Randomly pivoted Cholesky:
Practical approximation of a kernel matrix with few entry evaluations[*]

Yifan Chen[†], Ethan N. Epperly[†], Joel A. Tropp[†], and Robert J. Webber[†]

Abstract. Randomly pivoted Cholesky (RPCHOLESKY) is a natural algorithm for computing a rank-k approximation of an $N \times N$ positive semidefinite (psd) matrix. RPCHOLESKY can be implemented with just a few lines of code; it requires only $(k+1)N$ entry evaluations and $\mathcal{O}(k^2 N)$ additional arithmetic operations. This paper offers the first serious investigation of its experimental and theoretical behavior. Empirically, RPCHOLESKY matches or improves on the performance of alternative algorithms for low-rank psd approximation. Furthermore, RPCHOLESKY provably achieves near-optimal approximation guarantees. The simplicity, effectiveness, and robustness of this algorithm strongly support its use in scientific computing and machine learning applications.

Key words. kernel method, low-rank approximation, Nyström approximation

MSC codes. 65F55, 65C99, 68T05

1. Introduction. Kernel methods [27] are a class of machine learning algorithms that are often used for prediction and clustering tasks. Kernel methods require manipulating a positive semidefinite (psd) *kernel matrix* $A \in \mathbb{R}^{N \times N}$. The entries $a_{ij} = K(x^{(i)}, x^{(j)})$ of the kernel matrix are determined by a positive definite *kernel function* $K(\cdot, \cdot)$ that measures the similarity between pairs of data points [27, Def. 2.5]. The dimension N of the kernel matrix is equal to the number of data points, which can easily grow into the millions or higher. The size of modern data sets poses a challenge for kernel methods: even generating all N^2 entries of A may be infeasible when $N \geq 10^5$.

To speed up kernel computations, a standard practice is to replace the kernel matrix A with a low-rank approximation. Remarkably, in the common setting in which the $N \times N$ kernel matrix has rapidly decaying eigenvalues, it is possible to generate an accurate approximation using only $\mathcal{O}(N)$ entry evaluations and $\mathcal{O}(N)$ arithmetic operations. Nevertheless, it has been formidable to design a practical and provably justified algorithm that achieves this goal.

The *column Nyström approximation* [19, §19.2] is the most widely used technique for con-

[†]Division of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125 USA (yifanc@caltech.edu, eepperly@caltech.edu, jtropp@caltech.edu, rwebber@caltech.edu).

1

DRAFT

structing a low-rank psd approximation $\widehat{A}$ of a kernel matrix A. Given a set $\mathsf{S} = \{s_1, \ldots, s_k\} \subseteq \{1, \ldots, N\}$ of k column indices, the associated Nyström approximation is the rank-k psd matrix

$$(1.1) \qquad \widehat{A} = A(:, \mathsf{S}) A(\mathsf{S}, \mathsf{S})^\dagger A(\mathsf{S}, :) \in \mathbb{R}^{N \times N}.$$

In this expression, $A(:, \mathsf{S})$ is the submatrix with the selected columns, $A(\mathsf{S}, :)$ is the submatrix with the selected rows, and $A(\mathsf{S}, \mathsf{S})^\dagger$ is the Moore–Penrose pseudoinverse of the submatrix with the selected rows and columns. It can be shown [18, Eq. (1.2)] that the matrix (1.1) is the best psd approximation of A such that $\text{range}(\widehat{A}) \subseteq \text{range}(A(:, \mathsf{S}))$ and the residual $A - \widehat{A}$ is also psd.

Let us emphasize that the column Nyström approximation (1.1) only requires evaluating kN entries of the matrix A. As a consequence, this approximation is particularly well-suited for matrices that must be accessed by evaluating entry-wise formulas, such as kernel matrices [27] and Hamiltonian matrices from quantum chemistry [29, Ch. 2].

From a mathematical point of view, algorithms for column Nyström decomposition (1.1) differ primarily in how they select the set S of k columns. The challenge is to identify columns that capture as much action of the matrix as possible. This paper studies a natural approach, which we call the *randomly pivoted Cholesky* (RPCHOLESKY) algorithm (subsection 1.1). Similar approaches have been used in the past to approximate a rectangular matrix by a subset of its rows [8] or to sample from a determinantal point process [22], and a remark of [21] mentions RPCHOLESKY as a possibility for psd low-rank approximation. However, RPCHOLESKY has not been investigated as a serious strategy for psd matrix approximation before now.

Our work provides the first in-depth exploration of the empirical and theoretical behavior of RPCHOLESKY. Across a range of test problems, we find that RPCHOLESKY matches or improves over alternative methods for selecting columns in the Nyström approximation. To explain this surprising observation, we prove that RPCHOLESKY achieves a rigorous error bound that is nearly optimal within the class of Nyström algorithms. Altogether, these results make a strong case that RPCHOLESKY is the method of choice for low-rank approximation of a psd matrix under a constraint on the number of entry evaluations.

1.1. Randomly pivoted Cholesky.

We can best understand RPCHOLESKY as a member of a family of algorithms based on *partial Cholesky decomposition*. To that end, observe that the column Nyström approximation (1.1) can be constructed incrementally as follows. Beginning with $\widehat{A}^{(0)} := \mathbf{0}$ and $A^{(0)} := A$, the partial Cholesky decomposition performs the following operations at each step $i = 1, 2, \ldots, k$:

1. Select a pivot index $s_i \in \{1, \ldots, N\}$.
2. Update the Nyström approximation:

$$\widehat{A}^{(i)} = \widehat{A}^{(i-1)} + \frac{A^{(i-1)}(:, s_i) A^{(i-1)}(s_i, :)}{a_{s_i s_i}^{(i-1)}}.$$

3. Update the residual (Schur complement):

$$A^{(i)} = A^{(i-1)} - \frac{A^{(i-1)}(:, s_i) A^{(i-1)}(s_i, :)}{a_{s_i s_i}^{(i-1)}}.$$

By induction, each approximation $\widehat{A}^{(i)}$ and each residual $A^{(i)}$ remain psd. After k steps, the partial

Cholesky decomposition $\widehat{\boldsymbol{A}}^{(k)}$ coincides with the Nyström approximation (1.1) defined by the index set $\mathsf{S} = \{s_1, \ldots, s_k\}$. With a careful implementation (Algorithm 2.1), steps 2 and 3 involve just kN entry evaluations and $\mathcal{O}(k^2 N)$ arithmetic operations.

There are several standard methods for choosing the set S of pivots. A classical strategy is the *greedy method* [9], also known as diagonal or complete pivoting [16]. At each step i, the pivot s_i is the index of the largest diagonal entry of the current residual matrix, with ties broken arbitrarily:

$$(1.2) \qquad s_i \in \operatorname{argmax}\left\{ a_{jj}^{(i-1)} : 1 \le j \le N \right\}.$$

Another common strategy is to select each pivot s_i *uniformly at random* from $\{1, \ldots, N\}$, either with or without replacement [17, 34]. However, both the greedy method and the uniform sampling method can fail to find salient columns. The greedy method cannot explore the small diagonal entries of the residual matrix, while the uniform method cannot take advantage of the largest diagonal entries (subsections 4.1 and 4.2).

To circumvent this problem, we consider RPCHOLESKY as an alternative strategy for selecting the pivots. At each step i, this algorithm draws a random pivot s_i according to the probability distribution

$$(1.3) \qquad \mathbb{P}\{s_i = j\} = \frac{a_{jj}^{(i-1)}}{\operatorname{tr} \boldsymbol{A}^{(i-1)}} \quad \text{for each } j = 1, \ldots, N.$$

In other words, each pivot is sampled in proportion to the diagonal entries of the current residual matrix. RPCHOLESKY thus strikes a balance between exploring small diagonal entries and exploiting large ones. RPCHOLESKY was mentioned in [21] as a possible strategy for psd low-rank approximation, but we are not aware of any earnest investigation of its behavior.

Algorithm 1.1 describes a naive implementation of RPCHOLESKY, while subsection 2.1 presents a more efficient implementation.

Algorithm 1.1 RPCHOLESKY: Naive implementation

Input: Psd matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$, iteration count k or tolerance η
Output: Rank-k psd approximation $\widehat{\boldsymbol{A}}^{(k)}$

    Initialize $\widehat{\boldsymbol{A}}^{(0)} \leftarrow \boldsymbol{0}_{N \times N}$ and $\boldsymbol{A}_0 \leftarrow \boldsymbol{A}$
    for $i = 1$ to k **do**   ▷ Alternately, run until $\operatorname{tr} \boldsymbol{A}^{(i)} \le \eta \operatorname{tr} \boldsymbol{A}$
        Sample $s_i \sim \operatorname{diag} \boldsymbol{A}^{(i-1)} / \operatorname{tr} \boldsymbol{A}^{(i-1)}$ ▷ Sample new pivot
        $\widehat{\boldsymbol{A}}^{(i)} \leftarrow \widehat{\boldsymbol{A}}^{(i-1)} + \boldsymbol{A}^{(i-1)}(:, s_i) \boldsymbol{A}^{(i-1)}(s_i, :) / a_{s_i s_i}^{(i-1)}$ ▷ Update low-rank approximation
        $\boldsymbol{A}^{(i)} \leftarrow \boldsymbol{A}^{(i-1)} - \boldsymbol{A}^{(i-1)}(:, s_i) \boldsymbol{A}^{(i-1)}(s_i, :) / a_{s_i s_i}^{(i-1)}$ ▷ Update Schur complement
    end for

1.2. Numerical results. Given that RPCHOLESKY is a natural algorithm, we were perplexed by the lack of attention that it has received in the literature on randomized matrix computations. Therefore, we undertook a suite of experiments to compare RPCHOLESKY with other column Nyström approximations on a range of test problems. Surprisingly, we discovered that RPCHOLESKY reliably produces some of the best matrix approximations using the fewest number of entry evaluations. This work is documented in section 2.

For now, let us offer a visually appealing example that is representative of our broader findings. We aim to approximate the kernel matrices associated with two data sets:

1. **Smile:** A Gaussian kernel matrix constructed from 10^4 data points depicting a smile in $\mathbb{R}^2$. The eyes are constructed from just 10^2 points, making them easy to miss for certain sampling methods.
2. **Outliers:** A Gaussian kernel matrix constructed from a point cloud of 10^4 points in $\mathbb{R}^{20}$. Of these points, 50 are outliers that are positioned very far away from the others.
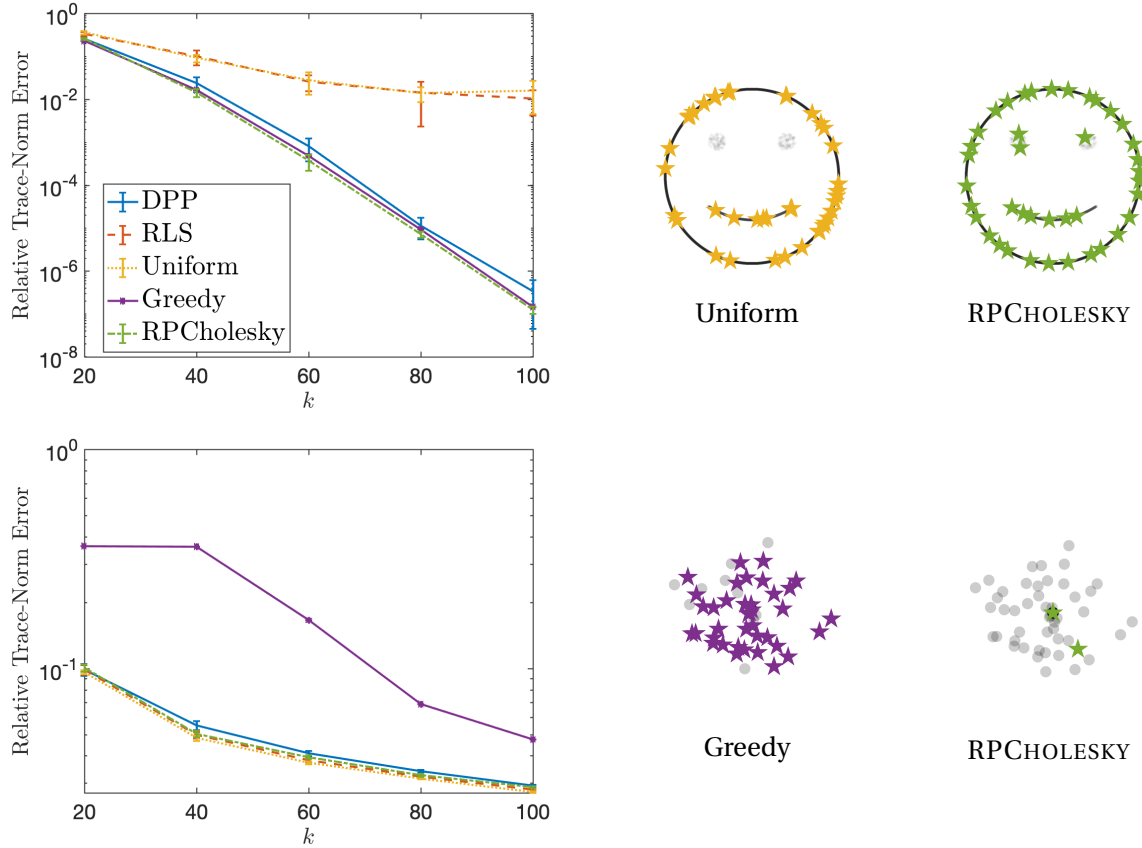
Figure 1: *Left:* Mean relative trace-norm error $\operatorname{tr}(\boldsymbol{A} - \widehat{\boldsymbol{A}}^{(k)})/\operatorname{tr}\boldsymbol{A}$ and standard deviation error bars for different Nyström methods. *Right:* Selected pivots (colored stars) and data points (grey circles) for the **Smile** and **Outliers** data, with **Outliers** data projected onto a random plane. The greedy method chooses many pivots from the 50 outliers, while RPCHOLESKY mostly avoids the outliers.

Figure 1 charts the accuracy of the column Nyström approximations obtained with different pivoting strategies. The left-hand panels show the average relative trace-norm error over 100 independent trials. The right-hand panels show the pivots selected by several Nyström methods with $k = 40$. For additional computational details, see Appendix B.

These tests bring to light the failure modes of uniform sampling and the greedy method. Uni-

form sampling fails to select the pivots representing small, important features of the data, such as the eyes in the **Smile** example. The greedy method heavily emphasizes outliers, leading to poor approximation accuracy in the **Outliers** example. By contrast, RPCHOLESKY avoids these failure modes and achieves high accuracy for both test problems.

For completeness, we also compare RPCHOLESKY with two other column selection schemes for Nyström approximation: *determinantal point process* (DPP) sampling [7] and *ridge leverage score* (RLS) sampling [1, 4, 20, 26]. These methods offer strong theoretical guarantees, yet existing implementations are complicated and require far more entry evaluations than RPCHOLESKY to achieve the same approximation quality. See subsections 4.3 and 4.4 for full details about these approaches.

Figure 2 displays the computational cost of different column Nyström methods, as measured by the number of entry evaluations. This is the total number of entry evaluations required to first form the sampling distribution and then generate the sampled columns. Uniform sampling is the cheapest Nyström method, requiring just kN entry evaluations. The greedy method and RPCHOLESKY follow closely behind, requiring $(k+1)N$ entry evaluations. RLS sampling requires roughly $3kN$ entry evaluations and DPP sampling (using the vfx sampler [11]) requires between $80kN$ and $1000kN$ entry evaluations, making these latter two methods comparatively very expensive.
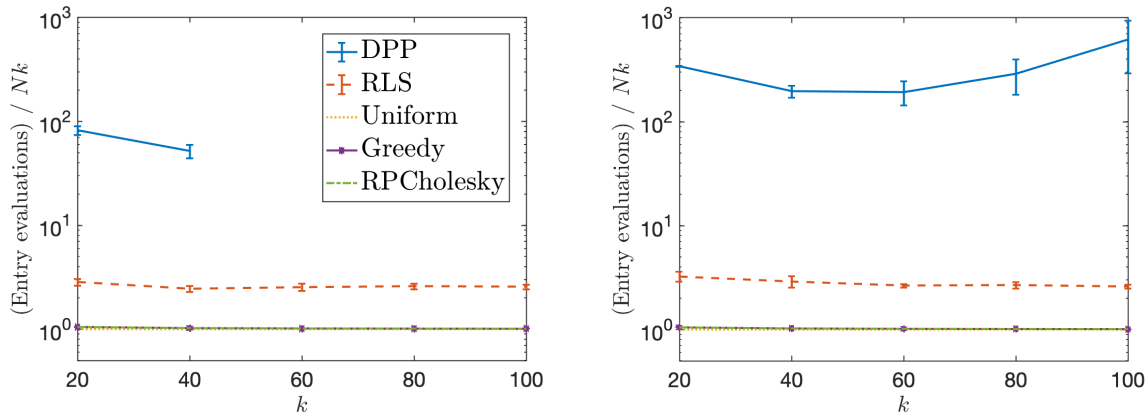

Figure 2: Entry evaluations for different Nyström methods applied to the **Smile** (*left*) and **Outliers** (*right*) matrices. The vfx DPP sampler fails for $k \geq 60$ on the **Smile** matrix.

In addition to being expensive, DPP sampling and RLS sampling exhibit other limitations. For the **Smile** kernel matrix with $k \geq 60$, DPP sampling using the vfx sampler [11] returns an error message due to the matrix A being close to low-rank and fails to produce any output. It is necessary to use a slower GS sampler [11] based on a complete eigendecomposition of the matrix A. For the **Smile** kernel matrix, RLS sampling provides a low-quality approximation, often failing to include data points from one or both eyes. The poor RLS sampling results are consistent with the limited theoretical error bounds for this method (see subsection 4.4), which necessitate using hundreds of columns to guarantee an accurate low-rank approximation for any psd matrix.

1.3. Theoretical results. After observing the excellent performance of the RPCHOLESKY algorithm empirically, we sought a more rigorous explanation. This paper proves that RPCHOLESKY

attains error bounds that are nearly optimal within the class of column Nyström approximations. Section 3 studies the theoretical performance of RPCHOLESKY, and section 4 outlines results for competing algorithms. We summarize the comparison here.

Let A be a psd matrix, and let $\widehat{A}$ be a column Nyström approximation (1.1) of A. We consider the approximation error as $\mathrm{tr}(A - \widehat{A})$, which is variously known as the trace norm, nuclear norm, or Schatten 1-norm of the residual $A - \widehat{A}$. We compare against the error attained by a best rank-r psd approximation $[\![A]\!]_r$, obtained from an r-truncated eigenvalue decomposition.

A randomized column Nyström approximation $\widehat{A}$ is called an (r, ε)-*approximation* of A when

$$(1.4) \qquad\qquad \mathbb{E}\,\mathrm{tr}(A - \widehat{A}) \le (1 + \varepsilon) \cdot \mathrm{tr}(A - [\![A]\!]_r)$$

for parameters $r \in \mathbb{N}$ and $\varepsilon > 0$. The expectation averages over the random choices in the algorithm (e.g., the columns that are sampled). The efficiency of a particular sampling scheme depends on how large we must take the number k of columns. Here is the key theoretical question:

Question 1.1. *How many columns k are sufficient to guarantee that a randomized column Nyström method attains an (r, ε)-approximation* (1.4) *for every $N \times N$ psd input matrix?*

To achieve an (r, ε)-approximation for the worst-case matrix, a column Nyström approximation must use at least $k \ge r/\varepsilon$ columns. See Theorem 4.3(c) for an example. We will establish the remarkable fact that RPCHOLESKY achieves an (r, ε)-approximation for every psd matrix after accessing only a modest multiple of r/ε columns.

Table 1 presents the best available upper bounds on the number k of columns for RPCHOLESKY and other column Nyström approximation methods to achieve (1.4). These bounds are expressed terms of r, ε, N, and the relative approximation error

$$\eta := \frac{\mathrm{tr}(A - [\![A]\!]_r)}{\mathrm{tr}\,A} \in [0, 1].$$

Method	Number k of columns	Reference
Greedy method	$(1 - (1 + \varepsilon)\eta)N$	Theorem 4.1
Uniform sampling*	$\frac{r-1}{\varepsilon\eta} + \frac{1}{\varepsilon}$	Theorem 4.2
DPP sampling	$\frac{r}{\varepsilon} + r - 1$	Theorem 4.3
RLS sampling**	$136\frac{r}{\varepsilon}\log\left(\frac{3r}{\varepsilon\eta}\right)$	Corollary 4.5
RPCHOLESKY	$\frac{r}{\varepsilon} + r\log_+\left(\frac{1}{\varepsilon\eta}\right)$	Theorem 3.1
RPCHOLESKY	$\frac{r}{\varepsilon} + r + r\log_+\left(\frac{2^r}{\varepsilon}\right)$	Theorem 3.1

Table 1: Upper bounds on the number of columns for column Nyström approximation methods to produce an (r, ε)-approximation (1.4). *The result for uniform sampling assumes the diagonal entries of A are set to one. **The result for RLS sampling assumes that $\varepsilon < \frac{1}{4}$.

DPP sampling has the strongest error bounds of all the methods in Table 1. To achieve an (r, ε)-

approximation with DPP sampling, it is sufficient to take the number k of columns as

$$(1.5) \qquad k \geq \frac{r}{\varepsilon} + r - 1.$$

RPCHOLESKY has the second strongest error bounds, requiring no more columns than

$$(1.6) \qquad k \geq \frac{r}{\varepsilon} + r \log_+\left(\frac{1}{\varepsilon\eta}\right),$$

where $\log_+(x) := \max\{\log x, 0\}$ for $x > 0$. The main difference between the DPP result (1.5) and the RPCHOLESKY result (1.6) is the multiplicative factor $\log(1/\eta)$ present in the latter. However, because the relative error η appears inside the logarithm, this factor has only a modest impact on the computational scaling. Indeed, $\log(1/\eta) < 37$ when $\eta > 10^{-16}$, the machine precision in double-precision arithmetic. See section 3 for additional RPCHOLESKY error bounds, which consider the limit as $\eta \to 0$.

The error bounds for RPCHOLESKY are *significantly stronger* than the bounds for the greedy method and uniform sampling. For a worst-case matrix A, the greedy method requires $\Theta(N)$ columns to approach the best rank-r approximation error, and the uniform sampling method requires $\Theta(r/\eta)$ columns (subsections 4.1 and 4.2). In contrast, RPCHOLESKY requires a number of columns that is independent of the dimension N and depends only logarithmically on the relative error η. These results rigorously justify why RPCHOLESKY does not exhibit the same failure modes as the greedy method and the uniform sampling method.

Our theoretical work confirms that RPCHOLESKY achieves a given approximation error with roughly the same number of columns as the best theoretical approach for column Nyström approximation. At the same time, RPCHOLESKY uses fewer entry evaluations than other more complicated algorithms to attain this level of error. Given the simplicity, efficiency, and reliability of RPCHOLESKY, we believe that there is a strong case for deploying this algorithm in applications.

1.4. Plan for paper. The rest of the paper is organized as follows. Section 2 describes a fast implementation of RPCHOLESKY and presents numerical demonstrations of RPCHOLESKY in machine learning tasks. Section 3 presents the theoretical analysis of RPCHOLESKY, while section 4 discusses the performance of alternative methods. Section 5 surveys future directions.

1.5. Notation. The symbol $\preceq$ denotes the *psd order* on symmetric matrices: $H \preceq A$ if and only if $A - H$ is psd. Submatrices are expressed using MATLAB notation, for example, $A(:, i)$ represents the ith column of A and $A(S, :)$ denotes the submatrix of A with rows indexed by the set S. RPCHOLESKY and its analysis extend to *complex-valued* psd matrices without significant modification, but for simplicity this work focuses on the real setting. The transpose of A is denoted A^*, which becomes the *conjugate transpose* for complex matrices.

2. Algorithms and experiments. This section gives efficient pseudocode for RPCHOLESKY (subsection 2.1) and presents a range of numerical examples (subsections 2.2 and 2.3).

2.1. Speeding up RPCholesky. To accelerate the naive implementation of RPCHOLESKY in Algorithm 1.1, we can apply the following techniques:

- The Nyström approximation $\widehat{A}$ can be stored and updated in factored form $\widehat{A} = FF^*$ where $F \in \mathbb{R}^{N \times k}$ is a thin matrix.

DRAFT

- Rather than forming and maintaining the residual matrix $A^{(i)}$, it suffices to generate the single column $A^{(i)}(:, s_i)$ at every iteration $i = 1, 2, \ldots, k$.

The resulting RPCHOLESKY algorithm requires $(k+1)N$ entry evaluations and $2k^2 N/3$ arithmetic operations; see Algorithm 2.1.

Algorithm 2.1 RPCHOLESKY: Efficient implementation

Input: Psd matrix $A \in \mathbb{R}^{N \times N}$ and iteration count k or tolerance η
Output: Factor matrix $F \in \mathbb{R}^{N \times k}$ defining Nyström approximation $\widehat{A} = FF^*$ and selected pivots S

    Initialize $F \leftarrow \mathbf{0}_{N \times k}$, $\mathsf{S} \leftarrow \varnothing$, and $d \leftarrow \operatorname{diag} A$
    for $i = 1$ to k **do** $\triangleright$ Alternately, run until $\sum_{j=1}^{N} d_j \le \eta \operatorname{tr} A$
        Sample $s \sim d / \sum_{j=1}^{N} d_j$
        $\mathsf{S} \leftarrow \mathsf{S} \cup \{s\}$
        $g \leftarrow A(:, s) - F(:, 1:i-1) F(s, 1:i-1)^*$
        $F(:, i) \leftarrow g / \sqrt{g_s}$
        $d \leftarrow d - |F(:, i)|^2$ $\triangleright$ $|F(:, i)|^2$ is the vector with jth entry $|F(j, i)|^2$
    end for

To provide an even greater speedup in some settings, RPCHOLESKY can be modified to sample multiple columns in a single iteration, as in Algorithm 2.2. In experiments, this *blocked* version of RPCHOLESKY produces similar quality approximations to standard RPCHOLESKY, and it reduces the runtime when generating multiple columns of A is faster then generating columns one at a time. For a discussion of these runtime considerations, see Appendix B.

Algorithm 2.2 RPCHOLESKY: Blocked variant

Input: Psd matrix $A \in \mathbb{R}^{N \times N}$, iteration count k, and block size B
Output: Factor matrix $F \in \mathbb{R}^{N \times k}$ defining Nyström approximation $\widehat{A} = FF^*$ and selected pivots S

    Initialize $F \leftarrow \mathbf{0}_{N \times k}$, $\mathsf{S} \leftarrow \varnothing$, $d \leftarrow \operatorname{diag} A$, and $i \leftarrow 0$
    while $i < k$ **do**
        Sample $s_1, s_2, \ldots, s_{\min(B, k-i)} \overset{\text{iid}}{\sim} d / \sum_{j=1}^{N} d_j$
        $\mathsf{U} \leftarrow \text{UNIQUE}(s_1, s_2, \ldots, s_{\min(B, k-i)})$ $\triangleright$ Identify unique elements
        $\mathsf{S} \leftarrow \mathsf{S} \cup \mathsf{U}$
        $G \leftarrow A(:, \mathsf{U}) - F(:, 1:i) [F(\mathsf{U}, 1:i)]^*$
        $R \leftarrow \text{CHOL}(G(\mathsf{U}, :))$ $\triangleright$ $G(\mathsf{U}, :) = R^* R$
        $F(:, i : i + |\mathsf{U}|) \leftarrow G R^{-1}$
        $d \leftarrow d - \text{SQUAREDROWNORMS}(F(:, i : i + |\mathsf{U}|))$
        $i \leftarrow i + |\mathsf{U}|$
    end while

2.2. Kernel ridge regression. One powerful application for RPCHOLESKY is to accelerate *kernel ridge regression* (KRR) [27, §4.9.1]. KRR is a nonlinear extension of least-squares regression that approximates an unknown input–output map using a positive definite kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and a set of input–output pairs $(x^{(1)}, y_1), \ldots, (x^{(N)}, y_N)$. As output, KRR provides a prediction

function of the form

$$f(\cdot\,; \boldsymbol{\beta}) := \sum_{i=1}^{N} \beta_i K\left(\boldsymbol{x}^{(i)}, \cdot\right), \tag{2.1}$$

with the coefficient vector $\boldsymbol{\beta}$ chosen to minimize a regularized least-squares loss:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^N} \frac{1}{N} \sum_{j=1}^{N} \left| f\left(\boldsymbol{x}^{(j)}; \boldsymbol{\beta}\right) - y_j \right|^2 + \lambda \sum_{i,j=1}^{N} \beta_i \beta_j K(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}).$$

Explicitly, the vector $\boldsymbol{\beta}$ is the solution to a linear system

$$\boldsymbol{\beta} = (\boldsymbol{A} + \lambda N \mathbf{I})^{-1} \boldsymbol{y}, \tag{2.2}$$

where $\boldsymbol{A}$ is the $N \times N$ kernel matrix and $\boldsymbol{y}$ is the vector of output values.

Directly computing the $\boldsymbol{\beta}$ coefficients by (2.2) would require solving a dense linear system at $\mathcal{O}(N^3)$ cost, but RPCHOLESKY provides a faster alternative. First, we apply RPCHOLESKY to the kernel matrix $\boldsymbol{A}$ to generate a random set of pivots $\mathsf{S} = \{s_1, \ldots, s_k\}$. Next, we identify a restricted prediction function

$$\widehat{f}(\cdot\,; \widehat{\boldsymbol{\beta}}) = \sum_{i=1}^{k} \widehat{\beta}_i K\left(\boldsymbol{x}^{(s_i)}, \cdot\right) \tag{2.3}$$

that minimizes the loss

$$\min_{\widehat{\boldsymbol{\beta}} \in \mathbb{R}^k} \frac{1}{N} \sum_{j=1}^{N} \left| \widehat{f}\left(\boldsymbol{x}^{(j)}; \widehat{\boldsymbol{\beta}}\right) - y_j \right|^2 + \lambda \sum_{i,j=1}^{k} \widehat{\beta}_i \widehat{\beta}_j K(\boldsymbol{x}^{(s_i)}, \boldsymbol{x}^{(s_j)}).$$

The coefficient vector $\widehat{\boldsymbol{\beta}} \in \mathbb{R}^k$ satisfies a smaller linear system

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{A}(\mathsf{S},:)\boldsymbol{A}(:,\mathsf{S}) + \lambda N \boldsymbol{A}(\mathsf{S},\mathsf{S})\right)^{-1} \boldsymbol{A}(\mathsf{S},:)\boldsymbol{y}. \tag{2.4}$$

involving a $k \times k$ matrix, which makes it relatively inexpensive to solve.

Algorithm 2.3 provides pseudocode for RPCHOLESKY-accelerated KRR. The most expensive steps are the RPCHOLESKY step and evaluation of the matrix–matrix product $\boldsymbol{A}(\mathsf{S},:)\boldsymbol{A}(:,\mathsf{S})$; each requires $\mathcal{O}(k^2 N)$ operations. After performing these steps, the coefficient vector $\widehat{\boldsymbol{\beta}}$ can be computed in $\mathcal{O}(k^3)$ operations via a standard dense linear solver. Evaluating the prediction function (2.3) from RPCHOLESKY-accelerated KRR requires just $\mathcal{O}(k)$ operations, which improves over the $\mathcal{O}(N)$ cost of evaluating (2.1).

To demonstrate the effectiveness of RPCHOLESKY-accelerated KRR, we use Algorithm 2.3 to predict the energy of the highest occupied molecular orbital (HOMO) for small organic molecules from the QM9 data set [24, 25]. The HOMO energy quantifies the electron-donating capacity of a molecule, and it is typically obtained by expensive first-principles calculations. Estimating HOMO energies and related properties from data has been a major goal of chemical machine learning [28].

To represent molecules as vectors for KRR, we apply the Coulomb matrix representation with ℓ_2 row norm sorting [28, §III.A], a standard approach based on Coulomb repulsions between atomic

---

Algorithm 2.3 RPCHOLESKY-accelerated kernel ridge regression

Input: Data points $X = \{x^{(1)}, \ldots, x^{(N)}\} \subseteq \mathbb{R}^d$; output values $y \in \mathbb{R}^N$; approximation rank k; regularization parameter $\lambda > 0$

Output: Pivots S and coefficients $\widehat{\beta}$ defining a prediction function $\widehat{f}(\cdot)$ by (2.3)

  $A \leftarrow \text{KERNELMATRIX}(X)$
  $(\sim, S) \leftarrow \text{RPCHOLESKY}(A, k)$
  $\widehat{\beta} \leftarrow \left(A(S,:)A(:,S) + \lambda N\, A(S,S) \right)^{-1} A(S,:)\, y$

---

nuclei. We standardize the data and, following [28], evaluate the similarity between data points using the ℓ_1 Laplace kernel:

$$(2.5) \qquad K(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{1}{\sigma} \sum_{j=1}^{d} |x_j - y_j| \right).$$

We divide the data into 100,000 data points for training and roughly 33,000 data points for testing and set the bandwidth σ and ridge parameter λ using cross-validation. We measure out-of-sample prediction errors using the symmetric mean absolute percentage error (SMAPE)

$$(2.6) \qquad \text{SMAPE} = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \frac{\left| y_i^{\text{test}} - \widehat{f}(\boldsymbol{x}^{(i),\text{test}}; \widehat{\boldsymbol{\beta}}) \right|}{(|y_i^{\text{test}}| + |\widehat{f}(\boldsymbol{x}^{(i),\text{test}}; \widehat{\boldsymbol{\beta}})|)/2}.$$

Forming and storing the full kernel matrix for the QM9 data set would require 40 GB and 4 trillion arithmetic operations ($20{,}000\times$ the operation count for the **Smile** matrix from subsection 1.2), making direct KRR highly expensive. Prior work [28] applied KRR to this data set using a random subsample of $N \leq 64{,}000$ training points. Here, RPCHOLESKY allows us to use take advantage of more of the data ($N = 100{,}000$ training points) at a reduced computational cost by using a small approximation rank $k \leq 1000 \ll N$.

Figure 3 displays the results. The prediction accuracies for RPCHOLESKY, uniform sampling, and RLS sampling are very close, with uniform sampling being the slight favorite. However, an examination shows that **uniform sampling has $18\times$ higher prediction errors** than RPCHOLESKY for the molecules in the data set with the largest number of atoms (29); see Table 2. This observation suggests that RPCHOLESKY can be better than uniform sampling at representing less populated regions of data space, as seen earlier in the **Smile** example (subsection 1.2). The greedy method has notably worse performance than the three other methods, and we were unable to use DPP sampling because of the large values of $N = 10^5$ and $k = 10^3$.

2.3. Spectral clustering. We can also use RPCHOLESKY to accelerate *kernel spectral clustering* [33]. Kernel spectral clustering is a machine learning approach that uses a nonnegative, positive definite kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ to identify features in a data set $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}$ and applies clustering based on those features.

In a standard formulation of kernel spectral clustering [33], the features are defined by the dominant left eigenvectors $\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(m)}$ of the matrix $\boldsymbol{D}^{-1}\boldsymbol{A}$. In this expression, $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ is the kernel matrix and $\boldsymbol{D} \in \mathbb{R}^{N \times N}$ is the diagonal matrix that lists the row sums of $\boldsymbol{A}$. The eigenvectors are
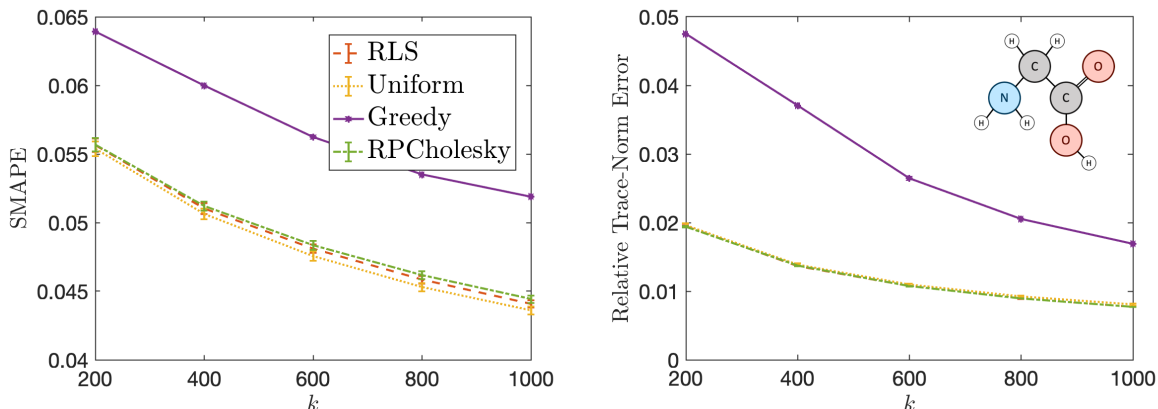
Figure 3: *Left:* Prediction error (2.6) for different Nyström algorithms. *Right:* Relative trace-norm error for different Nyström algorithms.

Table 2: Prediction error (2.6) for the largest molecules in the QM9 dataset

Compound #	Composition	Uniform	RPCHOLESKY
1,996	CC(C)CC1CO1	0.519	0.030
8,664	CC(CO)(C=O)C=O	0.582	0.016
13,812	CC(O)C1CC1CO	0.523	0.033
64,333	CC1(C)CCOC(=N)O1	0.525	0.027
81,711	OC1C2CC1OCCO2	0.576	0.016
109,816	CCC1C(O)C1CC#C	0.514	0.023
118,229	COCC(C)OC(C)C	0.575	0.075
122,340	CC1C(CCO)N1C	0.557	0.017
131,819	OCCCN1C=NC=N1	0.570	0.030

normalized such that $\left(\boldsymbol{v}^{(i)}\right)^{*}\boldsymbol{D}\boldsymbol{v}^{(j)}=\delta_{ij}$, and each data point $\boldsymbol{x}^{(i)}$ is projected as

$$\widetilde{\boldsymbol{x}}^{(i)}=\left(\boldsymbol{v}_{i}^{(1)},\boldsymbol{v}_{i}^{(2)},\ldots,\boldsymbol{v}_{i}^{(m)}\right).$$

The projected data points $\widetilde{\boldsymbol{x}}^{(1)},\ldots,\widetilde{\boldsymbol{x}}^{(N)}$ are clustered using any standard method, such as k-means with m centers.

Directly computing the eigendecomposition of $\boldsymbol{D}^{-1}\boldsymbol{A}$ would require $\mathscr{O}(N^3)$ operations. However, RPCHOLESKY provides a cheaper alternative (Algorithm 2.4) that involves just $\mathscr{O}(k^2 N)$ operations. In the RPCHOLESKY-accelerated approach, a low-rank approximation $\widehat{\boldsymbol{A}}^{(k)}$ is used in place of the kernel matrix $\boldsymbol{A}$, and the diagonal matrix $\widehat{\boldsymbol{D}}$ containing the row sums of $\widehat{\boldsymbol{A}}^{(k)}$ is used in place of $\boldsymbol{D}$. The data is projected onto the dominant normalized left eigenvectors of $\widehat{\boldsymbol{D}}^{-1}\widehat{\boldsymbol{A}}^{(k)}$ and the projected data points are clustered using k-means.

DRAFT

Algorithm 2.4 RPCHOLESKY-accelerated spectral clustering

Input: Data points $\mathsf{X} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\} \subseteq \mathbb{R}^d$; number m of clusters; approximation rank k
Output: Partition of X into m clusters $\mathsf{X} = \mathsf{X}_1 \cup \cdots \cup \mathsf{X}_m$

   $\boldsymbol{A} \leftarrow \text{KERNELMATRIX}(\mathsf{X})$
   $\boldsymbol{F} \leftarrow \text{RPCHOLESKY}(\boldsymbol{A}, k)$
   $\widehat{\boldsymbol{D}} \leftarrow \text{diag}(\boldsymbol{F}(\boldsymbol{F}^* \mathbb{1}))$
   $\boldsymbol{G} \leftarrow \widehat{\boldsymbol{D}}^{-1/2} \boldsymbol{F}$
   $(\boldsymbol{Q}, \boldsymbol{R}) \leftarrow \text{QR}(\boldsymbol{G}, \texttt{'econ'})$
   $(\boldsymbol{U}, \sim, \sim) \leftarrow \text{SVD}(\boldsymbol{R})$
   $\boldsymbol{V} \leftarrow \widehat{\boldsymbol{D}}^{-1/2} \boldsymbol{Q} \boldsymbol{U}$
   $\widetilde{\mathsf{X}} \leftarrow \{\boldsymbol{V}(i, 1:m) : 1 \le i \le |\mathsf{X}|\}$
   $\mathsf{X}_1, \ldots, \mathsf{X}_m \leftarrow \text{K-MEANS}(\widetilde{\mathsf{X}}, m)$

In Figure 4, we apply RPCHOLESKY-accelerated spectral clustering to separate the letters "RPC", written in papyrus font. After reading 140 of the 2187 columns of the kernel matrix, RPCHOLESKY reliably produces a perfect clustering as shown in Figure 4 (top left). In contrast, given the same approximation rank, RLS and uniform sampling frequently produce an incorrect clustering. The fraction of perfect clusterings over 100 independent trials is given at the bottom of Figure 4. For this example, the greedy method outperforms all the randomized sampling methods, but we have already seen (subsections 1.2 and 2.2) that the greedy method can be unreliable and has serious failure modes.

3. Theoretical analysis of RPCholesky. Given the appealing computational profile of the RPCHOLESKY algorithm, we would like to understand when it is guaranteed to produce an accurate low-rank approximation. In this section, we establish the following new result:

Theorem 3.1 (Randomly pivoted Cholesky). *Fix $r \in \mathbb{N}$ and $\delta \in (0,1)$, and let $\boldsymbol{A}$ be a psd matrix. The column Nyström approximation $\widehat{\boldsymbol{A}}^{(k)}$ produced by* RPCHOLESKY *(Algorithm 2.1) attains the bound*

$$(3.1) \qquad \mathbb{E}\,\text{tr}\left(\boldsymbol{A} - \widehat{\boldsymbol{A}}^{(k)}\right) \le \text{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r) + \delta\,\text{tr}\,\boldsymbol{A},$$

provided that the number k of columns satisfies

$$(3.2) \qquad k \ge \frac{r\eta}{\delta} + \min\left\{r\log\left(\frac{1}{\delta}\right), r + r\log_+\left(\frac{2^r\eta}{\delta}\right)\right\}.$$

The relative error η is defined by $\eta := \text{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r) / \text{tr}(\boldsymbol{A})$. As usual, $\log_+(x) := \max\{\log x, 0\}$ for $x > 0$.

The major takeaway from Theorem 3.1 is that the number k of columns depends *logarithmically* on the tolerance $1/\delta$, as long as $\delta \ge \eta$. The error bounds for RPCHOLESKY become more powerful as $\eta \to 0$ and RPCHOLESKY achieves the theoretically optimal error

$$\mathbb{E}\,\text{tr}\left(\boldsymbol{A} - \widehat{\boldsymbol{A}}^{(k)}\right) = \text{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r)$$

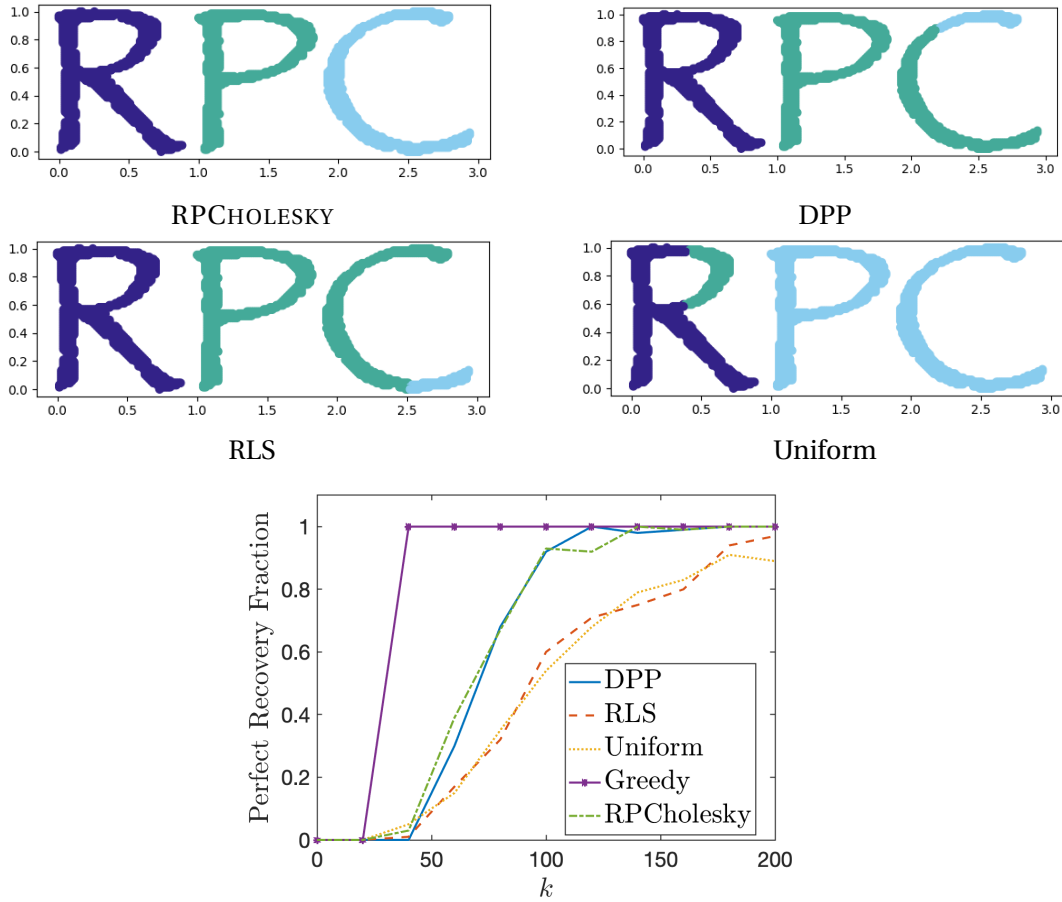with just $k = r$ columns when $\eta = 0$.

DRAFT

Figure 4: *Top:* A correct clustering produced by RPCHOLESKY along with selected incorrect clusterings produced by RLS, DPP, and uniform sampling with rank $k = 140$. *Bottom:* Fraction of correct clusterings produced by Nyström algorithms over 100 independent trials.

By setting $\delta = \varepsilon\eta$ for $\varepsilon > 0$, Theorem 3.1 guarantees that RPCHOLESKY must produce an (r, ε)-approximation as soon as the number k of columns satisfies

$$k \geq \frac{r}{\varepsilon} + r \log_+ \left(\frac{1}{\varepsilon\eta} \right).$$

In practice, the logarithmic factor is typically a modest constant, so the bound is comparable with the minimal cost of $k \geq \frac{r}{\varepsilon}$ columns. Thus, RPCHOLESKY improves on the greedy method, in which the number k of columns may be proportional to the dimension N in the worst case. It also improves on the uniform sampling method, in which the number k of columns can depend *polynomially* on the relative error $1/\eta$. See subsections 4.1 and 4.2 for details.

Additionally, Theorem 3.1 demonstrates that RPCHOLESKY produces an (r, ε)-approximation

when the number k of columns satisfies

$$(3.3) \qquad k \geq \frac{r}{\varepsilon} + r + r\log_+\left(\frac{2^r}{\varepsilon}\right).$$

This alternative error bound is significant because it completely eliminates the dependence on the relative error η. When r is small, (3.3) is competitive with the cost of $k \geq \frac{r}{\varepsilon} + r - 1$ columns for DPP sampling. However, when r is large, (3.3) deteriorates due to the quadratic dependence on r.

Although some error bounds for RPCHOLESKY can be deduced from earlier work [8, Thms. 4–5 & Prop. 1], Theorem 3.1 gives bounds that are strictly sharper.

3.1. Proof of Theorem 3.1.
The proof of Theorem 3.1 is based on the properties of the *expected residual function*:

$$(3.4) \qquad \boldsymbol{\Phi}(\boldsymbol{A}) := \mathbb{E}\left[\boldsymbol{A}^{(1)} \mid \boldsymbol{A}\right] = \boldsymbol{A} - \frac{\boldsymbol{A}^2}{\operatorname{tr}\boldsymbol{A}}.$$

This function returns the expectation of the residual $\boldsymbol{A}^{(1)}$ after applying one step of RPCHOLESKY to $\boldsymbol{A}$. The equality (3.4) follows from a short computation using the sampling distribution (1.3) and the definition of the residual (Algorithm 1.1). Note that $\boldsymbol{\Phi}$ is defined on psd matrices of any dimension.

The first lemma describes some basic facts about the expected residual function $\boldsymbol{\Phi}$. We postpone the proof to subsection 3.2.

Lemma 3.2 (Expected residual). *The expected residual map $\boldsymbol{\Phi}$ defined in (3.4) is positive, monotone, and concave with respect to the psd order. That is, for all psd $\boldsymbol{A}, \boldsymbol{H}$ with the same dimensions,*

$$(3.5) \qquad \boldsymbol{0} \preceq \boldsymbol{\Phi}(\boldsymbol{A}) \preceq \boldsymbol{\Phi}(\boldsymbol{A} + \boldsymbol{H});$$

$$(3.6) \qquad \theta\boldsymbol{\Phi}(\boldsymbol{A}) + (1-\theta)\boldsymbol{\Phi}(\boldsymbol{H}) \preceq \boldsymbol{\Phi}(\theta\boldsymbol{A} + (1-\theta)\boldsymbol{H}) \quad \text{for all } \theta \in [0,1].$$

The second lemma describes how the trace of the residual declines after multiple steps of the RPCHOLESKY procedure. The proof appears in subsection 3.3.

Lemma 3.3 (Contraction rate). *Consider the k-fold composition of the expected residual (3.4):*

$$\boldsymbol{\Phi}^{\circ k} := \underbrace{\boldsymbol{\Phi} \circ \boldsymbol{\Phi} \circ \cdots \circ \boldsymbol{\Phi}}_{k \text{ times}} \quad \text{for each } k \in \mathbb{N}.$$

Fix $r \in \mathbb{N}$. For each psd matrix $\boldsymbol{H}$ and each $\Delta > 0$,

$$\operatorname{tr}\boldsymbol{\Phi}^{\circ k}(\boldsymbol{H}) \leq \operatorname{tr}(\boldsymbol{H} - [\![\boldsymbol{H}]\!]_r) + \Delta\operatorname{tr}\boldsymbol{H} \quad \text{when} \quad k \geq \frac{r\operatorname{tr}(\boldsymbol{H} - [\![\boldsymbol{H}]\!]_r)}{\Delta\operatorname{tr}\boldsymbol{H}} + r\log_+\left(\frac{1}{\Delta}\right).$$

Last, we present a bound which shows that the error after k steps of RPCHOLESKY is comparable with the error in the best rank-k approximation. The proof appears in subsection 3.4.

Lemma 3.4 (Error doubling). *For each psd matrix $\boldsymbol{A}$, the residual matrix $\boldsymbol{A}^{(k)}$ after applying k steps of RPCHOLESKY satisfies*

$$\mathbb{E}\operatorname{tr}\boldsymbol{A}^{(k)} \leq 2^k\operatorname{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_k) \quad \text{for each } k \in \mathbb{N}.$$

With these results at hand, we quickly establish the main error bound for RPCHOLESKY.

Proof of Theorem 3.1. Let A be a psd matrix. By the concavity (3.6) of the expected residual map (3.4) and the matrix version of Jensen's inequality [5, Thm. 4.16], the residual matrices satisfy

$$\mathbb{E}\, A^{(j)} = \mathbb{E}\Big[\mathbb{E}\Big[A^{(j)}\,\Big|\,A^{(j-1)}\Big]\Big] = \mathbb{E}\,\Phi(A^{(j-1)}) \preceq \Phi(\mathbb{E}(A^{(j-1)})) \quad \text{for each } j \in \{1,\dots,k\}.$$

Next, by monotonicity (3.5) of Φ and the last display,

$$\mathbb{E}\, A^{(k)} \preceq \Phi(\mathbb{E}\, A^{(k-1)}) \preceq \Phi \circ \Phi(\mathbb{E}\, A^{(k-2)}) \preceq \cdots \preceq \Phi^{\circ k}(A).$$

Using the fact that the trace is linear and preserves the psd order,

$$\mathbb{E}\,\mathrm{tr}\big(A - \widehat{A}^{(k)}\big) = \mathbb{E}\,\mathrm{tr}\, A^{(k)} \leq \mathrm{tr}\,\Phi^{\circ k}(A).$$

Next, we apply Lemma 3.3 with $H = A$ and $\Delta = \delta$ to see that

$$\mathbb{E}\,\mathrm{tr}\big(A - \widehat{A}^{(k)}\big) \leq \mathrm{tr}(A - [\![A]\!]_r) + \delta\,\mathrm{tr}\, A \quad \text{when} \quad k \geq \frac{r\eta}{\delta} + r\log\left(\frac{1}{\delta}\right),$$

where we have recognized the relative error η.

By a similar argument,

$$\mathbb{E}\,\mathrm{tr}\big(A - \widehat{A}^{(k)}\big) = \mathbb{E}\,\mathrm{tr}\, A^{(k)} \leq \mathrm{tr}\,\Phi^{\circ(k-r)}\big(\mathbb{E}\, A^{(r)}\big).$$

We can apply Lemma 3.3 with $H = \mathbb{E}\, A^{(r)}$ and $\Delta = \delta \cdot \mathrm{tr}\, A / \mathbb{E}\,\mathrm{tr}\, A^{(r)}$ to see that

$$(3.7) \qquad\qquad \mathbb{E}\,\mathrm{tr}\big(A - \widehat{A}^{(k)}\big) \leq \mathrm{tr}\big(\mathbb{E}\, A^{(r)} - [\![\mathbb{E}\, A^{(r)}]\!]_r\big) + \delta\,\mathrm{tr}\, A$$

provided the number of columns satisfies

$$(3.8) \qquad\qquad k - r \geq \frac{r\,\mathrm{tr}\big(\mathbb{E}\, A^{(r)} - [\![\mathbb{E}\, A^{(r)}]\!]_r\big)}{\delta\,\mathrm{tr}\, A} + r\log\left(\frac{\mathbb{E}\,\mathrm{tr}\, A^{(r)}}{\delta\,\mathrm{tr}\, A}\right).$$

This bound can be simplified as follows. Observe that the (random) residual matrix $A^{(r)}$ is a Schur complement of A, so it satisfies $\mathbb{E}\, A^{(r)} \preceq A$. By the Ky Fan variational principle [36, Thm. 8.17], the best rank-r approximation error in the trace norm is monotone with respect to the psd order, so

$$\mathrm{tr}\big(\mathbb{E}\, A^{(r)} - [\![\mathbb{E}\, A^{(r)}]\!]_r\big) \leq \mathrm{tr}(A - [\![A]\!]_r).$$

Additionally, Lemma 3.4) guarantees that $\mathbb{E}\,\mathrm{tr}\, A^{(r)} \leq 2^r\,\mathrm{tr}(A - [\![A]\!]_r)$. Using these facts, we can simplify (3.7)–(3.8) to show that

$$\mathbb{E}\,\mathrm{tr}\big(A - \widehat{A}^{(k)}\big) \leq \mathrm{tr}(A - [\![A]\!]_r) + \delta\,\mathrm{tr}\, A \quad \text{when} \quad k \geq r + \frac{r\eta}{\delta} + r\log\left(\frac{2^r\eta}{\delta}\right),$$

This completes the proof of the second bound. ∎

3.2. Proof of Lemma 3.2. Let A, H be psd matrices, and recall the definition (3.4) of the expected residual map $\mathbf{\Phi}$. First, to prove that $\mathbf{\Phi}$ is positive, note that

$$(3.9) \qquad \mathbf{\Phi}(H) = \left(I - \frac{H}{\operatorname{tr} H}\right) H \succeq \mathbf{0}.$$

Next, to establish concavity, we choose $\theta \in [0,1]$ and make the calculation

$$\mathbf{\Phi}(\theta A + (1-\theta)H) - \theta\mathbf{\Phi}(A) - (1-\theta)\mathbf{\Phi}(H) = \frac{\theta(1-\theta)}{\theta \operatorname{tr} A + (1-\theta)\operatorname{tr} H}\left(\sqrt{\frac{\operatorname{tr} H}{\operatorname{tr} A}}A - \sqrt{\frac{\operatorname{tr} A}{\operatorname{tr} H}}H\right)^2 \succeq \mathbf{0}.$$

Last, to establish monotonicity, observe that $\mathbf{\Phi}$ is positive homogeneous; that is, $\mathbf{\Phi}(\tau A) = \tau\mathbf{\Phi}(A)$ for $\tau \geq 0$. Invoking the concavity property (3.6) with $\theta = 1/2$,

$$\mathbf{\Phi}(A + H) = 2\,\mathbf{\Phi}\left(\frac{A+H}{2}\right) \succeq \mathbf{\Phi}(A) + \mathbf{\Phi}(H) \succeq \mathbf{\Phi}(A).$$

We have used the positivity of $\mathbf{\Phi}(H)$ in the last step. ∎

3.3. Proof of Lemma 3.3. First, we show that it suffices to consider the case of a diagonal matrix. Let H be an $N \times N$ psd matrix with eigendecomposition $H = V\Lambda V^*$. The definition (3.4) of the expected residual map implies that

$$\mathbf{\Phi}(H) = V\mathbf{\Phi}(\Lambda)V^*.$$

By iteration, the same relation holds with $\mathbf{\Phi}^{\circ k}$ in place of $\mathbf{\Phi}$. In particular, $\operatorname{tr}\mathbf{\Phi}^{\circ k}(H) = \operatorname{tr}\mathbf{\Phi}^{\circ k}(\Lambda)$. Therefore, we may restrict our attention to the diagonal case where $H = \Lambda$.

Second, we obtain an upper bound on $\operatorname{tr}\mathbf{\Phi}^{\circ k}(\Lambda)$. Because the map $\Lambda \mapsto \operatorname{tr}\mathbf{\Phi}^{\circ k}(\Lambda)$ is concave and invariant under permutations of its arguments, averaging together some of the eigenvalues $\lambda_{11}, \ldots, \lambda_{NN}$ can only increase the value of $\operatorname{tr}\mathbf{\Phi}^{\circ k}(\Lambda)$. Therefore, by introducing the function

$$f_k(a, b, r, q) := \operatorname{tr}\mathbf{\Phi}^{\circ k}\Big(\operatorname{diag}\Big(\underbrace{\frac{a}{r}, \ldots, \frac{a}{r}}_{r \text{ times}}, \underbrace{\frac{b}{q}, \ldots, \frac{b}{q}}_{q \text{ times}}\Big)\Big),$$

we obtain the upper bound

$$(3.10) \qquad \operatorname{tr}\mathbf{\Phi}^{\circ k}(\Lambda) \leq f_k(\operatorname{tr}[\![\Lambda]\!]_r, \operatorname{tr}(\Lambda - [\![\Lambda]\!]_r), r, N - r).$$

For further reference, we note that $f_k(a, b, r, q)$ is weakly increasing as a function of q. Indeed, for every tuple (a, b, r, q),

$$f_k(a, b, r, q) = \operatorname{tr}\mathbf{\Phi}^{\circ k}\Big(\operatorname{diag}\Big(\underbrace{\frac{a}{r}, \ldots, \frac{a}{r}}_{r \text{ times}}, \underbrace{\frac{b}{q}, \ldots, \frac{b}{q}}_{q \text{ times}}\Big)\Big) = \operatorname{tr}\mathbf{\Phi}^{\circ k}\Big(\operatorname{diag}\Big(\underbrace{\frac{a}{r}, \ldots, \frac{a}{r}}_{r \text{ times}}, \underbrace{\frac{b}{q}, \ldots, \frac{b}{q}}_{q \text{ times}}, 0\Big)\Big)$$

$$\leq \operatorname{tr}\mathbf{\Phi}^{\circ k}\Big(\operatorname{diag}\Big(\underbrace{\frac{a}{r}, \ldots, \frac{a}{r}}_{r \text{ times}}, \underbrace{\frac{b}{q+1}, \ldots, \frac{b}{q+1}}_{q+1 \text{ times}}\Big)\Big) = f_k(a, b, r, q+1).$$

We have exploited the fact that $\boldsymbol{\Phi}$ is defined for matrices of every dimension.

Next, we derive a worst-case expression for the error $f_k(a, b, r, q)$. For each $k = 0, 1, 2, \ldots$, define the nonnegative quantities $a^{(k)}$ and $b^{(k)}$ via the relation

$$\boldsymbol{\Phi}^{\circ k}\left(\operatorname{diag}\left(\underbrace{\frac{a}{r}, \ldots, \frac{a}{r}}_{r \text{ times}}, \underbrace{\frac{b}{q}, \ldots, \frac{b}{q}}_{q \text{ times}}\right)\right) =: \operatorname{diag}\left(\underbrace{\frac{a^{(k)}}{r}, \ldots, \frac{a^{(k)}}{r}}_{r \text{ times}}, \underbrace{\frac{b^{(k)}}{q}, \ldots, \frac{b^{(k)}}{q}}_{q \text{ times}}\right).$$

By the definition (3.4) of the expected residual map $\boldsymbol{\Phi}$, the quantities $a^{(k)}$ and $b^{(k)}$ satisfy the recurrence relations

$$a^{(k)} - a^{(k-1)} = -\frac{\left(a^{(k-1)}\right)^2}{r\left(a^{(k-1)} + b^{(k-1)}\right)} \quad \text{and} \quad b^{(k)} - b^{(k-1)} = -\frac{\left(b^{(k-1)}\right)^2}{q\left(a^{(k-1)} + b^{(k-1)}\right)}$$

with initial conditions $a^{(0)} = a$ and $b^{(0)} = b$. This construction guarantees $f_k(a, b, r, q) = a^{(k)} + b^{(k)}$. Additionally, the quantities $a^{(k)}$ and $b^{(k)}$ depend continuously on q, and they converge as $q \to \infty$ to limiting values $\overline{a}^{(k)}$ and $\overline{b}^{(k)} \equiv b$, where the sequence $\overline{a}^{(k)}$ satisfies

$$\overline{a}^{(k)} - \overline{a}^{(k-1)} = -\frac{\left(\overline{a}^{(k-1)}\right)^2}{r\left(\overline{a}^{(k-1)} + b\right)} \quad \text{with initial condition } \overline{a}^{(0)} = a.$$

It follows that

$$f_k(a, b, r, q) \leq \overline{a}^{(k)} + b \leq a + b.$$

We have used the facts that $f_k(a, b, r, q)$ is increasing in q and $\overline{a}^{(k)}$ is decreasing in k.

All that remains is to determine how quickly $\overline{a}^{(k)}$ decreases as a function of k. To that end, we pass from discrete time to continuous time. At each instant $t = 0, 1, 2, \ldots$, the discrete-time process $\overline{a}^{(t)}$ is bounded from above by the continuous-time process $x(t)$ satisfying the ODE

$$\frac{\mathrm{d}}{\mathrm{d}t} x(t) = -\frac{x(t)^2}{r(x(t) + b)} \quad \text{with initial condition } x(0) = a.$$

The comparison between discrete- and continuous-time processes holds because $x \mapsto -x^2/(rx + rb)$ is decreasing over the range $x \in (0, \infty)$. Next, assuming $\Delta(a + b) \leq a$, we can use separation of variables to solve for the time $t_\star$ at which $x(t_\star) = \Delta(a + b)$:

$$t_\star = \int_{x=a}^{\Delta(a+b)} -\frac{r(x+b)}{x^2} \, \mathrm{d}x = rb\left(\frac{1}{\Delta(a+b)} - \frac{1}{a}\right) + r\log\left(\frac{a}{\Delta(a+b)}\right) \leq \frac{rb}{\Delta(a+b)} + r\log\left(\frac{1}{\Delta}\right).$$

It follows (even for $\Delta(a + b) > a$) that

$$\overline{a}^{(k)} \leq \Delta(a + b) \quad \text{when} \quad k \geq \frac{rb}{\Delta(a+b)} + r\log_+\left(\frac{1}{\Delta}\right).$$

To conclude, substitute $a = \operatorname{tr}[\![\boldsymbol{\Lambda}]\!]_r$ and $b = \operatorname{tr}(\boldsymbol{\Lambda} - [\![\boldsymbol{\Lambda}]\!]_r)$ and combine with (3.10). ∎

3.4. Proof of Lemma 3.4. Let P denote the orthogonal projection onto the k dominant eigenvectors of A and set $P_\perp := I - P$. Consider a hierarchical sampling procedure for the pivot index s_1:

- With probability $\operatorname{tr}(AP)/\operatorname{tr}A$, the "good" event G occurs and the pivot index is chosen as

$$\mathbb{P}(s_1 = i \mid \mathsf{G}) = \frac{(AP)_{ii}}{\operatorname{tr}(AP)}.$$

- With probability $\operatorname{tr}(AP_\perp)/\operatorname{tr}A$, the "bad" event $\mathsf{B} = \mathsf{G}^c$ occurs and the pivot is selected as

$$\mathbb{P}(s_1 = i \mid \mathsf{B}) = \frac{(AP_\perp)_{ii}}{\operatorname{tr}(AP_\perp)}.$$

It suffices to analyze this two-step sampling procedure. Indeed, a short calculation confirms that this procedure leads to the same sampling distribution for s_1 as in RPCHOLESKY.

On the bad event B, apply the crude bound

$$(3.11) \qquad \operatorname{tr}\big(A^{(1)} - [\![A^{(1)}]\!]_{k-1}\big) = \sum_{j=k}^{N} \lambda_j(A^{(1)}) \le \lambda_k(A) + \operatorname{tr}(A - [\![A]\!]_k).$$

This relation holds because $A^{(1)} \preceq A$, and the Weyl monotonocity principle [36, Thm. 8.11] yields $\lambda_j(A_1) \le \lambda_j(A)$ for $j = k, \dots, N$.

On the good event G, apply the Ky Fan variational principle [36, Thm. 8.17] to write

$$\operatorname{tr}\big(A^{(1)} - [\![A^{(1)}]\!]_{k-1}\big) = \min\bigg\{ \sum_{j=k}^{N} u_j^* A^{(1)} u_j : u_k, \dots, u_N \text{ orthonormal} \bigg\}.$$

Choose the vectors $u_{k+1}, \dots, u_N$ to be unit-norm eigenvectors $v_{k+1}(A), \dots, v_N(A)$ associated with the smallest eigenvalues. Set $u_k := Pe_{s_1}/\|Pe_{s_1}\|$, which is orthonormal to the other vectors by the choice of P. This gives the bound

$$(3.12) \quad \operatorname{tr}\big(A^{(1)} - [\![A^{(1)}]\!]_{k-1}\big) \le \sum_{j=k+1}^{N} v_j(A)^* A^{(1)} v_j(A) + \frac{e_{s_1}^* P A^{(1)} P e_{s_1}}{e_{s_1}^* P e_{s_1}} \le \operatorname{tr}(A - [\![A]\!]_k) + \frac{(PA^{(1)}P)_{s_1 s_1}}{p_{s_1 s_1}}.$$

Using the fact that $A^{(1)} = A - Ae_{s_1}e_{s_1}^* A / a_{s_1 s_1}$, it follows

$$(PA^{(1)}P)_{s_1 s_1} = (AP)_{s_1 s_1} - \frac{((AP)_{s_1 s_1})^2}{a_{s_1 s_1}} = \frac{(AP)_{s_1 s_1}(AP_\perp)_{s_1 s_1}}{a_{s_1 s_1}}.$$

Therefore, taking the conditional expectation of both sides of (3.12),

$$(3.13) \qquad \begin{aligned} \mathbb{E}\big[\operatorname{tr}\big(A^{(1)} - [\![A^{(1)}]\!]_{k-1}\big) \,\big|\, \mathsf{G}\big] &\le \operatorname{tr}(A - [\![A]\!]_k) + \sum_{i=1}^{N} \frac{(AP)_{ii}}{\operatorname{tr}(AP)} \cdot \frac{(AP)_{ii}(AP_\perp)_{ii}}{a_{ii} p_{ii}} \\ &\le \left(1 + \frac{\lambda_1(A)}{\operatorname{tr}(AP)}\right) \operatorname{tr}(A - [\![A]\!]_k). \end{aligned}$$

The last line uses the inequalities $(AP)_{ii} \le a_{ii}$ and $(AP)_{ii} \le \lambda_1(A) p_{ii}$.

Finally, combine the bound (3.11) for the bad event and the bound (3.13) for the good event to verify that

$$\mathbb{E}\,\mathrm{tr}\big(\boldsymbol{A}^{(1)} - [\![\boldsymbol{A}^{(1)}]\!]_{k-1}\big) = \mathbb{E}\big[\mathrm{tr}\big(\boldsymbol{A}^{(1)} - [\![\boldsymbol{A}^{(1)}]\!]_{k-1}\big)\,\big|\,\mathsf{G}\big] \cdot \mathbb{P}(\mathsf{G}) + \mathbb{E}\big[\mathrm{tr}\big(\boldsymbol{A}^{(1)} - [\![\boldsymbol{A}^{(1)}]\!]_{k-1}\big)\,\big|\,\mathsf{B}\big] \cdot \mathbb{P}(\mathsf{B})$$

$$\leq \left(1 + \frac{\lambda_1(\boldsymbol{A}) + \lambda_k(\boldsymbol{A})}{\mathrm{tr}\,\boldsymbol{A}}\right)\mathrm{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_k).$$

Since $\lambda_1(\boldsymbol{A}) + \lambda_k(\boldsymbol{A}) \leq \mathrm{tr}\,\boldsymbol{A}$, this bound can be weakened to give

$$\mathbb{E}\,\mathrm{tr}\big(\boldsymbol{A}^{(1)} - [\![\boldsymbol{A}^{(1)}]\!]_{k-1}\big) \leq 2\,\mathrm{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_k).$$

By iterating, we conclude that

$$\mathbb{E}\,\mathrm{tr}\,\boldsymbol{A}^{(k)} \leq 2\,\mathbb{E}\,\mathrm{tr}\big(\boldsymbol{A}^{(k-1)} - [\![\boldsymbol{A}^{(k-1)}]\!]_1\big) \leq 4\,\mathbb{E}\,\mathrm{tr}\big(\boldsymbol{A}^{(k-2)} - [\![\boldsymbol{A}^{(k-2)}]\!]_2\big) \leq \cdots \leq 2^k\,\mathrm{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_k).$$

This estimate is the statement of the doubling bound. ∎

4. Comparison with other methods. This section compares RPCHOLESKY with other psd low-rank approximation methods. We cover the greedy method (subsection 4.1), uniform sampling (subsection 4.2), DPP sampling (subsection 4.3), RLS sampling (subsection 4.4), and alternative algorithms that are not based on the column Nyström approximation (subsection 4.5). For brevity, the proofs are deferred to Appendix A.

4.1. The greedy method. The *greedy method* (1.2) is a column Nyström approximation with a long history in numerical analysis [16, §10.3] under the name *complete pivoting* or *diagonal pivoting*. Fine and Scheinberg [9] later popularized the method's use for kernel computations. Despite its popularity, the greedy method is known to fail dramatically when applied to some worst-case matrices $\boldsymbol{A}$, as discussed by Higham [16, §10.3.1]. Further supporting this conclusion, the following theorem analyzes the large number of columns needed for the greedy method to provide a (r, ε)-approximation.

> **Theorem 4.1 (Greedy method).** *The greedy method has the following error properties:*
> *(a) If $k \geq (1 - (1 + \varepsilon)\eta)N$, the Nyström approximation $\widehat{\boldsymbol{A}}^{(k)}$ produced by the greedy method is a (r, ε)-approximation (1.4).*
> *(b) For any $N \geq r \geq 1$, $\varepsilon > 0$, and $\eta \in [0, 1]$, there exists a psd matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ such that $\mathrm{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r) = \eta\,\mathrm{tr}\,\boldsymbol{A}$ and $\widehat{\boldsymbol{A}}^{(k)}$ produced by the greedy method is not a (r, ε)-approximation for $k \leq \min\{\eta N, (1 - (1 + \varepsilon)\eta)N\} - 1$.*

> *Proof.* See Appendix A. ∎

Theorem 4.1 shows that the greedy method requires accessing $\Omega(N)$ columns to make progress in approximating a worst-case matrix $\boldsymbol{A}$. This potential for slow convergence is a major concern in applications, and subsections 1.2 and 2.2 provide examples of kernel matrices for which the greedy method produces worse-quality approximations than other Nyström methods. To address the deficiencies of the greedy method, Gu and Miranian [13] introduced a *strong rank-revealing* pivoting strategy which produces higher-quality approximations on worst-case examples. However, this approach is sufficiently complicated that it has not seen widespread use.

4.2. Uniform sampling. Another popular column Nyström approximation scheme selects pivots *uniformly at random,* either with or without replacement. Uniform sampling without replacement leads to a larger number of unique columns selections, hence, a higher-quality Nyström approximation [17]. Uniform sampling can be effective when the dominant r eigenvectors have mass that is spread out equally over all the coordinates (a property known as "incoherence") [12]. However, uniform sampling fails when the dominant eigenvectors are heavily concentrated on just a few coordinates. This potential for bad behavior is reflected in the following worst-case error bounds:

Theorem 4.2 (Uniform sampling). *Suppose A is a psd matrix with ones on the diagonal, and the pivots are sampled uniformly at random either with or without replacement.*

(a) *If $k \geq \frac{r-1}{\varepsilon\eta} + \frac{1}{\varepsilon}$, the Nyström approximation $\widehat{A}^{(k)}$ produced by uniform sampling is a (r,ε)-approximation.*

(b) *For any $r \geq 2$ and any $\varepsilon > 0$ and $\eta > 0$ satisfying $\sqrt{\frac{\varepsilon}{\varepsilon+1}} + \eta < 1$, there exists a psd matrix A such that $\mathrm{tr}(A - [\![A]\!]_r) = \eta\,\mathrm{tr}\,A$ and $\widehat{A}^{(k)}$ produced by uniform sampling is not a (r,ε)-approximation for $k < \frac{r-1}{\eta}(\sqrt{\varepsilon^{-1}+1} - 1)^2$.*

Proof. See Appendix A. ∎

Although uniform sampling is typically applied to a kernel matrix A with ones on the diagonal, a more general *diagonal sampling* [10] method randomly selects the pivots with probabilities proportional to diag A. Diagonal sampling attains exactly the same error bounds as uniform sampling, as seen from the proof of Theorem 4.2. RPCHOLESKY is equivalent to performing diagonal sampling iteratively on the residual matrix, and it leads to significantly stronger error bounds.

4.3. Determinantal point process sampling. Determinantal point process (DPP) sampling [7] is a column Nyström approximation method that randomly selects a pivot set $S \subseteq \{1,\dots,N\}$ with cardinality $|S| = k$ according to the distribution

$$(4.1) \qquad \mathbb{P}\{\{s_1,\dots,s_k\} = S\} = \frac{\det A(S,S)}{\sum_{|S'|=k} \det A(S',S')}.$$

DPP sampling satisfies the following error bounds (see [3, Thm. 1], [14]):

Theorem 4.3 (k-DPP sampling). *The Nyström approximation produced by k-DPP sampling is nearly optimal in the following sense:*

(a) *If $k \geq r/\varepsilon + r - 1$, the approximation $\widehat{A}^{(k)}$ produced by a k-DPP is an (r,ε)-approximation.*

(b) *For any $r \geq 1$ and $\varepsilon > 0$, there exists a psd matrix A such that $\widehat{A}^{(k)}$ produced by a k-DPP is not an (r,ε)-approximation for $k < r/\varepsilon + r - 1$.*

(c) *For any $r \geq 1$ and $\varepsilon > 0$, there exists a matrix A such that no rank-k column Nyström approximation is an (r,ε)-approximation for $k < r/\varepsilon$.*

Proof. Parts (a) and (b) were essentially proved in [3] and [14]. For part (c), see Appendix A. ∎

The theory above demonstrates that DPP sampling has nearly optimal theoretical properties, but implementing DPP sampling for large k values remains expensive relative to peer methods [2, 6]. RPCHOLESKY is much cheaper to perform and is identical to performing 1-DPP sampling iteratively on the residual matrix.

4.4. Ridge leverage score sampling. Like DPP sampling, ridge leverage score (RLS) sampling [1, 4, 20, 26] is a Nyström approximation defined by a distribution on the pivot set S. However, the sampling distribution is potentially more tractable than in DPP sampling since it only depends on a single matrix inverse rather than numerous determinants of $k \times k$ submatrices. For any $\lambda > 0$, the vector of ridge leverage scores is defined as

$$\boldsymbol{\ell}^{\lambda} = \text{diag}(\boldsymbol{A}(\boldsymbol{A} + \lambda \mathbf{I})^{-1}),$$

and the vector of sampling probabilities is defined by [20] to be

$$\boldsymbol{p} = \min\left\{1, 16\,\boldsymbol{\ell}^{\lambda} \log\left(\frac{1}{\delta} \sum_{j=1}^{N} \ell_{j}^{\lambda}\right)\right\}$$

where $\delta \in (0, 1/8)$ is a parameter called the "failure probability". The coordinate set $\mathsf{S} \subseteq \{1, \ldots, N\}$ is generated by independently including each index i with probability $\boldsymbol{p}_i$. This form of RLS sampling has been analyzed by [20, Thm. 3], and their main result can be summarized as follows:

Theorem 4.4 (Ridge leverage score sampling). *With probability at least $1 - \delta$, the Nyström approximation $\widehat{\boldsymbol{A}}$ produced by RLS sampling satisfies*

$$\|\boldsymbol{A} - \widehat{\boldsymbol{A}}\| \le \lambda$$

and

$$\text{rank}\,\widehat{\boldsymbol{A}} \le 32\left(r + \frac{\text{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r)}{\lambda}\right) \log\left(\frac{r}{\delta} + \frac{\text{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r)}{\delta\lambda}\right).$$

Theorem 4.4 is different from the other error bounds presented in this work since it guarantees control in the spectral norm, rather than the trace norm, and it introduces a random approximation rank and a failure probability δ not seen in the other bounds. Due to these differences, the following calculation is needed to convert Theorem 4.4 into a bound in the expected trace norm:

Corollary 4.5 (Ridge leverage score (r, ε)-approximation). *For any $\varepsilon < \frac{1}{4}$, let $\widehat{\boldsymbol{A}}$ be the Nyström approximation obtained from RLS sampling with*

$$\lambda = \frac{\varepsilon}{2r}\,\text{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r), \qquad \delta = \frac{\varepsilon\eta}{2},$$

and set

$$\widehat{\boldsymbol{A}}^{(k)} = \begin{cases} \widehat{\boldsymbol{A}}, & \text{rank}\,\widehat{\boldsymbol{A}} \le k, \\ \boldsymbol{0}, & \text{rank}\,\widehat{\boldsymbol{A}} > k. \end{cases}$$

Then, $\widehat{\boldsymbol{A}}^{(k)}$ is an (r, ε)-approximation if $k \ge 136\frac{r}{\varepsilon}\log\left(\frac{3r}{\varepsilon\eta}\right)$.

Proof. See Appendix A. ∎

Corollary 4.5 guarantees better worst-case performance for RLS sampling than for the greedy method or uniform sampling, at least when N and η^{-1} are very large. However, the available implementations of RLS sampling [4, 20, 26] are fairly complicated, as they require selecting parameters

λ and δ and approximating the resulting RLS sampling distribution. Because of these preprocessing steps, RLS sampling requires a higher number of entry evaluations than RPCHOLESKY for a fixed approximation rank k.

4.5. Algorithms that are not based on column Nyström approximation. In addition to column Nyström approximation, there are two other popular approaches for randomized low-rank approximation of psd matrices. First, *random embeddings* can produce a high-quality approximation (even higher quality than RPCHOLESKY) if it is possible to compute matrix–vector products with the input matrix. Yet, these methods may be too expensive to deploy in the entry-wise access model. Random embedding methods include the single-pass Nyström method [31], randomized subspace iteration [15], and randomized block Krylov iteration [30].

Second, *random features* (RF) methods can be used to approximate kernel matrices for many well-known kernel functions K [19, 23]. RF methods do not require any entry evaluations, and the computations are highly parallelizable. However, due to the dependence on Monte Carlo sampling, the accuracy is limited by the $1/\sqrt{k}$ convergence rate from the central limit theorem, regardless of the decay of the eigenvalues in the input matrix [35]. In contrast, RPCHOLESKY takes advantage of the eigenvalue decay, so it can attain much higher accuracy.

5. Conclusion. This work has demonstrated the power of RPCHOLESKY for the low-rank approximation of a positive semidefinite kernel matrix $A \in \mathbb{R}^{N \times N}$. RPCHOLESKY can be programmed in just a few lines of code, and it accelerates many kernel algorithms such as kernel ridge regression and spectral clustering. RPCHOLESKY reduces the computational cost of these algorithms from $\mathcal{O}(N^3)$ operations to just $\mathcal{O}(k^2 N)$ operations, where the approximation rank k can be much smaller than the matrix dimension N.

Numerical experiments suggest that RPCHOLESKY dominates other column Nyström approximation algorithms in terms of efficiency and memory footprint. Given a fixed approximation rank k, RPCHOLESKY requires a very small number of entry evaluations, just $(k+1)N$. RPCHOLESKY typically produces approximations that match or improve on the greedy method, uniform sampling, RLS sampling, and DPP sampling. Moreover, theoretical error bounds guarantee that RPCHOLESKY converges nearly as fast as possible in the expected trace norm.

Taken as a whole, this work paves the way for greater use of RPCHOLESKY. The method can potentially be pushed even further, for example, through combinations with other accelerated methods for prediction, clustering, and other learning tasks. The future is indeed bright for this simple yet surprisingly effective algorithm.

Appendix A. More proofs. In this section, we prove error bounds for the greedy method (Theorem 4.1), uniform sampling (Theorem 4.2), DPP sampling (Theorem 4.3(c)), and RLS sampling (Corollary 4.5).

Proof of Theorem 4.1. At each iteration $1 \le i \le k$, there are at most $N - i + 1$ nonzero entries in the diagonal of the residual matrix $A^{(i-1)}$, and the largest entry is incorporated into the pivot set S.

DRAFT

Consequently,

$$\operatorname{tr} \boldsymbol{A}^{(i)} \le \left(1 - \frac{1}{N-i+1}\right) \operatorname{tr} \boldsymbol{A}^{(i-1)} = \frac{N-i}{N-i+1} \operatorname{tr} \boldsymbol{A}^{(i-1)}.$$

By induction, it follows that $\operatorname{tr} \boldsymbol{A}^{(k)} \le \frac{N-k}{N} \operatorname{tr} \boldsymbol{A}$ and

$$\frac{\operatorname{tr} \boldsymbol{A}^{(k)}}{\operatorname{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r)} \le \left(1 - \frac{k}{N}\right) \frac{\operatorname{tr} \boldsymbol{A}}{\operatorname{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r)} = \frac{1 - \frac{k}{N}}{\eta}.$$

If $k \ge (1 - (1+\varepsilon)\eta)N$, the right-hand side is bounded by $1 + \varepsilon$, establishing part (a).

Next, let M be the smallest integer strictly greater than $(1-\eta)N$. Set

$$\boldsymbol{A} = \begin{pmatrix} (1+\delta)\mathbf{I}_{N-M} & \\ & \boldsymbol{B}_M \end{pmatrix},$$

where $\mathbf{I}_{N-M}$ is the $(N-M) \times (N-M)$ identity matrix and $\boldsymbol{B}_M$ is the $M \times M$ matrix of all ones. Set

$$\delta = \frac{M - (1-\eta)N}{(1-\eta)(N-M)}.$$

so that the $\boldsymbol{B}_M$ block is responsible for a $1 - \eta$ fraction of the trace. Because $1 + \delta > 1$, the greedy method chooses entries from the $(1+\delta)\mathbf{I}_{N-M}$ block before the $\boldsymbol{B}_M$ block. An explicit calculation shows

$$\frac{\operatorname{tr}(\boldsymbol{A} - \widehat{\boldsymbol{A}}^{(k)})}{\operatorname{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_1)} = \frac{1}{\eta} - \frac{k}{N-M} > 1 + \varepsilon$$

for each $k \le \min\{\eta N, (1 - (1+\varepsilon)\eta)N\} - 1$. This establishes part (b) for any $r \ge 1$. ∎

Proof of Theorem 4.2. Part (a) is a modification of the error bound [10, Eq. (4)], and it is proved using similar techniques. Assume the sampling is conducted with replacement, because sampling with replacement leads to lower error. The trace-norm error takes the form

$$\operatorname{tr}\left(\boldsymbol{A} - \boldsymbol{A}(:,\mathsf{S})\boldsymbol{A}(\mathsf{S},\mathsf{S})^{\dagger}\boldsymbol{A}(\mathsf{S},:)\right) = \operatorname{tr}\left(\boldsymbol{A}^{1/2}\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{A}^{1/2}\right),$$

where $\boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}$ is the orthogonal projector onto the range of $\boldsymbol{A}^{1/2}(:,\mathsf{S})$. Let $\boldsymbol{v}_i$ denote the ith eigenvector of $\boldsymbol{A}$, and calculate

$$(\text{A.1}) \qquad \operatorname{tr}\left(\boldsymbol{A}^{1/2}\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{A}^{1/2}\right) = \operatorname{tr}\left(\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{A}\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\right)$$

$$(\text{A.2}) \qquad = \sum_{i=1}^{N} \boldsymbol{v}_i^*\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{A}\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{v}_i$$

$$(\text{A.3}) \qquad \le \sum_{i=1}^{r} \boldsymbol{v}_i^*\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{A}\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{v}_i + \sum_{i=r+1}^{N} \boldsymbol{v}_i^* \boldsymbol{A} \boldsymbol{v}_i$$

$$(\text{A.4}) \qquad = \sum_{i=1}^{r} \left\| \boldsymbol{A}^{1/2}\left(\mathbf{I} - \boldsymbol{P}_{\boldsymbol{A}^{1/2}(:,\mathsf{S})}\right)\boldsymbol{v}_i \right\|^2 + \operatorname{tr}\left(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r\right),$$

where (A.1) uses the idempotency of $I - P_{A^{1/2}(:,S)}$ and the cyclicity of the trace, (A.2)–(A.3) uses the psd ordering $(I - P_{A^{1/2}(:,S)})A(I - P_{A^{1/2}(:,S)}) \preceq A$, and (A.3)–(A.4) uses the characterization of $\mathrm{tr}(A - [\![A]\!]_r)$ in terms of eigenvectors of A.

Next, consider the error term $\|A^{1/2}(I - P_{A^{1/2}(:,S)})v_i\|^2$. Since $A^{1/2}P_{A^{1/2}(:,S)}$ is the optimal approximation to $A^{1/2}$ in the row space of $A^{1/2}(S,:)$, the matrix $A^{1/2}P_{A^{1/2}(:,S)}$ can be replaced with any other matrix having the same row space and the error term will only increase or stay the same. In particular,

$$\|A^{1/2}(I - P_{A^{1/2}(:,S)})v_i\|^2 \leq \left\| A^{1/2}\left(I - \frac{1}{k}\sum_{j=1}^{k}\frac{e_{s_j}e_{s_j}^*}{a_{s_j s_j}}\right)v_i \right\|^2,$$

where $\mathbf{e}_{s_i}$ is the unit vector in the direction of the ith random pivot. Observe that the random vectors

$$A^{1/2}\frac{\mathbf{e}_{s_i}\mathbf{e}_{s_i}^*}{a_{s_i s_i}}v_i, \quad i = 1,\ldots,k$$

are independent, they have mean $A^{1/2}v_i$, and the trace of their covariance matrix is $\mathrm{tr}\,A - \lambda_i(A)$. Hence,

$$\mathbb{E}\left\| A^{1/2}\left(I - \frac{1}{k}\sum_{j=1}^{k}\frac{\mathbf{e}_{s_j}\mathbf{e}_{s_j}^*}{a_{s_j s_j}}\right)v_i \right\|^2 = \frac{\mathrm{tr}\,A - \lambda_i(A)}{k}.$$

Last, summing over $i = 1,\ldots,r$ guarantees the error bound

$$\mathbb{E}\,\mathrm{tr}(A - \widehat{A}^{(k)}) \leq \frac{r-1}{k}\mathrm{tr}\,A + \left(1 + \frac{1}{k}\right)\mathrm{tr}(A - [\![A]\!]_r),$$

which completes part (a) of the theorem.

To prove part (b), consider the $N \times N$ matrix

$$A = \begin{pmatrix} B_{N-M(r-1)} & & & \\ & C_{M,\delta} & & \\ & & \ddots & \\ & & & C_{M,\delta} \end{pmatrix},$$

where $B_{N-M(r-1)}$ is the $(N - M(r-1)) \times (N - M(r-1))$ matrix of all ones and $C_{M,\delta}$ is the $M \times M$ matrix with entries

$$C_{M,\delta} = \begin{pmatrix} 1 & \delta & \cdots & \delta \\ \delta & 1 & \cdots & \delta \\ \vdots & \vdots & \ddots & \vdots \\ \delta & \delta & \cdots & 1 \end{pmatrix}.$$

Set

$$N = \left\lceil \left(1 - \sqrt{\frac{\varepsilon}{1+\varepsilon}}\right)\frac{(r-1)(M-1)}{\eta} \right\rceil, \qquad \delta = 1 - \frac{\eta N}{(r-1)(M-1)},$$

where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x. For large enough M, this construction guarantees that the top r eigenvalues of A are responsible for a $1 - \eta$ fraction of the trace, and δ lies very close to $\sqrt{\frac{\varepsilon}{1+\varepsilon}}$.

Next, consider the rank-k Nyström approximation $\widehat{A}^{(k)}$ that selects k_1 columns from the first block, k_2 columns from the second block, etc. The Schur complement of $C_{M,\delta}$ with respect to any k_i distinct columns has trace

$$(M - k_i)\left(1 + \frac{1}{\delta^{-1} + k_i - 1}\right)(1 - \delta).$$

Therefore, calculate

$$\frac{\operatorname{tr}(A - \widehat{A}^{(k)})}{\operatorname{tr}(A - [\![A]\!]_r)} \geq \frac{1}{r-1} \sum_{i=2}^{r} \frac{M - k_i}{M - 1}\left(1 + \frac{1}{\delta^{-1} + k_i - 1}\right).$$

Using the convexity of $f(x) = \frac{1}{x}$ and the fact that $\mathbb{E}\, k_i = \frac{Mk}{N}$ for $2 \leq i \leq r$, calculate

(A.5)
$$\frac{\mathbb{E}\operatorname{tr}(A - \widehat{A}^{(k)})}{\operatorname{tr}(A - [\![A]\!]_r)} \geq \frac{M - k}{M - 1}\left(1 + \frac{1}{\delta^{-1} + \frac{Mk}{N} - 1}\right).$$

For large enough M, (A.5) guarantees

$$\frac{\mathbb{E}\operatorname{tr}(A - \widehat{A}^{(k)})}{\operatorname{tr}(A - [\![A]\!]_r)} > 1 + \varepsilon$$

for each $k < \frac{r-1}{\eta}(\sqrt{\varepsilon^{-1} + 1} - 1)^2$. ∎

Proof of Theorem 4.3 *(c).* The proof uses the same construction given in [14, Lem. 6.2]. Consider the $Mr \times Mr$ marix

$$A = \begin{pmatrix} C_{M,\delta} & & & \\ & C_{M,\delta} & & \\ & & \ddots & \\ & & & C_{M,\delta} \end{pmatrix},$$

where $C_{M,\delta}$ is the $M \times M$ matrix with entries

$$C_{M,\delta} = \begin{pmatrix} 1 & \delta & \cdots & \delta \\ \delta & 1 & & \vdots \\ \vdots & & \ddots & \\ \delta & \cdots & & 1 \end{pmatrix},$$

and consider the rank-k Nyström approximation $\widehat{A}^{(k)}$ that selects k_1 columns from the first block, k_2 columns from the second block, etc. Similar to the proof of Theorem 4.2, calculate

$$\frac{\operatorname{tr}(A - \widehat{A}^{(k)})}{\operatorname{tr}(A - [\![A]\!]_r)} = \frac{1}{r} \sum_{i=1}^{r} \frac{M - k_i}{M - 1}\left(1 + \frac{1}{\delta^{-1} + k_i - 1}\right) \geq \frac{M - k}{M - 1}\left(1 + \frac{1}{\delta^{-1} + k/r - 1}\right)$$

<div align="center">DRAFT</div>

Choose M large enough and δ close enough to 1 so that

$$\frac{\mathrm{tr}(\boldsymbol{A} - \widehat{\boldsymbol{A}}^{(k)})}{\mathrm{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r)} > 1 + \varepsilon$$

for each $k < \frac{r}{\varepsilon}$.   ∎

Proof of Corollary 4.5. Plugging in the chosen values for λ and δ, Theorem 4.4 guarantees with probability at least $1 - \frac{\varepsilon\eta}{2}$,

(A.6) $$\|\boldsymbol{A} - \widehat{\boldsymbol{A}}\| \leq \frac{\varepsilon}{2r} \, \mathrm{tr}(\boldsymbol{A} - [\![\boldsymbol{A}]\!]_r)$$

and

$$\mathrm{rank}\,\widehat{\boldsymbol{A}} \leq 32\left(r + \frac{2r}{\varepsilon}\right) \log\left(\frac{2r}{\varepsilon\eta} + \frac{4r}{\varepsilon^2\eta}\right).$$

Additionally, since $\widehat{\boldsymbol{A}} \succeq \boldsymbol{0}$, the Weyl monotonicity principle [36, Thm. 8.11] guarantees

(A.7) $$\lambda_i(\boldsymbol{A} - \widehat{\boldsymbol{A}}) \leq \lambda_i(\boldsymbol{A}), \qquad 1 \leq i \leq N.$$

(A.6) and (A.7) together imply

$$\mathrm{tr}(\boldsymbol{A} - \widehat{\boldsymbol{A}}) \leq \left(1 + \frac{\varepsilon}{2}\right) \mathrm{tr}\,\boldsymbol{A}$$

With the remaining probability $\frac{\varepsilon\eta}{2}$, the trace-norm error is bounded by

$$\mathrm{tr}(\boldsymbol{A} - \widehat{\boldsymbol{A}}) \leq \mathrm{tr}\,\boldsymbol{A}.$$

Consequently, $\widehat{\boldsymbol{A}}^{(k)}$ is a (r, ε)-approximation for any

(A.8) $$k \geq 32\left(r + \frac{2r}{\varepsilon}\right) \log\left(\frac{2r}{\varepsilon\eta} + \frac{4r}{\varepsilon^2\eta}\right).$$

To complete the proof, bound the right-hand side of (A.8) using $136\frac{r}{\varepsilon} \log\left(\frac{3r}{\varepsilon\eta}\right)$.   ∎

Appendix B. Details of numerical experiments.   The numerical experiments in subsection 1.2 and section 2 are implemented in Python, with code available at https://github.com/eepperly/Randomly-Pivoted-Cholesky. Below we provide further implementation details for some of the Nyström methods:

RPCHOLESKY. All RPCHOLESKY results in subsection 1.2 and section 2 use the standard algorithm (Algorithm 2.1). However, the blocked RPCHOLESKY algorithm (Algorithm 2.2) leads to near-identical accuracy when applied to the $10^5 \times 10^5$ kernel matrix from subsection 2.2 with a block size $B = 50$. For this data set, blocked RPCHOLESKY yields a 5× speedup with the ℓ_1 Laplace kernel (2.5) and a 20× speedup after switching to the Gaussian kernel.

DPP sampling. DPP sampling is implemented using the DPPy Python package [11]. Yet, the samplers from this package all produce error messages when tried on certain inputs, particularly

when k is large and A is nearly low-rank. The theoretically fastest `alpha` sampler cannot be used for our numerical comparisons since it gives error messages even when approximating the **Outliers** matrix with $k = 20$. Rather, Figures 2 and 4 are produced using the `vfx` sampler introduced in [6], and Figure 1 is produced using the comparatively slow `GS` sampler which requires a full eigendecomposition of A. In this work, all DPP samplers generate exact samples from the k-DPP distribution (4.1); we did not carefully study inexact samplers based on MCMC [2].

RLS sampling. Several RLS sampling algorithms have been introduced in the literature, including recursive ridge leverage scores (RRLS) [20], SQUEAK [4], and BLESS [26]. The analysis and experiments in this paper are based on the RRLS algorithm [20], which is implemented in the `recursiveNystrom` method [32] for Python. We find little difference in accuracy between the standard and "accelerated" `recursiveNystrom` methods for our experiments, so we always use the accelerated method with the superior runtime.

For comparison, we have also implemented RLS sampling by directly forming the matrix $A(A + \lambda\mathbf{I})^{-1}$ and sampling k indices without replacement using the normalized ridge leverage scores as probabilities. Even with the ridge leverage scores computed exactly in this way, the relative tracenorm error for the **Smile** example with $k = 100$ is still roughly 10^{-2}, five orders of magnitude larger than the error due to RPCHOLESKY. This indicates that the poor performance of RLS on this example is not due to errors in the approximation of the ridge leverage scores.

REFERENCES

[1] A. ALAOUI AND M. W. MAHONEY, *Fast randomized kernel ridge regression with statistical guarantees*, in Advances in Neural Information Processing Systems, vol. 28, 2015, https://proceedings.neurips.cc/paper/2015/file/f3f27a324736617f20abbf2ffd806f6d-Paper.pdf.

[2] N. ANARI, S. OVEIS GHARAN, AND A. REZAEI, *Monte Carlo Markov chain algorithms for sampling strongly Rayleigh distributions and determinantal point processes*, in 29th Annual Conference on Learning Theory, vol. 49 of Proceedings of Machine Learning Research, 2016, pp. 103–115, https://proceedings.mlr.press/v49/anari16.html.

[3] M.-A. BELABBAS AND P. J. WOLFE, *Spectral methods in machine learning and new strategies for very large datasets*, Proceedings of the National Academy of Sciences, 106 (2009), pp. 369–374, https://doi.org/10.1073/pnas.0810600105.

[4] D. CALANDRIELLO, A. LAZARIC, AND M. VALKO, *Distributed adaptive sampling for kernel matrix approximation*, in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, vol. 54 of Proceedings of Machine Learning Research, 2017, pp. 1421–1429, https://proceedings.mlr.press/v54/calandriello17a.html.

[5] E. CARLEN, *Trace inequalities and quantum entropy: An introductory course*, in Entropy and the Quantum, R. Sims and D. Ueltschi, eds., vol. 529 of Contemporary Mathematics, Providence, RI, 2010, American Mathematical Society, pp. 73–140, https://doi.org/http://dx.doi.org/10.1090/conm/529.

[6] M. DEREZINSKI, D. CALANDRIELLO, AND M. VALKO, *Exact sampling of determinantal point processes with sublinear time preprocessing*, in Advances in Neural Information Processing Systems, vol. 32, 2019, https://proceedings.neurips.cc/paper/2019/file/fa3060edb66e6ff4507886f9912e1ab9-Paper.pdf.

[7] M. DEREZIŃSKI AND M. W. MAHONEY, *Determinantal point processes in randomized numerical linear algebra*, Notices of the American Mathematical Society, 68 (2021), p. 1, https://doi.org/10.1090/noti2202.

[8] A. DESHPANDE AND S. VEMPALA, *Adaptive sampling and fast low-rank matrix approximation*, in Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, J. Díaz, K. Jansen, J. D. P. Rolim, and U. Zwick, eds., Lecture Notes in Computer Science, Berlin, 2006, Springer, pp. 292–303, https://doi.org/10.1007/11830924_28.

[9] S. FINE AND K. SCHEINBERG, *Efficient SVM training using low-rank kernel representations*, Journal of Machine Learning Research, 2 (2002), p. 243–264, https://dl.acm.org/doi/10.5555/944790.944812.

[10] A. FRIEZE, R. KANNAN, AND S. VEMPALA, *Fast Monte-Carlo algorithms for finding low-rank approximations*, Journal of the ACM, 51 (2004), pp. 1025–1041, https://doi.org/10.1145/1039488.1039494.

[11] G. GAUTIER, G. POLITO, R. BARDENET, AND M. VALKO, *DPPy: DPP sampling with Python*, Journal of Machine Learning Research, 20 (2019), pp. 1–7, http://jmlr.org/papers/v20/19-179.html.

[12] A. GITTENS, *The spectral norm error of the naïve Nyström extension*, arXiv:1110.5305 [cs, math], (2011).

[13] M. GU AND L. MIRANIAN, *Strong rank revealing Cholesky factorization*, Electronic Transactions on Numerical Analysis, 17 (2004), pp. 76–92, http://eudml.org/doc/125035.

[14] V. GURUSWAMI AND A. K. SINOP, *Optimal column-based low-rank matrix reconstruction*, in Proceedings of the 2012 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2012, pp. 1207–1214, https://doi.org/10.1137/1.9781611973099.95.

[15] N. HALKO, P. G. MARTINSSON, AND J. A. TROPP, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), pp. 217–288, https://doi.org/10.1137/090771806.

[16] N. J. HIGHAM, *Accuracy and Stability of Numerical Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, 2nd ed ed., 2002, https://doi.org/10.1137/1.9780898718027.

[17] S. KUMAR, M. MOHRI, AND A. TALWALKAR, *Sampling techniques for the Nyström method*, in Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, vol. 5 of Proceedings of Machine Learning Research, 2009, pp. 304–311, https://proceedings.mlr.press/v5/kumar09a.html.

[18] C.-K. LI AND R. MATHIAS, *Extremal characterizations of the Schur complement and resulting inequalities*, SIAM Review, 42 (2000), pp. 233–246, https://doi.org/10.1137/S0036144599337290.

[19] P.-G. MARTINSSON AND J. A. TROPP, *Randomized numerical linear algebra: Foundations and algorithms*, Acta Numerica, 29 (2020), p. 403–572, https://doi.org/10.1017/S0962492920000021.

[20] C. MUSCO AND C. MUSCO, *Recursive sampling for the Nyström method*, in Advances in Neural Information Processing Systems, vol. 30, 2017, https://proceedings.neurips.cc/paper/2017/file/a03fa30821986dff10fc66647c84c9c3-Paper.pdf.

[21] C. MUSCO AND D. P. WOODRUFF, *Sublinear time low-rank approximation of positive semidefinite matrices*, in 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), 2017, pp. 672–683, https://doi.org/10.1109/FOCS.2017.68.

[22] J. POULSON, *High-performance sampling of generic determinantal point processes*, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 378 (2020), p. 20190059, https://doi.org/10.1098/rsta.2019.0059.

[23] A. RAHIMI AND B. RECHT, *Random features for large-scale kernel machines*, in Advances in Neural Information Processing Systems, vol. 20, 2007, https://proceedings.neurips.cc/paper/2007/file/013a006f03dbc5392effeb8f18fda755-Paper.pdf.

[24] R. RAMAKRISHNAN, P. O. DRAL, M. RUPP, AND O. A. VON LILIENFELD, *Quantum chemistry structures and properties of 134 kilo molecules*, Scientific Data, 1 (2014), p. 140022, https://doi.org/10.1038/sdata.2014.22.

[25] L. RUDDIGKEIT, R. VAN DEURSEN, L. C. BLUM, AND J.-L. REYMOND, *Enumeration of 166 billion organic small molecules in the chemical universe database gdb-17*, Journal of Chemical Information and Modeling, 52 (2012), pp. 2864–2875, https://doi.org/10.1021/ci300415d.

[26] A. RUDI, D. CALANDRIELLO, L. CARRATINO, AND L. ROSASCO, *On fast leverage score sampling and optimal learning*, in Advances in Neural Information Processing Systems, vol. 31, 2018, https://proceedings.neurips.cc/paper/2018/file/56584778d5a8ab88d6393cc4cd11e090-Paper.pdf.

[27] B. SCHÖLKOPF AND A. J. SMOLA, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press, Cambridge, MA, 2018, https://doi.org/10.7551/mitpress/4175.001.0001.

[28] A. STUKE, M. TODOROVIĆ, M. RUPP, C. KUNKEL, K. GHOSH, L. HIMANEN, AND P. RINKE, *Chemical diversity in molecular orbital energy predictions with kernel ridge regression*, The Journal of Chemical Physics, 150 (2019), p. 204121, https://doi.org/10.1063/1.5086105.

[29] A. SZABO AND N. S. OSTLUND, *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*, Dover Publications, Mineola, NY, 2017.

[30] J. A. TROPP AND R. J. WEBBER, *Randomized algorithms for low-rank matrix approximation: Design, analysis, and applications*, In preparation, (2022).

[31] J. A. TROPP, A. YURTSEVER, M. UDELL, AND V. CEVHER, *Fixed-rank approximation of a positive-semidefinite matrix from streaming data*, in Advances in Neural Information Processing Systems, vol. 30, 2017, https://proceedings.neurips.cc/paper/2017/file/4558dbb6f6f8bb2e16d03b85bde76e2c-Paper.pdf.

DRAFT

[32] A. VANRAES, C. MUSCO, AND C. MUSCO, *Python implementation of the RLS-Nyström Method*, July 19 2019, https://github.com/axelv/recursive-nystrom.

[33] U. VON LUXBURG, *A tutorial on spectral clustering*, Statistics and Computing, 17 (2007), pp. 395–416, https://doi.org/10.1007/s11222-007-9033-z.

[34] C. WILLIAMS AND M. SEEGER, *Using the Nyström method to speed up kernel machines*, in Advances in Neural Information Processing Systems, vol. 13, 2000, https://proceedings.neurips.cc/paper/2000/file/19de10adbaa1b2ee13f77f679fa1483a-Paper.pdf.

[35] T. YANG, Y.-F. LI, M. MAHDAVI, R. JIN, AND Z.-H. ZHOU, *Nyström method vs random Fourier features: A theoretical and empirical comparison*, in Advances in Neural Information Processing Systems, vol. 25, 2012, https://proceedings.neurips.cc/paper/2012/file/621bf66ddb7c962aa0d22ac97d69b793-Paper.pdf.

[36] F. ZHANG, *Matrix Theory: Basic Results and Techniques*, Springer, New York, 2nd ed., 2011, https://link.springer.com/book/10.1007/978-1-4614-1099-7.