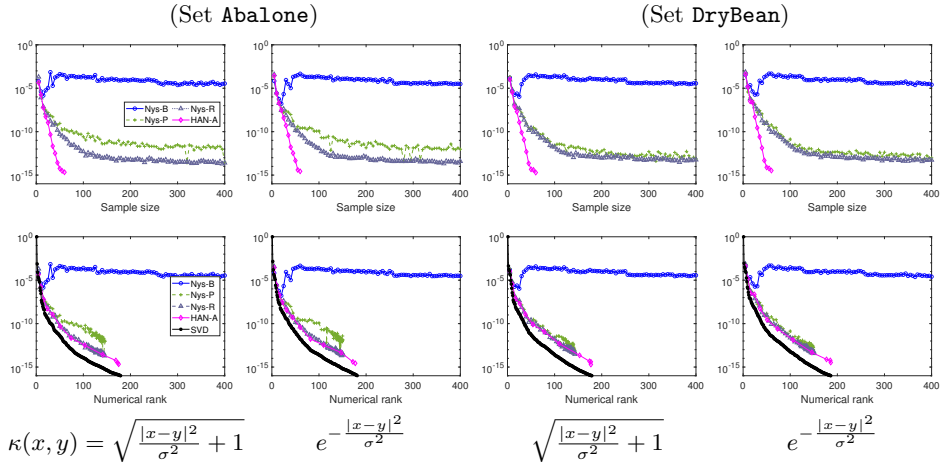


FIG. 4.10. *Example 3: Low-rank approximation errors* $\frac{\|A-\tilde{A}\|_2}{\|A\|_2}$ *for high-dimensional tests, where* σ *in the kernel functions is set to be four times the maximum distance between the data points and the origin, and the SVD line shows the scaled singular values.*

5. Conclusions. This work proposes a set of techniques that can make the Nyström method reach high accuracies in practice for kernel matrix low-rank approximations. The usual Nyström method is combined with strong rank-revealing factorizations to serve as a pivoting strategy. The low-rank basis matrices are refined through alternating direction row and column pivoting. This is incorporated into a progressive sampling scheme until a desired accuracy or numerical rank is reached. A fast subset update strategy further leads to improved efficiency and also convenient randomized accuracy control. The design of the resulting HAN schemes is based on

some strong heuristics, as supported by some relevant accuracy and singular value analysis. Extensive numerical tests show that the schemes can quickly reach high accuracies, sometimes with quality close to SVDs.

The schemes are useful for low-rank approximations related to kernel matrices in many numerical computations. They can also be used in rank-structured methods to accelerate various data analysis tasks. The design of the schemes is fully algebraic and does not require particular information from the kernel or the data sets. It remains open to give statistical or deterministic analysis of the decay of the approximation error in the progressive sampling and refinement steps. We are also attempting a probabilistic study of some steps in the HAN schemes that may be viewed as a randomized rank-revealing factorization.

Acknowledgement. The author would like to thank the two anonymous referees for their valuable suggestions.

REFERENCES

[1] H. Avron and S. Toledo, *Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix*, J. ACM 58 (2011), Article 8.
[2] Z.-Z. Bai and M. K. Ng, *Preconditioners for nonsymmetric block toeplitz-like-plus-diagonal linear systems*, Numer. Math., 96 (2003), pp. 197–220.
[3] Z.-Z. Bai and K.-Y. Lu, *On regularized Hermitian splitting iteration methods for solving discretized almost-isotropic spatial fractional diffusion equations*, 27, (2020), e2274.
[4] M. Bebendorf, *Approximation of boundary element matrices*, Numer. Math., 86 (2000), pp. 565–589.
[5] D. Cai, J. Nagy, and Y. Xi, *Fast deterministic approximation of symmetric indefinite kernel matrices with high dimensional datasets*, SIAM J. Matrix Anal. Appl., 43 (2022), pp. 1003–1028.
[6] T. F. Chan, *Rank revealing QR factorizations*, Linear Algebra Appl., 88/89 (1987), pp. 67–82.
[7] S. Chandrasekaran, M. Gu, X. Sun, J. Xia, and J. Zhu, *A superfast algorithm for Toeplitz systems of linear equations*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 1247–1266.
[8] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang, *Matrix approximation and projective clustering via volume sampling*, Theory Comput., 2 (2006), pp. 225–247.
[9] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication*, SIAM Journal on Computing 36 (2006), pp. 132–157.
[10] P. Drineas and M. W. Mahoney, *On the Nyström method for approximating a Gram matrix for improved kernel-based learning*, J. Machine Learning, 6 (2005), pp. 2153–2175.
[11] P. Drineas, M. W. Mahoney, and D. P. Woodruff, *Fast approximation of matrix coherence and statistical leverage*, J. Machine Learning, 13 (2012) pp. 3441–3472.
[12] J. R. Gilbert and S.-H. Teng, *MESHPART, A Matlab Mesh Partitioning and Graph Separator Toolbox*, http://aton.cerfacs.fr/algor/Softs/MESHPART/.
[13] A. Gittens and M. W. Mahoney, *Revisiting the Nyström method for improved large-scale machine learning*, J. Machine Learning, 16 (2016), pp. 1–65.
[14] S. A. Goreinov and E. E. Tyrtyshnikov, *The maximal-volume concept in approximation by low-rank matrices*, in Contemporary Mathematics, vol 280, 2001, pp. 47–52.
[15] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.
[16] M. Gu and S. C. Eisenstat, *A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 79–92.
[17] M. Gu and S. C. Eisenstat, *Efficient algorithms for computing a strong rank-revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.
[18] W. Hackbusch, *A sparse matrix arithmetic based on $\mathcal{H}$-matrices*, Computing, 62 (1999), pp. 89–108.
[19] N. Halko, P.G. Martinsson, and J. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), pp. 217–288.
[20] K. Hamm and L. Huang, *Perspectives on CUR decompositions*, Appl. Comput. Harmon. Anal., 48 (2020), pp. 1088–1099.
[21] K. Hamm and L. Huang, *Perturbations of CUR decompositions*, SIAM J. Matrix Anal. Appl.

42 (2021), pp. 351–375.
[22] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, 1991.
[23] I. C. F. Ipsen and T. Wentworth, *Sensitivity of leverage scores and coherence for randomized matrix algorithms*, Extended abstract, Workshop on Advances in Matrix Functions and Matrix Equations, Manchester, UK, 2013.
[24] S. Kumar, M. Mohri, and A. Talwalkar, *Sampling methods for the Nyström method*, J. Machine Learning, 13 (2012), pp. 981–1006.
[25] Q. Luan and V. Y. Pan, *CUR LRA at sublinear cost based on volume maximization*, In LNCS 11989, Book: Mathematical Aspects of Computer and Information Sciences (MACIS 2019), D. Salmanig et al (Eds.), Springer Nature Switzerland AG 2020, Chapter No: 10, pp. 1–17.
[26] T. Mach, L. Reichel, M. Van Barel, and R. Vandebril, *Adaptive cross approximation for ill-posed problems*, J. Comput. Appl. Math., 303 (2016), pp. 206–217.
[27] P. G. Martinsson, G. Quintana-Orti, N. Heavner, and R. van de Geijn, *Householder QR factorization with randomization for column pivoting (HQRRP)*, SIAM J. Sci. Comput., 39 (2017), pp. C96–C115.
[28] L. Miranian and M Gu, *Strong rank revealing LU factorizations*, Linear Alg. Appl., 367 (2003), pp. 1–16.
[29] J. Nagy, P. Pauca, R. Plemmons, and T. Torgersen, *Space-varying restoration of optical images*, J. Opt. Soc. Amer. A, 14 (1997), pp. 3162–3174.
[30] X. Ou and J. Xia, *SuperDC: Superfast divide-and-conquer eigenvalue decomposition with improved stability for rank-structured matrices*, SIAM J. Sci. Comput., 44 (2022), pp. A3041–A3066.
[31] V. Y. Pan, Q. Luan, J. Svadlenka, and L. Zhao, *CUR low rank approximation of a matrix at sublinear cost*, arXiv:1906.04112.
[32] P.-O. Persson, *DistMesh - A Simple Mesh Generator in MATLAB*, http://persson.berkeley .edu/distmesh.
[33] J. Shen, Y. Wang, and J. Xia, *Fast structured direct spectral methods for differential equations with variable coefficients, I. The one-dimensional case*, SIAM J. Sci. Comput., 38 (2016), pp. A28–A54.
[34] A. Talwalkar and A. Rostamizadeh, *Matrix coherence and the Nyström method*, UAI'10: Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, (2010), pp. 572–579.
[35] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, *Practical sketching algorithms for low-rank matrix approximation*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 1454–1485.
[36] J. Vogel, J. Xia, S. Cauley, and V. Balakrishnan, *Superfast divide-and-conquer method and perturbation analysis for structured eigenvalue solutions*, SIAM J. Sci. Comput., 38 (2016), pp. A1358–A1382.
[37] C. Williams and M. Seeger, *Using the Nyström method to speed up kernel machines*, Advances in Neural Information Processing Systems 13, (2001), pp. 682–688.
[38] J. Xia, *Randomized sparse direct solvers*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 197–227.
[39] J. Xia, *Efficient structured multifrontal factorization for general large sparse matrices*, SIAM J. Sci. Comput., 35 (2013), pp. A832–A860.
[40] J. Xia, *Multi-layer hierarchical structures*, CSIAM Trans. Appl. Math., 2 (2021), pp. 263–296.
[41] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Fast algorithms for hierarchically semiseparable matrices*, Numer. Linear Algebra Appl., 17 (2010), pp. 953–976.
[42] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Superfast multifrontal method for large structured linear systems of equations*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1382–1411.
[43] J. Xia and M. Gu, *Robust approximate Cholesky factorization of rank-structured symmetric positive definite matrices*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2899–2920.
[44] J. Xia and M. Lepilov, *Why are many circulant matrices rank structured?* Preprint.
[45] J. Xia, Y. Xi, and M. Gu, *A superfast structured solver for Toeplitz linear systems via randomized sampling*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 837–858.
[46] J. Xiao, M. Gu, and J. Langou, *Fast parallel randomized QR with column pivoting algorithms for reliable low-rank matrix approximations*, 24th IEEE International Conference on High Performance Computing, Data, and Analytics (HIPC), Jaipur, India, 2017.
[47] X. Ye, J. Xia, and L. Ying, *Analytical low-rank compression via proxy point selection*, SIAM J. Matrix Anal. Appl., 41 (2020), pp. 1059–1085.
[48] K. Zhang, I. W. Tsang, and J. T. Kwok, *Improved Nyström low-rank approximation and error analysis*, Proceedings of the 25th international conference on Machine learning, (2008), pp. 1232–1239.