Journal of Machine Learning Research 27 (2026) 1-48 Submitted 2/25; Revised 1/26; Published 6/26

Have ASkotch: A Neat Solution for Large-Scale Kernel Ridge Regression

Pratik Rathore PRATIKR@ALUMNI.STANFORD.EDU
Department of Electrical Engineering
Stanford University

Zachary Frangella ZFRANGELLA@ALUMNI.STANFORD.EDU
Department of Management Science & Engineering
Stanford University

Jiaming Yang JIAMYANG@UMICH.EDU
Department of Electrical Engineering and Computer Science
University of Michigan

Michał Dereziński DEREZIN@UMICH.EDU
Department of Electrical Engineering and Computer Science
University of Michigan

Madeleine Udell UDELL@STANFORD.EDU
Department of Management Science & Engineering
Stanford University

Editor: Zaid Harchaoui

Abstract

Kernel ridge regression (KRR) is a fundamental computational tool, appearing in problems that range from computational chemistry to health analytics, with a particular interest due to its starring role in Gaussian process regression. However, full KRR solvers are challenging to scale to large datasets: both direct (e.g., Cholesky decomposition) and iterative methods (e.g., PCG) incur prohibitive computational and storage costs. The standard approach to scale KRR to large datasets chooses a set of inducing points and solves an approximate version of the problem, inducing points KRR. However, the resulting solution tends to have worse predictive performance than the full KRR solution. In this work, we introduce a new solver, `ASkotch`, for full KRR that provides better solutions faster than state-of-the-art solvers for full and inducing points KRR. `ASkotch` is a scalable, accelerated, iterative method for full KRR that provably obtains linear convergence. Under appropriate conditions, we show that `ASkotch` obtains condition-number-free linear convergence. This convergence analysis rests on the theory of ridge leverage scores and determinantal point processes. `ASkotch` outperforms state-of-the-art KRR solvers on a testbed of 23 large-scale KRR regression and classification tasks derived from a wide range of application domains, demonstrating the superiority of full KRR over inducing points KRR. Our work opens up the possibility of as-yet-unimagined applications of full KRR across a number of disciplines.

Keywords: kernel ridge regression, sketch-and-project methods, Nesterov acceleration, Nyström approximation, ridge leverage scores, determinantal point processes

1. Introduction

Kernel ridge regression (KRR) is one of the most popular methods in machine learning, with important applications in computational chemistry (Stuke et al., 2019; Blücher et al., 2023; Parkinson and Wang, 2023), healthcare (Cheng et al., 2020; Wu et al., 2021; Townes and Engelhardt, 2023), and, more recently, scientific machine learning (Raissi et al., 2017; Meanti et al., 2024; Batlle et al., 2024). Given a kernel function $k(x, x')$ and training set $\{(x_i, y_i)\}_{i=1}^n$, *full KRR* seeks the function f in a reproducing kernel Hilbert space $\mathcal{H}$ that satisfies

$$\underset{f\in\mathcal{H}}{\text{minimize}} \ \frac{1}{2}\sum_{i=1}^{n}\left(f(x_i)-y_i\right)^2+\frac{\lambda}{2}\|f\|_{\mathcal{H}}^2. \tag{1}$$

The celebrated representer theorem (Kimeldorf and Wahba, 1970; Schölkopf and Smola, 2002) says that the solution of (1) lies in the subspace

$$\mathcal{H}_n=\left\{f\in\mathcal{H} : f(x)=\sum_{j=1}^{n} w^{(j)}k(x,x_j), \text{ where } w^{(j)}\in\mathbb{R}\right\}.$$

With this reduction, the infinite-dimensional optimization problem in (1) collapses to a finite-dimensional, convex, least-squares problem:

$$\underset{w\in\mathbb{R}^n}{\text{minimize}} \ \mathcal{L}_{\text{full}}(w) \coloneqq \frac{1}{2}\|Kw-y\|^2+\frac{\lambda}{2}\|w\|_K^2. \tag{2}$$

Here K is a $n\times n$ kernel matrix whose entries are given by $K_{ij}=k(x_i,x_j)$. The optimal weights $w_\star$ in (2) are the solution of the linear system

$$K_\lambda w_\star = y, \tag{3}$$

where $K_\lambda \coloneqq K+\lambda I$.

Despite its popularity, full KRR is challenging to scale to large datasets: the state-of-the-art methods for solving (3) have costs that are superlinear in n. Direct methods (e.g., Cholesky decomposition) have $\mathcal{O}(n^3)$ computational complexity and $\mathcal{O}(n^2)$ storage complexity. Therefore, when $n \gtrsim 10^4$, Cholesky decomposition becomes unsuitable for solving (3). Iterative methods such as preconditioned conjugate gradient (PCG) have per-iteration complexity $\mathcal{O}(n^2)$. In addition to expensive iterations, most PCG methods employ low-rank preconditioners, which require $\mathcal{O}(nr)$ storage, where r is a rank parameter which needs to be sufficiently large to ensure effective preconditioning. Therefore, when $n \gtrsim 10^6$, PCG is either (i) too slow or (ii) requires too much memory to solve (3).

The standard approach for addressing the scalability issues of full KRR is to select a set of m *inducing points* from the training set, and solve the *inducing points KRR* problem (Schölkopf and Smola, 2002; Rudi et al., 2015). The inducing points KRR objective is

$$\underset{w\in\mathbb{R}^m}{\text{minimize}} \ \mathcal{L}_{\text{ind}}(w) \coloneqq \frac{1}{2}\|K_{nm}w-y\|^2+\frac{\lambda}{2}\|w\|_{K_{mm}}^2, \tag{4}$$

where K_{nm} is the $n \times m$ matrix of the columns of K identified by the inducing points, and K_{mm} is the corresponding principal submatrix of K. The optimal weights $w_\star$ in (4) are the solution of the linear system

$$(K_{nm}^T K_{nm} + \lambda K_{mm}) w_\star = K_{nm}^T y. \tag{5}$$

The linear system (5) accesses K only through K_{nm} and K_{mm}, reducing the storage and per-iteration computation needed to solve the KRR problem. As a result, Cholesky decomposition has $\mathcal{O}(nm^2 + m^3)$ computational complexity and requires $\mathcal{O}(m^2)$ storage. Consequently, when $m \gtrsim 10^5$, Cholesky is unsuitable for solving (5). State-of-the-art PCG methods for inducing points KRR, like Falkon (Rudi et al., 2017; Meanti et al., 2020) and KRILL (Díaz et al., 2023), require $\mathcal{O}(m^3)$ time to construct the preconditioner and $\mathcal{O}(m^2)$ storage. Thus, when $m \gtrsim 10^5$, these PCG methods are also unsuitable for solving (5).

Alas, scale matters: previous work has established that increasing the number of inducing points leads to better predictive performance (Frangella et al., 2023; Díaz et al., 2023; Abedsoltan et al., 2023). Moreover, full KRR often allows better predictive performance than inducing points KRR (Wang et al., 2019; Frangella et al., 2023; Díaz et al., 2023). Therefore, a new, more scalable approach to KRR is needed for better prediction on large-scale tasks.

In this work, we turn conventional wisdom around, demonstrating that better, faster solutions can be achieved by targeting the full KRR problem rather than the inducing points KRR approximation. We do so by developing the iterative method `ASkotch` to solve full KRR. `ASkotch` has (i) per-iteration costs that are linear in n and (ii) storage costs that are linear in the preconditioner rank r but are independent of n. These properties allow `ASkotch` to scale to larger datasets than existing state-of-the-art methods for solving (3). `ASkotch` possesses strong theoretical guarantees and easy-to-set hyperparameters that deliver reliable performance. Furthermore, `ASkotch` exploits parallelism on modern hardware accelerators like GPUs, which is essential for scaling to large KRR problems.

Fig. 1 demonstrates the core messages of our work—full KRR can scale to massive datasets and outperform inducing points KRR with respect to predictive performance. On a dataset with $n = 10^8$ samples, `ASkotch` scales better than the standard approach to full KRR: PCG is unable to complete a single iteration within a 24-hour time limit. Moreover, `ASkotch` is more reliable than the KRR solvers EigenPro 2.0 and EigenPro 3.0 (Ma and Belkin, 2019; Abedsoltan et al., 2023) when all three methods are run with their default hyperparameters: EigenPro 2.0 and EigenPro 3.0 diverge, while `ASkotch` reaches a root mean square error (RMSE) of approximately 230. Finally, `ASkotch` outperforms Falkon on RMSE. While Meanti et al. (2020) scales an optimized version of Falkon to the full taxi dataset ($n \approx 10^9$) with $m = 10^5$ inducing points, `ASkotch` is solving a problem of size $10^8 \cdot 10^8 = 10^{16}$, which is two orders of magnitude larger than the problem solved by Falkon ($10^5 \cdot 10^9 = 10^{14}$).

Table 1 compares the capabilities of `ASkotch` to state-of-the-art methods for KRR. `ASkotch` is the only method that solves full KRR while having a moderate memory requirement, reliable default hyperparameters, and theoretical convergence guarantees. In the EigenPro GitHub repositories, there is no option for the user to set hyperparameters such as the learning rate or gradient batchsize. Since the EigenPro methods diverge on taxi

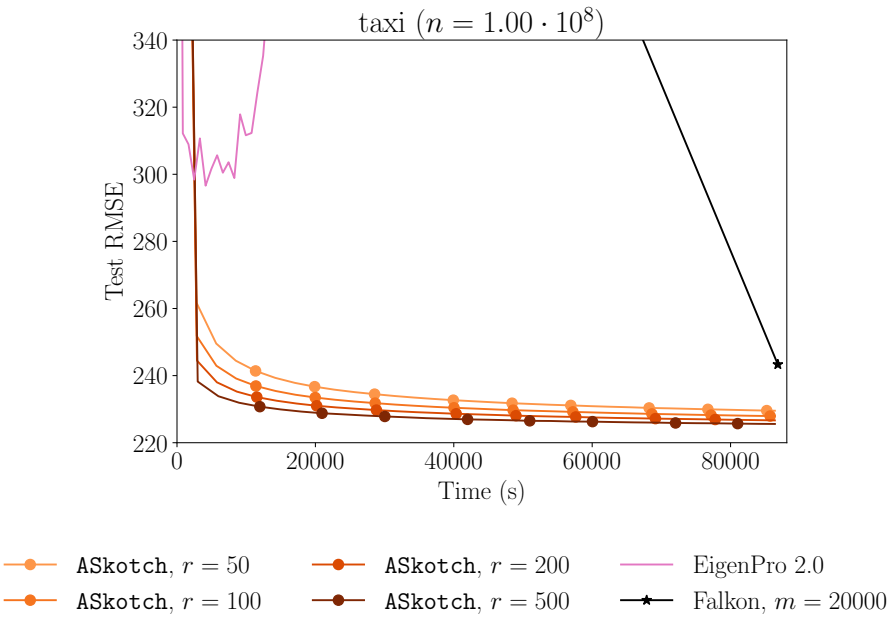

Figure 1: Full KRR is advantageous over inducing points KRR, even for large problems. Our method `ASkotch`, run with its default hyperparameters, outperforms the state-of-the-art for both full and inducing points KRR on a subsample of the taxi dataset. Falkon is limited to $m = 2 \cdot 10^4$ inducing points due to memory constraints. State-of-the-art Nyström PCG methods—Gaussian Nyström (Frangella et al., 2023) and Randomly Pivoted Cholesky (Díaz et al., 2023; Epperly et al., 2025)—with a rank $r = 50$ preconditioner fail to complete a single iteration. EigenPro 2.0 and EigenPro 3.0 (not shown) diverge on their default hyperparameters. All methods have a 24-hour time limit and are run on a single 48 GB NVIDIA RTX A6000 GPU.

Algorithm	Full KRR?	Memory-efficient?	Reliable defaults?	Converges?
`ASkotch`	✓	✓	✓	✓
EigenPro 2.0	✓	✓	✗	✓
EigenPro 3.0	✗	✓	✗	✗
PCG	✓	✗	✓	✓
Falkon	✗	✗	✓	✓

Table 1: A comparison of the capabilities of `ASkotch` and state-of-the-art methods for KRR. "Reliable defaults" indicates whether the method comes with default hyperparameters that work well in practice. "Converges" indicates whether the method has a rigorous linear convergence guarantee.

and several other datasets in this work, we conclude that the default hyperparameters for EigenPro can be unreliable.

Algorithm	Per-iteration complexity	Storage complexity	Total computational complexity
PCG	$\mathcal{O}(n^2)$	$\mathcal{O}(d^\lambda(K)n)$	$\tilde{\mathcal{O}}(n^2)$
EigenPro 2.0	$\mathcal{O}(nb_g + rs)$	$\mathcal{O}(rs)$	$\tilde{\mathcal{O}}\left(\frac{\lambda_r(K)}{\lambda_{\min}(K)}n^2\right)$
`ASkotch`	$\mathcal{O}(nb)$	$\mathcal{O}\left(d^\lambda(K)b\right)$	$\tilde{\mathcal{O}}\left(n^2\right)$

Table 2: Iteration and storage complexities for state-of-the-art full KRR methods. Storage complexity refers to memory that is explicitly used by the algorithm (this excludes matrix-vector product oracles). Total computational complexity refers to the total cost required by the algorithm to compute an ϵ-approximate solution, omitting logarithmic dependencies upon ϵ. Here, $b_g \leq n$ is the stochastic gradient batchsize used by EigenPro 2.0, $s \leq n$ is the sample size used to construct the rank-r preconditioner in EigenPro 2.0, $b \leq n$ is the blocksize in `ASkotch`, and $d^\lambda(K)$ is the λ-effective dimension of K (Definition 4). `ASkotch` has the same total cost as PCG but has lower storage and per-iteration costs.

1.1 Contributions

We make significant algorithmic and theoretical contributions to the literature on KRR solvers and sketch-and-project methods.

1. We develop the iterative method `ASkotch` for solving full KRR. The key to our approach lies in carefully combining two methodologies: sketch-and-project iterative solvers (Gower and Richtárik, 2015) and Nyström matrix approximations (Williams and Seeger, 2000; Tropp et al., 2017). We incorporate the Nyström approximation into the sketch-and-project update in a way that preserves convergence, and combine this with Nesterov acceleration to solve (3) with reduced time and memory requirements.

2. We establish fine-grained convergence guarantees for `ASkotch` based on ridge leverage score sampling, determinantal point processes, and careful spectral approximation arguments. A detailed overview of our theoretical contributions is presented in Section 1.2.

3. We perform an extensive empirical evaluation comparing `ASkotch` to the existing state-of-the-art for full and inducing points KRR. We show that `ASkotch` outperforms EigenPro 2.0, EigenPro 3.0, PCG, and Falkon on predictive performance for 23 large-scale KRR problems (typically, $n \geq 10^5$). Moreover, we show that full KRR consistently obtains equivalent or better predictive performance than inducing points KRR, establishing the superiority of full KRR when combined with `ASkotch`. We provide open-source code in PyTorch to make it easy to reproduce our experiments and run `ASkotch` on new problems on CPU and GPU hardware.

1.2 Our Techniques

Our main theoretical contributions lie in the convergence analysis of `ASkotch`. Establishing convergence of `ASkotch` requires overcoming two fundamental challenges, discussed below.

Technical Contribution 1: *First-moment analysis of SAP projector via ARLS-to-DPP reduction.* Our first contribution overcomes a fundamental limitation in the existing

theory of sketch-and-project (SAP) solvers. The convergence of SAP methods is controlled by a random projection matrix that we call the *SAP projector*, with the convergence rate being determined by the smallest eigenvalue of the expected SAP projector. Existing approaches for analyzing the smallest eigenvalue of the expected SAP projector (e.g., Mutny et al., 2020; Dereziński and Yang, 2024; Dereziński et al., 2025a) require either (1) an expensive, impractical sampling scheme known as a Determinantal Point Process (DPP), or (2) preprocessing the input matrix with the Randomized Hadamard Transform, which is prohibitive for large problems due to its $\mathcal{O}(n^2)$ storage cost.

To address this challenge, we adopt a sampling scheme called approximate ridge leverage score (ARLS) sampling (Alaoui and Mahoney, 2015), which is less expensive (both in compute and storage) than either of these approaches. Our convergence analysis develops a reduction from ARLSs to DPPs (Lemma 12), by relating the former to the marginals of a DPP through a novel measure concentration argument. This reduction allows us to establish a non-trivial lower bound on the smallest eigenvalue of the expected SAP projector when using ARLS sampling (Lemma 10). We believe this result is of independent interest and could be used to improve the convergence analysis of other iterative solvers that use the sketch-and-project paradigm.

Technical Contribution 2: *First-moment analysis of Nyström projector.* `ASkotch` replaces the linear system solve in SAP with an approximate solution based on Nyström sketch-and-solve (Bach, 2013; Alaoui and Mahoney, 2015; Frangella et al., 2023). In contrast, prior sketch-and-project solvers (Gower and Richtárik, 2015; Gower et al., 2018; Dereziński and Yang, 2024; Dereziński et al., 2025a) assume the SAP linear system is either solved exactly or to sufficient accuracy at each iteration using an iterative solver. The use of Nyström sketch-and-solve complicates the analysis in two ways: 1) the convergence rate is no longer controlled by an expected projection matrix; 2) the Nyström solution is generally not close to the exact solution (Frangella et al., 2023). The first issue prevents us from directly applying the classical convergence analysis of SAP, which is deeply reliant on the SAP projector actually being a projection matrix. The second issue means we cannot appeal to existing inexact SAP theory, which requires the approximate solution to be close to the exact solution.

Remarkably, although the matrix that controls convergence of `ASkotch` is not the expectation of a projection matrix, it is the expectation of a matrix that is *nearly* a projection matrix. We refer to this matrix as the *Nyström projector*. Using careful spectral approximation arguments, we show that the expected Nyström projector is lower bounded in the Loewner ordering by the expected SAP projector multiplied by a factor that captures the price of replacing the SAP linear system solve with Nyström sketch-and-solve. Thus, we are able to transfer the first-moment projector analysis from Technical Contribution 1 (which is for exact SAP) to `ASkotch`, which yields a lower bound on the smallest eigenvalue of the expected Nyström projector (Theorem 13). This lower bound on the smallest eigenvalue of the expected Nyström projector lets us establish a fine-grained convergence rate for `ASkotch` (Theorem 17). Using this theory, we prove that when the effective dimension of the kernel matrix is not too large, `ASkotch` enjoys fast condition-number-free linear convergence, for a near-optimal $\tilde{\mathcal{O}}(n^2 \log\left(\frac{1}{\epsilon}\right))$ computational complexity to reach an ϵ-approximate full KRR solution (Corollary 18). This convergence rate is comparable to that of PCG while improving over EigenPro 2.0 (Table 2).

1.3 Roadmap

Section 2 introduces `ASkotch` (and its non-accelerated variant, `Skotch`), and describes the techniques (sketch-and-project, Nyström approximation, automatic computation of stepsizes) that are used in this algorithm. Section 3 reviews existing methods to solve full and inducing points KRR and places `ASkotch` in the context of these works. Section 4 introduces several key quantities used in the convergence analysis of `ASkotch`. Section 5 develops lower bounds on the quantities introduced in Section 4 by exploiting connections between ARLS sampling and DPPs. Section 6 uses the lower bounds from Section 5 to establish a fine-grained convergence rate for `ASkotch` on full KRR. Section 7 demonstrates the superior performance of `ASkotch` over state-of-the-art full and inducing-points KRR solvers.

1.4 Notation

We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. Throughout the paper, $\mathcal{B}$ is a subset of $[n]$ that is sampled randomly according to some distribution. $I_{\mathcal{B}}$ concatenates the rows of I_n indexed by $\mathcal{B}$. Given a square matrix A, $A_{\mathcal{B}\mathcal{B}}$ denotes the principal submatrix of A indexed by $\mathcal{B}$.

$\mathrm{Tr}(\cdot)$ is the trace of a matrix, $\det(\cdot)$ is the determinant of a matrix, and $\mathrm{null}(\cdot)$ is the null space of a matrix. $\mathbb{S}^n_{++}$ ($\mathbb{S}^n_+$) denotes the convex cone of pd (psd) matrices in $\mathbb{R}^{n\times n}$. The symbol $\preceq$ denotes the Loewner order on the convex cone of psd matrices: $A \preceq B$ means $B - A$ is psd. For a square matrix A and constant λ, we define $A_\lambda \coloneqq A + \lambda I$. We use $\|\cdot\|$ to denote norms; $\|\cdot\|$ with no subscript is the Euclidean norm. For a matrix $A \in \mathbb{S}^n_+$, its eigenvalues in decreasing order are $\lambda_1(A) \geq \lambda_2(A) \geq \ldots \geq \lambda_n(A)$. A^+ is the Moore-Penrose pseudoinverse of a matrix A. If $A \in \mathbb{S}^n_+$, the smallest non-zero eigenvalue is denoted by $\lambda^+_{\min}(A)$. For a positive integer b and matrix A, $\lfloor A \rfloor_b$ is the best rank-b approximation of A with respect to the spectral norm.

The symbols $\Omega(\cdot)$, $\Theta(\cdot)$, and $\mathcal{O}(\cdot)$ are the standard quantities from asymptotic complexity analysis. $\tilde{\Omega}(\cdot)$, $\tilde{\Theta}(\cdot)$, and $\tilde{\mathcal{O}}(\cdot)$ are used to hide poly-log factors.

2. Algorithms

We propose `ASkotch` (**A**ccelerated **s**calable **k**ernel **o**ptimization and **t**raining with **c**oordinate sketch-and-project and approximate **H**essians, Algorithm 1) to solve full KRR. We introduce the `ASkotch` algorithm in Section 2.1, and we follow this by describing the key building blocks of the method, such as approximate sketch-and-project (Section 2.2), randomized Nyström approximation (Section 2.3), and automatic computation of the stepsize (Section 2.4). `ASkotch` comes with default hyperparameter settings (Section 2.5), which reduces the effort required by practitioners to tune the method; we use these defaults in our experiments (Section 7).

2.1 `ASkotch`

At each iteration, `ASkotch` randomly samples a block $\mathcal{B}$ containing b distinct coordinates (we omit the dependence of $\mathcal{B}$ on i to avoid notational clutter). `ASkotch` then computes a rank-r Nyström approximation of $K_{\mathcal{B}\mathcal{B}}$ and computes a *preconditioned smoothness constant* for the block, $L_{P_{\mathcal{B}}}$, which is used to set the stepsize. Next, `ASkotch` computes the approximate

Algorithm 1 `ASkotch`

Require: blocksize b, coordinate sampling distribution $\mathcal{P}$, acceleration parameters $\hat{\mu}, \hat{\nu}$, acceleration flag use_accel, rank r, damping ρ, kernel oracle K, targets y, ridge parameter λ, number of iterations N, initialization w_0

 # Compute acceleration parameters
$\beta \leftarrow 1 - \sqrt{\hat{\mu}/\hat{\nu}}$
$\gamma \leftarrow 1/\sqrt{\hat{\mu}\hat{\nu}}$
$\alpha \leftarrow 1/(1+\gamma\hat{\nu})$
$v_0 \leftarrow w_0, z_0 \leftarrow w_0$

for $i = 0, 1, \ldots, N-1$ **do**
  Sample a block of coordinates $\mathcal{B} \subseteq [n]$ according to $\mathcal{P}$ ▷ $|\mathcal{B}| = b$
  $\hat{K}_{\mathcal{B}\mathcal{B}} \leftarrow \texttt{Nyström}(K_{\mathcal{B}\mathcal{B}}, r)$ ▷ Low-rank approximation of $K_{\mathcal{B}\mathcal{B}}$
  $L_{P_\mathcal{B}} \leftarrow \texttt{get\_L}(K_{\mathcal{B}\mathcal{B}} + \lambda I, \hat{K}_{\mathcal{B}\mathcal{B}}, \rho)$ ▷ Via powering

  # Update iterates
  if use_accel **then**
    $d_i \leftarrow I_\mathcal{B}^T(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1}((K_\lambda)_{\mathcal{B}:}z_i - y_\mathcal{B})$ ▷ Approximate projection; costs $\mathcal{O}(nb + rb)$
    $w_{i+1} \leftarrow z_i - (L_{P_\mathcal{B}})^{-1}d_i$ ▷ Costs $\mathcal{O}(b)$
    $v_{i+1} \leftarrow \beta v_i + (1-\beta)z_i - \gamma(L_{P_\mathcal{B}})^{-1}d_i$ ▷ Costs $\mathcal{O}(n)$
    $z_{i+1} \leftarrow \alpha v_{i+1} + (1-\alpha)w_{i+1}$ ▷ Costs $\mathcal{O}(n)$
  else
    # Non-accelerated version (`Skotch`)
    $d_i \leftarrow I_\mathcal{B}^T(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1}((K_\lambda)_{\mathcal{B}:}w_i - y_\mathcal{B})$ ▷ Approximate projection; costs $\mathcal{O}(nb + rb)$
    $w_{i+1} \leftarrow w_i - (L_{P_\mathcal{B}})^{-1}d_i$ ▷ Costs $\mathcal{O}(b)$
  end if
end for
return w_N ▷ Approximate solution to $K_\lambda w = y$

projection step d_i. Finally, `ASkotch` combines the approximate projection step d_i with Nesterov acceleration (Nesterov, 2018) to update its estimate of the solution.

`ASkotch` is compatible with any coordinate sampling distribution $\mathcal{P}$. Our implementation uses two approaches inspired by popular sampling distributions in the kernel literature: uniform sampling and ARLS sampling. The uniform sampling distribution places an equal weight of $1/n$ on each coordinate, while ARLS sampling weights coordinates with probabilities proportional to their approximate RLSs (Definition 4). In our implementation, we approximate the RLSs using BLESS (Rudi et al., 2018; Gautier et al., 2019).

`ASkotch` has a flag indicating whether or not to use Nesterov acceleration: when this flag is False, acceleration is disabled. Throughout the rest of the paper, we will use the name `Skotch` to refer to the non-accelerated version of `ASkotch`.

2.2 Approximate Sketch-and-Project

Sketch-and-project (SAP) (Gower and Richtárik, 2015; Richtárik and Takáč, 2020) is a randomized, iterative framework to solve consistent linear systems. SAP for the full KRR linear system (3) takes in two parameters: (i) a fixed, pd matrix $Q \in \mathbb{S}^n_{++}$ and (ii) a distribution $\mathcal{D}$ over matrices in $\mathbb{R}^{b\times n}$, where b is a positive integer. At iteration i, SAP draws a sketching matrix $S_i \overset{\text{i.i.d.}}{\sim} \mathcal{D}$ and computes the next iterate by solving the optimization problem

$$w_{i+1} = \begin{array}{ll} \arg\min_w & \|w - w_i\|_Q^2 \\ \text{subject to} & S_i K_\lambda w = S_i y. \end{array} \tag{6}$$

In other words, SAP progresses by projecting the current iterate w_i (with respect to the Q-norm) onto the solution space of a sketched version of (3). Importantly, when $Q = K_\lambda$ and $\mathcal{D}$ is the uniform distribution over row selection matrices of size $b \times n$, the SAP update has the following closed form:

$$w_{i+1} = w_i - Q^{-1}K_\lambda S_i^T (S_i K_\lambda Q^{-1} K_\lambda S_i^T)^+ (S_i K_\lambda w_i - S_i y) \tag{7}$$

$$= w_i - I_{\mathcal{B}}^T (K_{\mathcal{BB}} + \lambda I)^{-1} ((K_\lambda)_{\mathcal{B}:} w_i - y_{\mathcal{B}}). \tag{8}$$

Nesterov acceleration has been applied to a wide range of iterative optimization methods to obtain algorithms with faster convergence rates (Allen-Zhu et al., 2016; Liu and Wright, 2016; Allen-Zhu, 2018; Ye et al., 2020). SAP is no exception—Gower et al. (2018) combines SAP with Nesterov acceleration and shows that Nesterov-accelerated SAP (NSAP) converges at least as fast as its non-accelerated counterpart.

In theory, NSAP improves over PCG when $b = o(n^{2/3})$. However, a larger blocksize b is often beneficial for SAP methods—for example, the convergence rate for the randomized Newton method enjoys a superlinear speedup as b increases (Gower and Richtárik, 2015). Using a direct method to solve the linear system in (7) requires $\mathcal{O}(b^3)$ computation, which becomes slow for $b \gtrsim 10^4$. Dereziński and Yang (2024); Dereziński et al. (2025a) propose inexactly solving this linear system to a tolerance ϵ using PCG with a preconditioner P. Doing so requires $\mathcal{O}(b^2\sqrt{\kappa_P}\log(1/\epsilon))$ computation, where κ_P is the preconditioned condition number. However, for $b \gtrsim 10^4$, repeatedly applying PCG may be too slow to be practical.

`ASkotch` also approximates the SAP update, but it does so in a way that allows for a blocksize $b \gtrsim 10^4$. The matrix $K_{\mathcal{BB}} + \lambda I$ in the SAP update (8) is replaced by a regularized rank-r Nyström approximation (Section 2.3), $\hat{K}_{\mathcal{BB}} + \rho I$, where $\rho > 0$. This Nyström approximation can be computed rapidly, and the resulting linear system can be solved in $\mathcal{O}(br)$ time.

The theoretical analysis of SAP uses the fact that $Q^{-1}K_\lambda S_i^T (S_i K_\lambda Q^{-1} K_\lambda S_i^T)^+ S_i K_\lambda$ in (7) is a projection with respect to the Q-inner product (Gower and Richtárik, 2015). However, by replacing $S_i K_\lambda Q^{-1} K_\lambda S_i^T$ by a low-rank approximation, `ASkotch` loses this projection property. Consequently, `ASkotch` requires a (dynamic) stepsize to converge, unlike SAP, which is guaranteed to converge with a constant stepsize of 1. We provide a principled, automated method for computing this stepsize in Section 2.4.

2.3 Randomized Nyström Approximation

The Nyström approximation (Williams and Seeger, 2000; Bach, 2013; Alaoui and Mahoney, 2015; Tropp et al., 2017) is a well-known technique for producing low-rank approximations to psd matrices. Since kernel matrices have fast spectral decay (i.e., they have approximate low-rank structure) (Caponnetto and DeVito, 2007; Bach, 2013; Tu et al., 2016; Ma and Belkin, 2017; Belkin, 2018) we replace $K_{\mathcal{BB}}$ in (8) with a Nyström approximation, which is easier to invert.

Given a symmetric psd matrix $M \in \mathbb{S}_+^p$, the randomized Nyström approximation of M with respect to a random test matrix $\Omega \in \mathbb{R}^{p \times r}$ is

$$\hat{M} = (M\Omega)\left(\Omega^T M \Omega\right)^+ (M\Omega)^T.$$

Common choices for Ω include standard Gaussian random matrices, randomized Hadamard transforms, and sparse sign embeddings (Tropp et al., 2017). The latter two test matrices reduce the computational cost of the sketch $M\Omega$. Our theoretical analysis uses sparse sign embeddings due to their computational efficiency, but our implementation uses Gaussian random matrices for simplicity. In the future, we will extend our implementation to use randomized Hadamard transforms and sparse sign embeddings.

Our practical implementation, `Nyström` (Algorithm 2), follows Tropp et al. (2017, Algorithm 3). `Nyström` takes in a psd matrix $M \in \mathbb{S}_+^p$ and rank r and outputs a low-rank approximation $\hat{M} = \hat{U}\,\mathrm{diag}(\hat{\Lambda})\hat{U}^T$ to M in $\mathcal{O}(p^2r + pr^2)$ time[1], where $\hat{U} \in \mathbb{R}^{p\times r}$ is an orthogonal matrix containing approximate top-r eigenvectors of M and $\hat{\Lambda} \in \mathbb{R}^r$ is a vector containing approximate top-r eigenvalues of M. Note that `Nyström` never forms $\hat{M}$ as a matrix; rather, it returns the factors $\hat{U}$ and $\hat{\Lambda}$. For more details, please see Appendix A.1.

At each iteration, `ASkotch` computes a randomized Nyström approximation of the block kernel $K_{\mathcal{B}\mathcal{B}}$ in the SAP update (8). This approximation is always used in conjunction with a damping $\rho > 0$ to ensure positive definiteness. For any vector $v \in \mathbb{R}^p$, $(\hat{M} + \rho I)^{-1}v$ can be computed in $\mathcal{O}(pr)$ time using the Woodbury formula, which lets `ASkotch` cheaply compute the approximate projection step d_i.

2.4 Automatic Computation of the Stepsize

The stepsize is often challenging to set in iterative optimization methods: if the stepsize is too large, `ASkotch` will diverge, while if the stepsize is too small, `ASkotch` will make little progress towards the solution. In practice, tuning the stepsize can require an expensive hyperparameter search, which is inconvenient for practitioners. We present a practical approach for automatically computing the stepsize in `ASkotch`.

Our idea is to use a preconditioned smoothness constant to set the stepsize in `ASkotch`. The preconditioned smoothness constant in `ASkotch` is defined as

$$L_{P_{\mathcal{B}}} \coloneqq \lambda_1\left(\left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1/2}\left(K_{\mathcal{B}\mathcal{B}} + \lambda I\right)\left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1/2}\right).$$

`ASkotch` uses randomized powering (Kuczyński and Woźniakowski, 1992; Martinsson and Tropp, 2020) to compute $L_{P_{\mathcal{B}}}$. The total cost of the procedure is $\tilde{\mathcal{O}}(b^2)$. In practice, `ASkotch` uses just 10 iterations of powering. Hence, the cost of computing $L_{P_{\mathcal{B}}}$ is dominated by the $\mathcal{O}(nb)$ cost of computing the search direction d_i in `ASkotch`. `ASkotch` computes $L_{P_{\mathcal{B}}}$ using the subroutine `get_L` (Algorithm 3).

In Sections 4 and 6, we show that our approach for setting the stepsize guarantees that `Skotch` and `ASkotch` obtain linear convergence.

2.5 Default Hyperparameters

We provide default recommendations for setting hyperparameters in `ASkotch`. These recommendations are summarized in Table 3.

- Blocksize b and rank r: Choose as large as hardware allows. A good, general default is $b = n/100$ and $r = 100$.

1. For sparse sign embeddings, this complexity is reduced to $\tilde{\mathcal{O}}(pr + pr^2)$.

Hyperparameter	Default recommendation
b	$n/100$
$\mathcal{P}$	Uniform
$\hat{\mu}$	λ
$\hat{\nu}$	n/b
r	100
ρ	$\lambda + \lambda_r(\hat{K}_{\mathcal{B}\mathcal{B}})$

Table 3: Default hyperparameters for `ASkotch`.

- Damping ρ: Set adaptively for each block $\mathcal{B}$; we use $\rho = \lambda + \lambda_r(\hat{K}_{\mathcal{B}\mathcal{B}})$.
- Sampling distribution $\mathcal{P}$: Use the uniform sampling distribution. While we use ARLS sampling to prove fast convergence rates for `ASkotch`, it (i) performs similarly to uniform sampling in our ablations (Section 7) and (ii) computing approximate RLSs via BLESS costs $\tilde{\mathcal{O}}(nb^2)$.
- Acceleration parameters $\hat{\mu}$ and $\hat{\nu}$: Set $\hat{\mu} = \lambda$ and $\hat{\lambda} = n/b$. These choices are guided by the convergence analysis of `ASkotch`. However, the user must ensure that $\hat{\mu} \leq \hat{\nu}$ and $\hat{\mu}\hat{\nu} \leq 1$. We believe it is possible to make the acceleration in `ASkotch` parameter-free, as done in Dereziński et al. (2025b), but we leave this extension to future work.

3. Related Work

We review existing solvers for full and inducing points KRR and discuss how our methods compare to the literature.

3.1 Full KRR

Many prior works have developed methods to solve full KRR that avoid the $\mathcal{O}(n^3)$ cost of direct methods such as Cholesky decomposition. The various approaches can be roughly divided into 3 categories: (i) PCG methods, (ii) stochastic gradient methods, and (iii) SAP methods.

For PCG, much work has been done on developing efficient preconditioners (Cutajar et al., 2016; Avron et al., 2017; Frangella et al., 2023; Díaz et al., 2023). Among the numerous proposals, the most popular preconditioners are based on the Nyström approximation (Nyström, 1930). Nyström approximations can be constructed from the kernel matrix via uniform sampling of columns (Williams and Seeger, 2000), greedy pivoting (Harbrecht et al., 2012), leverage-score sampling (Musco and Musco, 2017), random projection (Tropp et al., 2017), and randomized pivoting (Chen et al., 2025; Epperly et al., 2025). Preconditioners have also been constructed using random-features approximations (Avron et al., 2017), but these do not perform as well in practice as Nyström preconditioners (Díaz et al., 2023; Frangella et al., 2023). Typically, if a Nyström preconditioner is constructed with a rank $r = \mathcal{O}(d^{\lambda}(K))$, then PCG converges at a condition-number-free rate with high probability (Zhao et al., 2022; Frangella et al., 2023). The downside of these PCG approaches is that

they exhibit an $\mathcal{O}(n^2)$ per-iteration cost and require $\mathcal{O}(nr)$ storage, making it difficult to scale these methods beyond $n \gtrsim 10^5$.

EigenPro and EigenPro 2.0 (Ma and Belkin, 2017, 2019) are the most well-known stochastic gradient methods for solving full KRR. Both EigenPro and EigenPro 2.0 set the regularization $\lambda = 0$ and solve the resulting KRR problem via preconditioned stochastic gradient descent. EigenPro 2.0 reduces the per-iteration cost to $\mathcal{O}(nb_g + rs)$, a significant improvement over the $\mathcal{O}(n^2)$ cost of PCG.

The third approach to solve full KRR problems is based on SAP. Tu et al. (2016) proposes randomized block Gauss-Seidel for full KRR. In follow-up work, Tu et al. (2017) develops NSAP for solving the full KRR problem. Gazagnadou et al. (2022) applies SAP for solving full KRR but uses count sketches instead of coordinate sampling to form $SK_\lambda S^T$ in the SAP update. Unfortunately, these methods have $\mathcal{O}(b^3)$ computational cost per iteration, which is too expensive for our largest example, taxi, where $b = 5 \cdot 10^4$. More recently, Lin et al. (2024) proposes a variant of NSAP where $K_{\mathcal{BB}} + \lambda I$ is replaced by the identity matrix. While this replacement removes the $\mathcal{O}(b^3)$ iteration cost associated with matrix factorization, its convergence becomes much slower.

Among these (approximate) SAP methods, `Skotch` and `ASkotch` are the only methods that guarantee condition-number-free convergence under reasonable assumptions on the kernel matrix, while also avoiding $\mathcal{O}(b^3)$ per-iteration costs.

3.2 Inducing Points KRR

PCG is also popular for solving inducing points KRR, with Falkon (Rudi et al., 2017; Meanti et al., 2020) and KRILL (Díaz et al., 2023) being the current state-of-the-art. KRILL is similar to Falkon, but solves (5) via PCG with a preconditioner constructed using a sparse sketch. KRILL has robust theoretical guarantees, and numerical tests in Díaz et al. (2023) show it yields comparable or better performance than Falkon. Nevertheless, KRILL still has a $\tilde{\mathcal{O}}(nm + m^3)$ runtime and requires $\mathcal{O}(m^2)$ storage. Consequently, it encounters the same runtime and storage barriers as Falkon for large m.

Meanti et al. (2020) develops a software package for Falkon that relies on extensive code and hardware optimizations. These optimizations include stabilizing Falkon in single precision, out-of-core matrix operations, parallelized memory transfers, and distribution over multiple GPUs. Consequently, Meanti et al. (2020) scales Falkon to $n = 10^9$ with $m = 10^5$. Despite these optimizations, Falkon still encounters fundamental memory barriers: Abedsoltan et al. (2023) shows that the optimized version of Falkon cannot scale beyond 2.56×10^5 inducing points when given 340 GB of RAM, which is larger than the memory budget available to most practitioners.

Abedsoltan et al. (2023) develops the stochastic gradient method EigenPro 3.0 for inducing points KRR. EigenPro 3.0 has a lower iteration and storage complexity than Falkon, which allows it to use more inducing points. Unfortunately, this method has no convergence guarantees.

4. Key Quantities in the Analysis of `ASkotch`

Establishing convergence of `ASkotch` involves quantities whose relevance is not obvious a priori. Therefore, we show how the key quantities in the convergence analysis arise. For

simplicity, our analysis focuses on `ASkotch` without acceleration (i.e., `Skotch`). Throughout the analysis, we omit the dependence of $\mathcal{B}$ on i to avoid notational clutter.

4.1 SAP and Nyström Projectors

We begin by introducing two matrices that will play a central role in the convergence analysis of `ASkotch`.

Definition 1 (SAP projector) *For a block $\mathcal{B}$, kernel matrix K, and regularization λ, the SAP projector is*

$$\Pi_{\mathcal{B}} \coloneqq K_\lambda^{1/2} I_{\mathcal{B}}^T (K_{\mathcal{B}\mathcal{B}} + \lambda I)^{-1} I_{\mathcal{B}} K_\lambda^{1/2}.$$

The SAP projector $\Pi_{\mathcal{B}}$ is an *exact* projection matrix, which controls the convergence rate of SAP methods.

Definition 2 (Nyström projector) *For a block $\mathcal{B}$, kernel matrix K, regularization λ, damping $\rho > 0$, and stepsize $\eta_{\mathcal{B}} > 0$, the Nyström projector is*

$$\hat{\Pi}_{\mathcal{B},\rho} \coloneqq \eta_{\mathcal{B}} K_\lambda^{1/2} I_{\mathcal{B}}^T (\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1} I_{\mathcal{B}} K_\lambda^{1/2}.$$

The Nyström projector controls the convergence rate of `ASkotch`. Note that the name Nyström projector is a slight abuse of terminology, as $\hat{\Pi}_{\mathcal{B},\rho}$ is not idempotent. However, for appropriately chosen $\eta_{\mathcal{B}}$, we will show that $\hat{\Pi}_{\mathcal{B},\rho}$ is closely related to the SAP projector.

4.2 One-Step Analysis of `Skotch` and SAP

We now show how the SAP and Nyström projectors arise by performing a one-step analysis of SAP and `Skotch`.

We begin by writing the updates for SAP and `Skotch`:

$$w_i = w_{i-1} - I_{\mathcal{B}}^T \left(K_{\mathcal{B}\mathcal{B}} + \lambda I\right)^{-1} I_{\mathcal{B}} (K_\lambda w_{i-1} - y), \tag{SAP}$$

$$w_i = w_{i-1} - \eta_{\mathcal{B}} I_{\mathcal{B}}^T \left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1} I_{\mathcal{B}} (K_\lambda w_{i-1} - y). \tag{Skotch}$$

Some elementary manipulations show that

$$\|w_i - w_\star\|_{K_\lambda}^2 \leq (w_{i-1} - w_\star)^T K_\lambda^{1/2} \left(I - \Pi_{\mathcal{B}}\right)^2 K_\lambda^{1/2} (w_{i-1} - w_\star), \tag{SAP}$$

$$\|w_i - w_\star\|_{K_\lambda}^2 \leq (w_{i-1} - w_\star)^T K_\lambda^{1/2} \left(I - \hat{\Pi}_{\mathcal{B},\rho}\right)^2 K_\lambda^{1/2} (w_{i-1} - w_\star). \tag{Skotch}$$

As $\Pi_{\mathcal{B}}$ is a projection, we can further deduce for SAP that

$$\|w_i - w_\star\|_{K_\lambda}^2 \leq (w_{i-1} - w_\star)^T K_\lambda^{1/2} \left(I - \Pi_{\mathcal{B}}\right) K_\lambda^{1/2} (w_{i-1} - w_\star). \tag{SAP}$$

Define[2]

$$\mu \coloneqq \lambda_{\min}(\mathbb{E}\left[\Pi_{\mathcal{B}}\right]).$$

2. If K_λ was only psd, then this quantity must be replaced by $\lambda_{\min}^+(\mathbb{E}[\Pi_{\mathcal{B}}])$, as $\mathbb{E}[\Pi_{\mathcal{B}}]$ would be singular. For our setting, this does not matter as K_λ is pd.

Then, taking the expectation conditioned on w_{i-1} yields

$$\mathbb{E}[\|w_i - w_\star\|_{K_\lambda}^2 \mid w_{i-1}] \leq (1-\mu)\,\|w_{i-1} - w_\star\|_{K_\lambda}^2. \tag{SAP}$$

Thus, the convergence of SAP is controlled by the smallest positive eigenvalue of the expected projection matrix $\mathbb{E}\left[\Pi_{\mathcal{B}}\right]$. As `Skotch` is an approximate version of SAP, we would like to deduce a similar result. Indeed, if we can ensure

$$\hat{\Pi}_{\mathcal{B},\rho} \preceq I,$$

then

$$\|w_i - w_\star\|_{K_\lambda}^2 \leq (w_{i-1} - w_\star)^T K_\lambda^{1/2}\left(I - \hat{\Pi}_{\mathcal{B},\rho}\right) K_\lambda^{1/2}(w_{i-1} - w_\star). \tag{Skotch}$$

Defining

$$\hat{\mu} \coloneqq \lambda_{\min}(\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]),$$

it immediately follows from the preceding display that

$$\mathbb{E}[\|w_i - w_\star\|_{K_\lambda}^2 \mid w_{i-1}] \leq (1-\hat{\mu})\,\|w_{i-1} - w_\star\|_{K_\lambda}^2. \tag{Skotch}$$

4.2.1 Setting the Stepsize η_B

The convergence of `Skotch` is controlled by $\hat{\mu}$, the smallest positive eigenvalue of $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$, similar to how μ controls the convergence of SAP. To conclude this relation, we require

$$\hat{\Pi}_{\mathcal{B},\rho} \preceq I.$$

This condition tells us that the stepsize $\eta_{\mathcal{B}}$ cannot be arbitrarily large. However, we also want to characterize the behavior of $\hat{\mu}$: when $\hat{K}_{\mathcal{B}\mathcal{B}}$ is a good approximation to $K_{\mathcal{B}\mathcal{B}}$, $\hat{\mu}$ should be closely related to μ. The following lemma formalizes this intuition.

Lemma 3 *Let $\rho > 0$ and $\mathcal{B} \subseteq [n]$ be fixed. Let $\hat{K}_{\mathcal{B}\mathcal{B}}$ be a Nyström approximation of $K_{\mathcal{B}\mathcal{B}}$. Define*

$$\begin{aligned}
L_{P_{\mathcal{B}}} &\coloneqq \lambda_{\max}\left((K_{\mathcal{B}\mathcal{B}} + \lambda I)^{1/2}\left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1}(K_{\mathcal{B}\mathcal{B}} + \lambda I)^{1/2}\right),\\
\hat{L}_{P_{\mathcal{B}}} &\coloneqq \max\{1, L_{P_{\mathcal{B}}}\},\\
\sigma_{P_{\mathcal{B}}} &\coloneqq \lambda_{\min}\left((K_{\mathcal{B}\mathcal{B}} + \lambda I)^{1/2}\left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1}(K_{\mathcal{B}\mathcal{B}} + \lambda I)^{1/2}\right).
\end{aligned}$$

If we set $\eta_B = 1/\hat{L}_{P_{\mathcal{B}}}$ in Definition 2, then the Nyström projector satisfies

$$\frac{\sigma_{P_{\mathcal{B}}}}{\hat{L}_{P_{\mathcal{B}}}}\Pi_{\mathcal{B}} \preceq \hat{\Pi}_{\mathcal{B},\rho} \preceq \Pi_{\mathcal{B}}. \tag{9}$$

Setting the `ASkotch` stepsize to $\eta_{\mathcal{B}} = 1/\hat{L}_{P_{\mathcal{B}}}$ ensures convergence and allows for a lower bound on $\lambda_{\min}(\hat{\Pi}_{\mathcal{B},\rho})$ in terms of $\lambda_{\min}(\Pi_{\mathcal{B}})$. Recall that we set $\eta_{\mathcal{B}} = 1/L_{P_{\mathcal{B}}}$ as the stepsize for `ASkotch` in Section 2. The conclusions of Lemma 3 still hold for this simpler stepsize. We only require $1/\hat{L}_{P_{\mathcal{B}}}$ for our theoretical analysis to show `ASkotch` achieves a better upper bound on the number of iterations to reach an ϵ-approximate solution of (3) than `Skotch`.

5. First-Moment Analysis of the Expected Nyström Projector

The argument in Section 4.2 shows that the convergence of `ASkotch` is controlled by $\hat{\mu}$, which is the smallest eigenvalue of the first moment of the Nyström projector, $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$. Analyzing $\hat{\mu}$ and $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$ directly seems impenetrable. Instead, we would like to derive a lower bound on $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$ in terms of the first moment of the SAP projector, $\mathbb{E}[\Pi_{\mathcal{B}}]$:

$$\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}] \succeq g(\rho,\lambda)\mathbb{E}[\Pi_{\mathcal{B}}], \tag{10}$$

where $g(\rho,\lambda)$ is a scalar function that captures the price of replacing $K_{\mathcal{B}\mathcal{B}}$ with $\hat{K}_{\mathcal{B}\mathcal{B}}$. Once (10) has been established, we immediately obtain

$$\hat{\mu} \geq g(\rho,\lambda)\mu. \tag{11}$$

Provided that we have a non-trivial lower bound on μ, we can combine (11) with the one-step analysis in Section 4.2 to obtain an explicit convergence rate for `Skotch`. We can obtain an explicit convergence rate for `ASkotch` using a similar approach, with some additional work to account for Nesterov acceleration.

There are two challenges in establishing the lower bound on $\hat{\mu}$ in (11):

1. **Deriving a meaningful lower bound on μ is non-trivial**. Existing works that establish such a lower bound rely on either expensive DPP sampling techniques (Mutny et al., 2020) or preprocessing the kernel matrix using a Hadamard transform (Dereziński and Yang, 2024; Dereziński et al., 2025a), which incurs a $\mathcal{O}(n^2)$ cost and requires materializing a $n \times n$ matrix in memory. Therefore, we must develop a different approach for establishing a lower bound on μ.

2. **Identifying the form of $g(\rho,\lambda)$ requires care**. The analysis to determine $g(\rho,\lambda)$ requires careful spectral approximation arguments based on the theory of the Nyström approximation.

In the following subsections, we address these challenges. In Section 5.1, we provide background on (approximate) ridge leverage scores and DPPs. In Section 5.2, we develop a novel ARLS-to-DPP reduction (Lemma 12) to establish a non-trivial lower bound on μ (Lemma 10). This reduction and its associated results can be applied to any psd matrix, making them of independent interest outside of this manuscript. In Section 5.3, we relate $\hat{\Pi}_{\mathcal{B},\rho}$ to $\Pi_{\mathcal{B}}$ using spectral approximation arguments based on the Nyström approximation. Combining these ideas, we show the following lower bound on $\hat{\mu}$:

$$\hat{\mu} \geq \frac{\lambda}{16\bar{\kappa}_{k:n}\rho}\frac{k}{n},$$

where $\bar{\kappa}_{k:n} \coloneqq \frac{1}{n-k}\sum_{i>k}\lambda_i(K_\lambda)/\lambda_{\min}(K_\lambda)$. The exact statement is presented in Theorem 13.

For clarity, we illustrate our argument to lower bound $\hat{\mu}$ in Fig. 2.

5.1 Approximate Ridge Leverage Score Sampling and DPPs

In this subsection, we formally introduce approximate ridge leverage scores and ARLS sampling and discuss their efficient computation. As our argument relies on a reduction from ARLS sampling to DPP sampling, we also define determinantal point processes and introduce some fundamental facts about them.

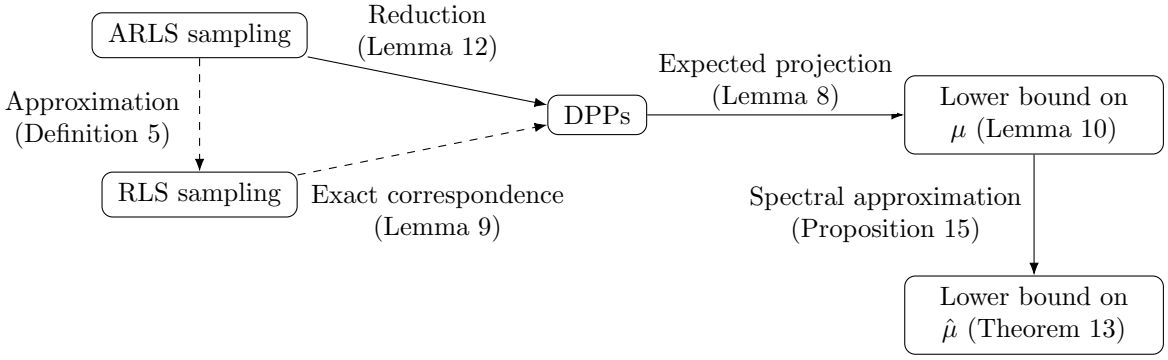

Figure 2: Summary of reductions and correspondences between ARLS/RLS sampling, DPPs, and the resulting lower bounds on μ and $\hat{\mu}$. The dashed arrows are for intuition only: ARLS is closely related to RLS and RLS is related to DPPs, so we expect ARLS to be related to DPPs. The solid arrows correspond to our approach for obtaining a lower bound on $\hat{\mu}$.

5.1.1 Approximate Ridge Leverage Score Sampling

We begin by introducing ridge leverage scores.

Definition 4 (Ridge leverage scores, effective dimension, and degrees of freedom) *Given $A \in \mathbb{S}_+^n$ and $\lambda > 0$, its i-th λ-ridge leverage score $\ell_i^\lambda(A)$ is the i-th diagonal entry of matrix $A(A+\lambda I)^{-1}$:*

$$\ell_i^\lambda(A) = e_i^T A(A+\lambda I)^{-1} e_i.$$

The sum of all its λ-ridge leverage scores is called its λ-effective dimension:

$$d^\lambda(A) \coloneqq \operatorname{Tr}\left(A(A+\lambda I)^{-1}\right) = \sum_{i=1}^n \ell_i^\lambda(A).$$

The λ-maximal degrees of freedom of A is given by:

$$d_{\max}^\lambda(A) \coloneqq n \max_{i\in[n]} \ell_i^\lambda(A).$$

The effective dimension measures the degrees of freedom of A, taking into account the regularization λ. For matrices with decaying eigenvalues (e.g., kernel matrices), $d^\lambda(A)$ is much smaller than the ambient dimension n (Alaoui and Mahoney, 2015). The i-th RLS captures how much row i of A contributes to the effective dimension. The maximal degrees of freedom is always larger than the effective dimension, and can be significantly larger when the leverage score distribution is highly non-uniform.

The RLS scores naturally define a distribution on the rows of A: sample row i with probability $p_i = \ell_i^\lambda(A)/d^\lambda(A)$. This sampling distribution admits strong theoretical guarantees (Alaoui and Mahoney, 2015; Rudi et al., 2018), but is impractical as computing the leverage scores requires an eigendecomposition at a cost of $\mathcal{O}(n^3)$, which is more expensive than the problem we wish to solve. Fortunately, good guarantees can still be obtained by using approximate ridge leverage score sampling (ARLS sampling), which we now define.

Definition 5 (Approximate RLSs and ARLS_c^λ-sampling) *Given $A \in \mathbb{S}_+^n$ and $\lambda > 0$, we say that $\{\tilde{\ell}_i\}_{i=1}^n$ are a c-approximation of λ-ridge leverage scores $\{\ell_i^\lambda(A)\}_{i=1}^n$ for some $c \geq 1$, if they satisfy*

$$\tilde{\ell}_i \geq \ell_i^\lambda(A) \quad \forall i \qquad \text{and} \qquad \tilde{\ell} := \sum_{i=1}^n \tilde{\ell}_i \leq c \cdot d^\lambda(A).$$

Define $p_i := \frac{\tilde{\ell}}{n}\left\lceil \frac{n}{\tilde{\ell}}\tilde{\ell}_i \right\rceil$ for all $i \in [n]$, and sample indices $\{i_j\}_{j=1}^b$ i.i.d. according to the probabilities $\{p_i/\|p\|_1\}_{i=1}^n$. Then $\mathcal{B} = \bigcup_{j=1}^b \{i_j\}$ is an ARLS_c^λ-sampling.[3]

Approximate RLSs were first introduced by Alaoui and Mahoney (2015) for constructing randomized Nyström approximations of kernel matrices. The key advantage of ARLSs over RLSs is that they can be computed efficiently. In particular, `ASkotch` uses the BLESS algorithm proposed by Rudi et al. (2018) to compute the ARLSs. BLESS admits the following complexity guarantee:

Lemma 6 (Rudi et al. (2018), Theorem 1) *Given $A \in \mathbb{S}_+^n$, a positive integer k, $c \geq 1$ and $\lambda > 0$ such that the λ-effective dimension of A satisfies $d^\lambda(A) \leq k$, there is an algorithm (BLESS) that with high probability returns c-approximations for all n λ-ridge leverage scores of A in $\tilde{\mathcal{O}}(nk^2)$ time.*

Lemma 6 shows we can obtain ARLSs to construct the sampling distribution in $\tilde{O}(nk^2)$ time. This stands in contrast to SAP solvers in Dereziński and Yang (2024); Dereziński et al. (2025a), which require $\mathcal{O}(n^2)$ time to preprocess K by a Hadamard transform H, and $\mathcal{O}(n^2)$ storage to store HKH^T, from which rows are subsampled. Thus, ARLSs yield a significant improvement in computational and storage complexities over Hadamard preprocessing.

5.1.2 Determinantal Point Processes and their Connection to RLSs

The heart of our analysis is a novel ARLS-to-DPP reduction argument that exploits a fundamental connection between RLS and determinantal point processes (DPPs). DPPs are a family of non-i.i.d. probability measures used in machine learning to sample diverse subsets of data points (Kulesza and Taskar, 2012).

Definition 7 (Determinantal point process) *Given $A \in \mathbb{S}_+^n$, a (random-size) determinantal point process $\mathrm{DPP}(A)$ is a distribution over all 2^n index subsets $\mathcal{B} \subseteq [n]$ such that $\Pr(\mathcal{B}) = \frac{\det(A_{\mathcal{B}\mathcal{B}})}{\det(A+I)}$. Moreover, a (fixed-size) k-$\mathrm{DPP}(A)$ is a distribution over subsets $\mathcal{B} \subseteq \binom{[n]}{k}$ such that $\Pr(\mathcal{B}) \propto \det(A_{\mathcal{B}\mathcal{B}})$.*

DPP-based row selection matrices yield strong convergence guarantees for SAP methods thanks to the following formula connecting random projectors with DPP sampling.

Lemma 8 (Dereziński et al. (2020), Lemma 5) *Given $A \in \mathbb{S}_+^n$, let $\mathcal{B} \sim \mathrm{DPP}(A)$, and define the random projection matrix $\Pi_{\mathcal{B}} := A^{1/2}S^T(SAS^T)^+SA^{1/2}$, where S is a row selection matrix for set $\mathcal{B}$. Then*

$$\mathbb{E}[\Pi_{\mathcal{B}}] = A(A+I)^{-1}. \tag{12}$$

3. The indices in $\mathcal{B}$ are actually distinct since our analysis allows us to discard duplicates.

Mutny et al. (2020) gives a sharp convergence analysis for DPP-based SAP methods using (12). Unfortunately, directly sampling from a DPP is impractical due to its prohibitive computational cost (Dereziński et al., 2019).

The goal of our ARLS-to-DPP reduction is to show that when using ARLS sampling, a relation similar to (12) holds:

$$\mathbb{E}[\Pi_{\mathcal{B}}] \succeq A(A+I)^{-1} - \delta A^{+}A.$$

That is, we want to obtain a perturbed version of the DPP expectation formula (12) for a small δ. Our arguments show that this is indeed the case. The key idea is to exploit the following connection between the RLSs of A and $\mathrm{DPP}(A)$:

Lemma 9 (Dereziński and Mahoney (2021), Theorem 10) *For $\mathcal{B} \sim \mathrm{DPP}(A)$ and index $i \in [n]$, the marginal probability of $i \in \mathcal{B}$ is the i-th ridge leverage score of A, i.e.,*

$$\Pr(i \in \mathcal{B}) = \left[A(A+I)^{-1}\right]_{i,i} = \ell_i^1(A).$$

5.2 Analysis of μ with ARLS Sampling

The main goal of this subsection is establish the following result, which provides tight control on $\mathbb{E}[\Pi_{\mathcal{B}}]$ and μ when using ARLS sampling.

Lemma 10 (Projection analysis for ARLS sampling) *Given $A \in \mathbb{S}_+^n$ with rank r, $k = \Omega(\log n)$, let $\bar{\lambda}, \tilde{\lambda} > 0$ be such that $d^{\tilde{\lambda}}(A) \geq 2d^{\bar{\lambda}}(A) = 4k$. If $\mathcal{B}$ is $\mathrm{ARLS}_c^{\tilde{\lambda}}$-sampled with blocksize $b = \Omega(ck\log^3 n)$, then the projection $\Pi_{\mathcal{B}} \coloneqq A^{1/2}I_{\mathcal{B}}^T(I_{\mathcal{B}}AI_{\mathcal{B}}^T)^{+}I_{\mathcal{B}}A^{1/2}$ satisfies*

$$\mathbb{E}[\Pi_{\mathcal{B}}] \;\succeq\; \frac{1}{2}A\left(A+\bar{\lambda}I\right)^{-1}. \tag{13}$$

Furthermore, this implies the following lower bound for the SAP convergence parameter μ:

$$\mu = \lambda_{\min}^{+}(\mathbb{E}[\Pi_{\mathcal{B}}]) \geq \frac{k}{2r\bar{\kappa}_{k:r}}, \quad \textit{where} \quad \bar{\kappa}_{k:r} \coloneqq \frac{1}{r-k}\sum_{i>k}\lambda_i(A)/\lambda_{\min}^{+}(A).$$

Lemma 10 gives a result similar to Lemma 8, only instead of exact equality as in (12), we obtain a lower bound. Nevertheless, this is sufficient for our needs, and allows us to obtain a lower bound on μ comparable to what is obtained in Dereziński and Yang (2024); Dereziński et al. (2025a) using Hadamard preprocessing. Thus, using ARLS sampling to select rows of A yields the same control over $\mathbb{E}[\Pi_{\mathcal{B}}]$, but at much lower computational and storage costs compared to SAP methods that use Hadamard preprocessing.

Lemma 10 immediately yields a projection analysis for uniform sampling, since uniform sampling corresponds to ARLS sampling with a specific choice of scores.

Corollary 11 (Projection analysis for uniform sampling) *Suppose A, $\bar{\lambda}$, $\tilde{\lambda}$, and k are set as in Lemma 10. If $\mathcal{B}$ is sampled according to the uniform distribution with blocksize $b = \Omega(d_{\max}^{\tilde{\lambda}}(A)\log^3 n)$, then $\Pi_{\mathcal{B}}$ satisfies*

$$\mathbb{E}[\Pi_{\mathcal{B}}] \;\succeq\; \frac{1}{2}A\left(A+\bar{\lambda}I\right)^{-1}. \tag{14}$$

Under uniform sampling, the blocksize has to be proportional to the maximal degrees of freedom: $b = \mathcal{O}(d^{\tilde{\lambda}}_{\max}(A)\log^3 n)$. When the ridge leverage scores are relatively uniform, i.e., $\max_{i\in[n]} \ell^{\tilde{\lambda}}_i(A) \approx \frac{d^{\tilde{\lambda}}(A)}{n}$, then $d^{\tilde{\lambda}}_{\max}(A) = \Theta(d^{\tilde{\lambda}}(A))$, so uniform sampling performs very well without requiring a large blocksize. As uniform sampling works well empirically across a broad range of learning tasks (Section 7), we believe this setting is most relevant for practice.

5.2.1 ARLS-to-DPP Reduction

To prove Lemma 10, we rely on connections between RLSs and DPPs. Recall from Lemma 8 that for any $A \in \mathbb{S}^n_+$ and $\bar{\lambda} > 0$, the sample $\mathcal{B}_{\text{DPP}} \sim \text{DPP}(A/\bar{\lambda})$ satisfies $\mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}}] = A(A+\bar{\lambda}I)^{-1}$. Moreover, from Lemma 9, the marginal probability of sampling element i into $\mathcal{B}$ is the i-th $\bar{\lambda}$-ridge leverage score of A, i.e., $\Pr(i \in \mathcal{B}_{\text{DPP}}) = \ell^{\bar{\lambda}}_i(A)$. We would like to exploit this connection to transfer the projection analysis from DPPs to ARLSs. Indeed, the following lemma shows that a sufficiently large ARLS sample contains a DPP sample, which is crucial for the proof of Lemma 10:

Lemma 12 (ARLS-to-DPP reduction) *Let $k, \tilde{\lambda}$, and $\mathcal{B}$ be defined as in Lemma 10. Given $\delta = n^{-\mathcal{O}(1)}$, for any $j \in \left[2k - \sqrt{6k\log\left(\frac{2}{\delta}\right)}, 2k + \sqrt{6k\log\left(\frac{2}{\delta}\right)}\right]$, we can couple a DPP sample $\mathcal{B}_{j\text{-DPP}} \sim j\text{-DPP}(A/\tilde{\lambda})$ with $\mathcal{B}$ such that with probability at least $1-\delta$, $\mathcal{B}_{j\text{-DPP}} \subseteq \mathcal{B}$.*

The proof of Lemma 12 is given in Appendix B.2.3. Our analysis is inspired by Dereziński and Yang (2024), who use a Markov Chain Monte Carlo (MCMC) argument (Anari et al., 2024) to show that after conditioning $\mathcal{B}_{\text{DPP}}$ on a fixed sample size j, the i.i.d. sampling distribution over the marginals $p_{i|j} = \Pr(i \in \mathcal{B}_{\text{DPP}} \mid |\mathcal{B}_{\text{DPP}}| = j)$ is an effective proxy for j-DPP$(A/\bar{\lambda})$. We modify the approach in Dereziński and Yang (2024) by showing that the approximate ridge leverage scores $\tilde{\ell}^{\bar{\lambda}}_i(A)$ are close to the marginals $p_{i|j}$ as long as the size j is close to the expected size of $\mathcal{B}_{\text{DPP}}$, $\mathbb{E}[|\mathcal{B}_{\text{DPP}}|]$. Fortunately, this occurs with high probability (Lemma 21). We then use the law of total expectation to decompose the distribution of DPP$(A/\bar{\lambda})$ into a mixture of fixed-size DPP distributions j-DPP(A), and apply an MCMC argument to each fixed-size DPP. Combining all of these ideas yields the result.

5.2.2 Proof of Lemma 10

To complete the proof of Lemma 10, we leverage the ARLS-to-DPP reduction from Lemma 12 and apply the DPP projection formula from Lemma 8.

Proof Define the interval $\mathcal{I} = \left[2k - \sqrt{6k\log\left(\frac{2}{\delta}\right)}, 2k + \sqrt{6k\log\left(\frac{2}{\delta}\right)}\right]$ where $k = d^{\bar{\lambda}}(A)/2$, and denote event $\mathcal{E}_j = \{\mathcal{B}_{j\text{-DPP}} \subseteq \mathcal{B}\}$. According to Lemma 12, for any $j \in \mathcal{I}$ we have $\Pr(\mathcal{E}_j) \geq 1-\delta$. By using the fact that $\Pi_{\mathcal{B}_1} \preceq \Pi_{\mathcal{B}_2}$ whenever $\mathcal{B}_1 \subseteq \mathcal{B}_2$ (Dereziński and Yang, 2024, Lemma 6.3), we have

$$\begin{aligned}
\mathbb{E}[\Pi_{\mathcal{B}}] &\succeq \mathbb{E}[\Pi_{\mathcal{B}} \mid \mathcal{E}_j]\Pr(\mathcal{E}_j) \\
&\succeq \mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}} \mid \mathcal{E}_j]\Pr(\mathcal{E}_j) \\
&= \mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}}] - \mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}} \mid \mathcal{E}^c_j]\Pr(\mathcal{E}^c_j) \\
&\succeq \mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}}] - \delta \cdot A^+A, \qquad (15)
\end{aligned}$$

where the last step follows since $\Pi_{\mathcal{B}_{j\text{-DPP}}}$ is an orthogonal projector onto a subspace of range(A^T), thus $\mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}} \mid \mathcal{E}_j^c] \preceq \Pi_{[n]} = A^+A$. Let $\mathcal{B}_{\text{DPP}} \sim \text{DPP}(A/\bar{\lambda})$ and recall that $\mathbb{E}[|\mathcal{B}_{\text{DPP}}|] = d^{\bar{\lambda}}(A) = 2k$. Also, define $w_{\bar{\lambda},j} := \Pr(|\mathcal{B}_{\text{DPP}}| = j)$. Using the law of total expectation, we can express the expected projection matrix for a DPP in terms of fixed-size j-DPPs:

$$\begin{aligned}\mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}}] &= \sum_{j\in\mathcal{I}} w_{\bar{\lambda},j}\mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}} \mid |\mathcal{B}_{\text{DPP}}| = j] + \sum_{j\notin\mathcal{I}} w_{\bar{\lambda},j}\mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}} \mid |\mathcal{B}_{\text{DPP}}| = j] \\ &= \sum_{j\in\mathcal{I}} w_{\bar{\lambda},j}\mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}}] + \sum_{j\notin\mathcal{I}} w_{\bar{\lambda},j}\mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}}],\end{aligned}$$

where the second equality uses that $\mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}} \mid |\mathcal{B}_{\text{DPP}}| = j] = \mathbb{E}[\Pi_{\mathcal{B}_{j\text{-DPP}}}]$ by definition. Recalling (15) and using $\Pi_{\mathcal{B}_{j\text{-DPP}}} \preceq \Pi_{[n]} = A^+A$, we can bound the preceding display as

$$\begin{aligned}\mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}}] &\preceq \left(\sum_{j\in\mathcal{I}} w_{\bar{\lambda},j}\right)\left(\mathbb{E}[\Pi_{\mathcal{B}}] + \delta\cdot A^+A\right) + \left(\sum_{j\notin\mathcal{I}} w_{\bar{\lambda},j}\right)A^+A \\ &\preceq \mathbb{E}[\Pi_{\mathcal{B}}] + \delta\cdot A^+A + \left(\sum_{j\notin\mathcal{I}} w_{\bar{\lambda},j}\right)A^+A,\end{aligned}$$

where the last relation follows as $\sum_{j\in\mathcal{I}} w_{\bar{\lambda},j} \le 1$. Now, as $\Pr(j\notin\mathcal{I}) = \sum_{j\notin\mathcal{I}} w_{\bar{\lambda},j}$, Lemma 21 yields $\sum_{j\notin\mathcal{I}} w_{\bar{\lambda},j} \le \delta$. Combining this with the preceding display, we deduce

$$\mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}}] \preceq \mathbb{E}[\Pi_{\mathcal{B}}] + 2\delta A^+A.$$

Rearranging the preceding display yields

$$\mathbb{E}[\Pi_{\mathcal{B}}] \succeq \mathbb{E}[\Pi_{\mathcal{B}_{\text{DPP}}}] - 2\delta A^+A.$$

Applying Lemma 8 and setting $\delta \le \min\{\lambda_{\min}^+(A)/(4(\lambda_{\min}^+(A)+\bar{\lambda})), n^{-c}\}$ for some constant $c > 0$, so that $\delta = n^{-\mathcal{O}(1)}$, we conclude

$$\mathbb{E}[\Pi_{\mathcal{B}}] \succeq \frac{1}{2}A\left(A+\bar{\lambda}I\right)^{-1}, \tag{$*$}$$

which is precisely (13).

We now establish the lower bound on μ. To this end, we first show that $\bar{\lambda} \le \frac{1}{k}\sum_{i>k}\lambda_i(A)$. Indeed, if we suppose $\bar{\lambda} > \frac{1}{k}\sum_{i>k}\lambda_i(A)$, then

$$2k \le d^{\bar{\lambda}}(A) = \sum_{i=1}^{k}\frac{\lambda_i}{\lambda_i+\bar{\lambda}} + \sum_{i>k}\frac{\lambda_i}{\lambda_i+\bar{\lambda}} < k + \frac{\sum_{i>k}\lambda_i}{\bar{\lambda}} < k+k = 2k,$$

resulting in a contradiction. Hence $\bar{\lambda} \le \frac{1}{k}\sum_{i>k}\lambda_i(A)$.

Finally, it follows from $(*)$ that

$$\mu = \lambda_{\min}^{+}\left(\mathbb{E}[\Pi_{\mathcal{B}}]\right) \geq \frac{1}{2}\lambda_{\min}^{+}\left(A\left(A+\bar{\lambda}I\right)^{-1}\right) = \frac{1}{2}\frac{\lambda_{\min}^{+}(A)}{\lambda_{\min}^{+}(A)+\bar{\lambda}} \geq \frac{1}{2\left(1+\frac{1}{k}\sum_{i>k}\frac{\lambda_i(A)}{\lambda_{\min}^{+}(A)}\right)}$$
$$= \frac{1}{2\left(1+\frac{r-k}{k}\frac{1}{r-k}\sum_{i>k}\frac{\lambda_i(A)}{\lambda_{\min}^{+}(A)}\right)} \geq \frac{k}{2\left(\frac{1}{r-k}\sum_{i>k}\frac{\lambda_i(A)}{\lambda_{\min}^{+}(A)}\right)r} = \frac{k}{2\bar{\kappa}_{k:r}(A)r}.$$

■

5.3 Lower Bound on $\hat{\mu}$

Having established a lower bound on μ, we now turn to lower bounding $\hat{\mu}$, which will allow us to establish a fine-grained convergence rate for `ASkotch`.

The main goal of this subsection is to establish the following theorem:

Theorem 13 (Lower bound on $\hat{\mu}$) *Suppose the blocksize satisfies $b = \Omega(k\log^3 n)$ for some $k \in [n]$. Then using ARLS sampling as in Lemma 10 with $A = K_\lambda$ and $c = \mathcal{O}(1)$, the Nyström approximation as in Proposition 15, and defining $\bar{\lambda}$ as in Lemma 10,*

$$\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}] \succeq \frac{\lambda}{8\rho}K_\lambda(K_\lambda+\bar{\lambda}I)^{-1}.$$

Consequently, recalling $\bar{\kappa}_{k:n} = \frac{1}{n-k}\sum_{i>k}\lambda_i(K_\lambda)/\lambda_{\min}(K_\lambda)$, we have

$$\hat{\mu} \geq \frac{\lambda}{16\bar{\kappa}_{k:n}\rho}\frac{k}{n}.$$

Theorem 13 provides us with the desired lower bound on $\hat{\mu}$. When b is sufficiently large, the following corollary shows that $\hat{\mu}$ is lower bounded by a quantity independent of the conditioning of K_λ.

Corollary 14 *Fix a failure probability $\delta = n^{-\mathcal{O}(1)}$. Suppose the blocksize satisfies $b = \Omega\left(k\log^3 n\right)$ where $k \geq 2d^\lambda(K)$. If the rank of the Nyström approximation satisfies $r = \mathcal{O}\left(d^\rho(\lfloor K\rfloor_b)\log\left(\frac{d^\rho(\lfloor K\rfloor_b)}{\delta}\right)\right)$ and $\rho \geq \lambda$, then*

$$\hat{\mu} \geq \frac{\lambda}{32\rho}\frac{k}{n}.$$

When the effective blocksize $k = \Omega(d^\lambda(K))$, Corollary 14 shows the lower bound on $\hat{\mu}$ no longer depends on the conditioning of K_λ. Instead, $\hat{\mu}$ is controlled by k and ρ. Using a smaller ρ (larger r) leads to a larger $\hat{\mu}$. Conversely, larger ρ corresponds to using a smaller rank, and the lower bound on $\hat{\mu}$ decreases. We shall see shortly that Corollary 14 implies `Skotch` and `ASkotch` enjoy condition-number-free convergence rates.

5.3.1 The Nyström Projector is Close to the SAP Projector

Lemma 10 guarantees a tight lower bound on μ when $\mathcal{B}$ is an $\text{ARLS}_c^{\tilde{\lambda}}$-sampling for any $c = \mathcal{O}(1)$ with respect to K_λ. Thus, if we can determine the form of $g(\rho, \lambda)$, we can obtain a lower bound on $\hat{\mu}$.

We begin with Proposition 15, which explicitly characterizes the *shrinkage factor* $\sigma_{P_\mathcal{B}}/\hat{L}_{P_\mathcal{B}}$ in Lemma 3 with high probability. The proof of Proposition 15 appears in Appendix B.4.

Proposition 15 (Controlling the shrinkage factor) *Let $\mathcal{B} \subseteq [n]$ satisfy $|\mathcal{B}| = b < n$, and set $\rho \geq \lambda$. If the Nyström approximation $\hat{K}_{\mathcal{B}\mathcal{B}}$ of $K_{\mathcal{B}\mathcal{B}}$ is constructed from a sparse sign embedding Ω with $r = \mathcal{O}\left(d^\rho(\lfloor K \rfloor_b) \log\left(\frac{d^\rho(\lfloor K \rfloor_b)}{\delta}\right)\right)$ columns and $\zeta = \mathcal{O}\left(\log\left(\frac{d^\rho(\lfloor K \rfloor_b)}{\delta}\right)\right)$ non-zeros per column, then*

$$\frac{\lambda}{2\rho}\Pi_\mathcal{B} \preceq \hat{\Pi}_{\mathcal{B},\rho} \preceq \Pi_\mathcal{B}, \quad \textit{with probability at least } 1-\delta.$$

The shrinkage factor obtained in Proposition 15 depends upon λ/ρ. If $\rho = \mathcal{O}(\lambda)$, then the shrinkage factor is *constant* with high probability, in which case little is lost by replacing $K_{\mathcal{B}\mathcal{B}}$ with the Nyström approximation $\hat{K}_{\mathcal{B}\mathcal{B}}$. Selecting a larger ρ allows smaller Nyström rank r, but incurs a trade-off in the shrinkage factor. We note that $d^\rho(\lfloor K \rfloor_b)$ is always upper-bounded by $d^\rho(K)$, and in some cases, may be substantially smaller.

Remark 16 *While the guarantee in Proposition 15 is for when $\hat{K}_{\mathcal{B}\mathcal{B}}$ is constructed with a sparse sign embedding, the same guarantee also holds when Ω is a Gaussian embedding (which we use in our experiments) or a randomized Hadamard transform.*

5.3.2 Proof of Theorem 13

Here we prove Theorem 13, which provides a lower bound on $\hat{\mu}$.

Proof Define the event

$$\mathcal{E} = \left\{\frac{\lambda}{2\rho}\Pi_\mathcal{B} \preceq \hat{\Pi}_{\mathcal{B},\rho} \preceq \Pi_\mathcal{B}\right\}.$$

Then Proposition 15 tells us that by setting δ appropriately, we can guarantee $\Pr(\mathcal{E}) \geq 1 - \frac{\mu}{2}$. Thus, the law of total expectation yields

$$\begin{aligned}
\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}] &= \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho} \mid \mathcal{E}] \Pr(\mathcal{E}) + \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho} \mid \mathcal{E}^C] \Pr(\mathcal{E}^C) \\
&\succeq \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho} \mid \mathcal{E}] \Pr(\mathcal{E}) \\
&\succeq \frac{\lambda}{2\rho}\mathbb{E}\left[\Pi_\mathcal{B} \mid \mathcal{E}\right] \Pr(\mathcal{E}). \qquad (*)
\end{aligned}$$

We now lower bound $\mathbb{E}\left[\Pi_\mathcal{B} \mid \mathcal{E}\right]$ in the Loewner ordering. To this end, the law of total expectation and a simple rearrangement imply

$$\mathbb{E}\left[\Pi_\mathcal{B} \mid \mathcal{E}\right] = \frac{1}{\Pr(\mathcal{E})}\left(\mathbb{E}[\Pi_\mathcal{B}] - \mathbb{E}[\Pi_\mathcal{B} \mid \mathcal{E}^C] \Pr(\mathcal{E}^C)\right).$$

The preceding display, along with the facts that $\Pi_\mathcal{B} \preceq I$ and $\Pr(\mathcal{E}^C) \leq \mu/2$, yields

$$\mathbb{E}\left[\Pi_\mathcal{B} \mid \mathcal{E}\right] \succeq \frac{1}{\Pr(\mathcal{E})}\left(\mathbb{E}[\Pi_\mathcal{B}] - \Pr(\mathcal{E}^C) I\right) \succeq \frac{1}{\Pr(\mathcal{E})}\left(\mathbb{E}[\Pi_\mathcal{B}] - \frac{\mu}{2} I\right).$$

As $\frac{\mu}{2} I \preceq \frac{1}{2}\mathbb{E}[\Pi_{\mathcal{B}}]$, we obtain

$$\mathbb{E}\left[\Pi_{\mathcal{B}} \mid \mathcal{E}\right] \succeq \frac{1}{2\Pr(\mathcal{E})}\mathbb{E}[\Pi_{\mathcal{B}}].$$

Thus, combining this last display with $(*)$ and applying Lemma 10 with $A = K_\lambda$, we conclude

$$\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}] \succeq \frac{\lambda}{4\rho}\mathbb{E}[\Pi_{\mathcal{B}}] \succeq \frac{\lambda}{8\rho} K_\lambda \left(K_\lambda + \bar{\lambda} I\right)^{-1}.$$

The claimed lower bound on $\hat{\mu}$ follows from this last display and Lemma 10. ■

6. Convergence of ASkotch

We have established all the preliminary results for the convergence analysis of ASkotch. We begin with the following general convergence result (Theorem 17). In Section 6.1.1, we discuss when Skotch and ASkotch achieve near-optimal log-linear runtimes and how they can use large blocksizes to leverage the benefits offered by modern computing hardware.

Theorem 17 (Convergence of ASkotch) *Consider Skotch and ASkotch (Algorithm 1) under the following hyperparameter settings: the blocksize satisfies $b = \Omega(k \log^3 n)$, where $k \geq 2d^\lambda(K)$, $\rho \geq \lambda$, $\mathcal{P}$ is a $\mathrm{ARLS}^{\tilde{\lambda}}_{\mathcal{O}(1)}$-sampling distribution for K_λ with $\tilde{\lambda} > 0$ such that $d^{\tilde{\lambda}}(K_\lambda) \geq 4k$, $\hat{K}_{\mathcal{B}\mathcal{B}}$ is constructed from a sparse sign embedding with $r = \mathcal{O}\left(d^\rho(\lfloor K \rfloor_b) \log\left(\frac{d^\rho(\lfloor K \rfloor_b)}{\delta}\right)\right)$ columns and $\zeta = \mathcal{O}\left(\log\left(\frac{d^\rho(\lfloor K \rfloor_b)}{\delta}\right)\right)$ non-zeros per column, and $\eta_{\mathcal{B}} = 1/\hat{L}_{P_{\mathcal{B}}}$. Then the following statements hold:*

1. *After t iterations, the output of Skotch satisfies*

$$\mathbb{E}[\|w_t - w_\star\|^2_{K_\lambda}] \leq \left(1 - \frac{\lambda}{32\rho}\frac{k}{n}\right)^t \|w_0 - w_\star\|^2_{K_\lambda}.$$

   Thus, the total number of iterations required to achieve an ϵ-approximate solution is bounded by $\mathcal{O}\left(\frac{\rho}{\lambda}\frac{n}{k}\log\left(\frac{1}{\epsilon}\right)\right)$.

2. *After t iterations, the output of ASkotch satisfies*

$$\mathbb{E}[\|w_t - w_\star\|^2_{K_\lambda}] \leq 2\left(1 - \frac{1}{16\sqrt{2}}\min\left\{\sqrt{\frac{\lambda}{\rho}}\frac{k}{n}, \sqrt{\frac{k}{n}}\frac{\lambda}{\rho}\right\}\right)^t \|w_0 - w_\star\|^2_{K_\lambda}.$$

   Thus, the total number of iterations required to achieve an ϵ-approximate solution is bounded by $\mathcal{O}\left(\max\left\{\sqrt{\frac{\rho}{\lambda}}\frac{n}{k}, \frac{\rho}{\lambda}\sqrt{\frac{n}{k}}\right\}\log\left(\frac{1}{\epsilon}\right)\right)$.

The proof of Theorem 17 is provided in Section 6.2.

Theorem 17 shows Skotch and ASkotch converge linearly to the solution of (3). Skotch requires $\mathcal{O}(\frac{\rho}{\lambda}\frac{n}{k}\log\left(\frac{1}{\epsilon}\right))$ iterations to produce an ϵ-approximate solution, while ASkotch requires $\mathcal{O}\left(\max\left\{\sqrt{\frac{\rho}{\lambda}}\frac{n}{k}, \frac{\rho}{\lambda}\sqrt{\frac{n}{k}}\right\}\log\left(\frac{1}{\epsilon}\right)\right)$ iterations. As $\rho/\lambda, n/k \geq 1$, the upper bound for ASkotch is always an improvement over the upper bound for Skotch, demonstrating the benefit of acceleration.

6.1 ASkotch: Two Convergence Regimes

Theorem 17 shows that ASkotch's convergence exhibits two different regimes: (i) $n/k > \rho/\lambda$, (ii) $\rho/\lambda > n/k$. In regime (i), the "low-rank approximation condition number" ρ/λ is dominated by the "dimensional condition number" n/k, and ASkotch converges in $\tilde{\mathcal{O}}\left(\sqrt{\frac{\rho}{\lambda}\frac{n}{k}}\right)$ iterations. In this regime, ASkotch can use a larger ρ (and hence, a smaller Nyström rank r) than Skotch, as the convergence rate depends only upon $\sqrt{\frac{\rho}{\lambda}}$. However, if ρ is set too large, so that $\rho > (n/k)\lambda$, ASkotch enters regime (ii), where the low-rank approximation condition number ρ/λ dominates the dimensional condition number n/k, leading to an iteration complexity of $\tilde{\mathcal{O}}\left(\frac{\rho}{\lambda}\sqrt{\frac{n}{k}}\right)$.

Which convergence regime of ASkotch is preferable? In the worst case, a n/k dependence in the iteration complexity is necessary, since ASkotch must make a full pass through K_λ to solve (3). Thus, regime (i) is preferable as it allows ASkotch to reduce the price ρ/λ of using a low-rank approximation. ASkotch is guaranteed to be in regime (i) provided it does not over-regularize, since the phase transition to regime (ii) occurs when $\rho > (n/k)\lambda$.

When the effective blocksize k is small, i.e., $k = o(n)$, n/k is large, so ASkotch can still use large ρ and small rank r while avoiding over-regularization. When the effective blocksize is large, i.e., $k = \Omega(n)$, $n/k = \mathcal{O}(1)$, so ASkotch must set $\rho = \mathcal{O}(\lambda)$ to stay in regime (i). Consequently, if K exhibits heavy-tailed spectral decay, the rank r might have to increase significantly for ASkotch to remain in regime (i). Fortunately, most kernel matrices exhibit fast spectral decay—indeed, under standard regularity assumptions on the kernel function, $d^\lambda(K) = \mathcal{O}(\sqrt{n})$ (Rudi et al., 2017, 2018). Thus, even if $k = \Omega(n)$, ASkotch remains in regime (i) when it uses rank $r = \mathcal{O}(\sqrt{n})$. Overall, convergence regime (i) for ASkotch is both preferable and also more likely to occur, compared to regime (ii).

6.1.1 When Does ASkotch Converge in Log-Linear Time?

A natural question to ask is if there is a reasonable setting where Skotch and ASkotch enjoy a near-optimal runtime of $\tilde{\mathcal{O}}(n^2)$. A priori, the answer is not obvious, as we use a low-rank approximation to $K_{\mathcal{BB}}$. Consequently, the price of using this approximation may be too steep to ensure an $\tilde{\mathcal{O}}(n^2)$ runtime. Fortunately, the following corollary shows that for matrices whose effective dimension is not too large, the low-rank approximation has a minimal effect on the overall runtime. As the effective dimension of kernel matrices is typically $\mathcal{O}(\sqrt{n})$ (Rudi et al., 2017, 2018), this corollary shows that Skotch and ASkotch can achieve a near-optimal runtime for most KRR tasks, despite approximating $K_{\mathcal{BB}}$.

Corollary 18 (Log-linear convergence of ASkotch) *Let $\mathcal{P}$ in ASkotch be the uniform distribution. Let k and $\tilde{\lambda}$ be as in Lemma 10 and suppose that $\max_{i\in[n]} \ell_i^{\tilde{\lambda}}(K_\lambda) = \Theta(d^{\tilde{\lambda}}(K_\lambda)/n)$, $\rho = c_\rho\lambda$ for $c_\rho \in [1, n/k]$, and $d^\lambda(K) = \mathcal{O}(\sqrt{n})$. Then under the assumptions of Theorem 17, with blocksize $b = \Theta(k\log^3 n)$, the total runtime of ASkotch to produce an ϵ-approximate solution is bounded by*

$$\tilde{\mathcal{O}}\left(\sqrt{c_\rho}n^2\log\left(\frac{1}{\epsilon}\right)\right).$$

The proof of Corollary 18 appears in Appendix B.6. Corollary 18 shows that when (i) the effective dimension grows like $\mathcal{O}(\sqrt{n})$, (ii) the ridge leverage scores are relatively uniform,

and (iii) `ASkotch` uses uniform sampling with $\rho = \mathcal{O}(\lambda)$, `ASkotch` obtains a near-optimal runtime.

Another powerful consequence of Corollary 18 is that the total runtime is independent of the blocksize. `ASkotch` can use effective blocksize $k = \mathcal{O}(n)$ (which corresponds to $b = \tilde{\mathcal{O}}(n)$) and still have runtime $\tilde{\mathcal{O}}\left(n^2\right)$. Thus, when the leverage scores are relatively uniform, `ASkotch` can use large blocksizes and obtain a log-linear runtime. From a practical perspective, this is invaluable, as large blocksizes can exploit the massive parallelism offered by modern computing architectures. Indeed, in our experiments, we use $b = n/100$ and observe excellent performance. The ability of `ASkotch` to use a large blocksize separates it from SAP, which incurs an $\mathcal{O}(n^3)$ cost per iteration when $b = \mathcal{O}(n)$.

Remark 19 *The same runtime bounds hold for ARLS sampling, which allows us to avoid the assumption that the ridge leverage scores satisfy* $\max_{i\in[n]} \ell_i^{\tilde{\lambda}}(K_\lambda) = \Theta\left(\frac{d^{\tilde{\lambda}}(K_\lambda)}{n}\right)$. *Instead, we require that* $k = \Theta(d^{\lambda}(K))$, *since this ensures that the cost of running BLESS is bounded by* $\tilde{\mathcal{O}}(n^2)$. *Thus, ARLS cannot use blocksizes as large as uniform sampling while maintaining near-optimal runtime. We focus on uniform sampling, as this is what works best in practice.*

6.2 Proof of Theorem 17

Here, we provide the convergence proof of `Skotch`, which corresponds to the first claim in Theorem 17. The convergence proof for `ASkotch` is more involved and requires additional background, so it is deferred to Appendix B.5.

Proof We have seen in Section 4.2 that at iteration i,

$$\mathbb{E}\left[\|w_i - w_\star\|_{K_\lambda}^2 \mid w_{i-1}\right] \leq (1-\hat{\mu})\|w_{i-1} - w_\star\|_{K_\lambda}^2.$$

Invoking the lower bound on $\hat{\mu}$ in Corollary 14, this becomes

$$\mathbb{E}\left[\|w_i - w_\star\|_{K_\lambda}^2 \mid w_{i-1}\right] \leq \left(1 - \frac{\lambda k}{32\rho n}\right)\|w_{i-1} - w_\star\|_{K_\lambda}^2.$$

Applying the law of total expectation in this last display, we deduce

$$\mathbb{E}\left[\|w_t - w_\star\|_{K_\lambda}^2\right] \leq \left(1 - \frac{\lambda k}{32\rho n}\right)^t \|w_0 - w_\star\|_{K_\lambda}^2.$$

The claimed iteration complexity bound follows immediately. ■

7. Experiments

We perform an empirical evaluation of `ASkotch`, EigenPro 2.0, EigenPro 3.0, PCG, and Falkon on KRR problems drawn from diverse application domains, including computer vision, particle physics, ecological modeling, online advertising, computational chemistry, music, socioeconomics, and transportation. We use three different kernels in our experiments: Laplacian, Matérn-5/2, and radial basis function (RBF), which are among the most

popular in the literature (Rasmussen and Williams, 2005; Gardner et al., 2018). Across this broad spectrum of application domains and kernels, `ASkotch` obtains better predictive performance than the competing methods. These results show that (i) full KRR can effectively scale to large problems and obtain better predictive performance than inducing points KRR and (ii) `ASkotch` is a new state-of-the-art solver for large-scale full KRR.

We present the following results:

- Performance comparisons (Section 7.1): We compare `ASkotch` to the state-of-the-art methods for full KRR and inducing points KRR: EigenPro 2.0 (Ma and Belkin, 2019) and PCG for full KRR, and Falkon (Rudi et al., 2017; Meanti et al., 2020) and EigenPro 3.0 (Abedsoltan et al., 2023) for inducing points KRR. For PCG, we consider two of the most effective types of preconditioners: Gaussian Nyström (Nyström) (Frangella et al., 2023) and randomly pivoted Cholesky (RPC) (Díaz et al., 2023; Epperly et al., 2025). `ASkotch` consistently obtains better predictive performance than the competitors.

- Showcase on huge-scale transportation data analysis (Section 7.2): We run `ASkotch` on KRR for a subsample of the New York City taxi dataset ($n = 10^8$). To the best of our knowledge, no previous work has scaled full KRR to a dataset of this size. `ASkotch` once again outperforms the competition.

- Verification of linear convergence (Section 7.3): We show that `ASkotch` achieves global linear convergence on three KRR problems with $n \geq 5 \cdot 10^5$ samples, which verifies the linear convergence guarantee in Section 6. `ASkotch` reaches machine precision in at most 100 passes through each KRR linear system, demonstrating its potential as a high-precision solver.

Each experiment is run on a single 48 GB NVIDIA RTX A6000 GPU with PyTorch 2.5.1 (Paszke et al., 2019), CUDA 12.5, PyKeOps 2.2.3 (Charlier et al., 2021), and Python 3.10.12. The code for our experiments is available at https://github.com/pratikrathore8/fast_krr.

Our arXiv report (https://arxiv.org/abs/2407.10070v3) includes expanded experimental results and an ablation study that are omitted from this version of the paper. We also provide additional information about datasets, kernel hyperparameters, regularizations, inducing points, preprocessing, and time limits in Appendix C of our arXiv report.

Optimizer hyperparameters. Throughout the experiments, we use the default hyperparameters that we recommend for `ASkotch` (Section 2.5), unless stated otherwise. We compute the damping for PCG with the Gaussian Nyström preconditioner using the same procedure as `ASkotch`, ensuring a fair comparison. We also run PCG, EigenPro 2.0, and EigenPro 3.0 with the same rank as `ASkotch` to ensure a fair comparison. We set b_g, s, and the stepsize η in EigenPro 2.0 and EigenPro 3.0 using the defaults in the EigenPro 2.0 and EigenPro 3.0 GitHub repositories. We also run EigenPro 2.0 and 3.0 with regularization $\lambda = 0$, as recommended in Ma and Belkin (2019); Abedsoltan et al. (2023).

Numerical precision. `ASkotch`, EigenPro 2.0, and EigenPro 3.0 are stable in single precision, while Falkon and PCG often require double precision for satisfactory performance. Consequently, the figures in the main paper show results when `ASkotch`, EigenPro 2.0, and EigenPro 3.0 run in single precision and Falkon and PCG run in double precision. We show

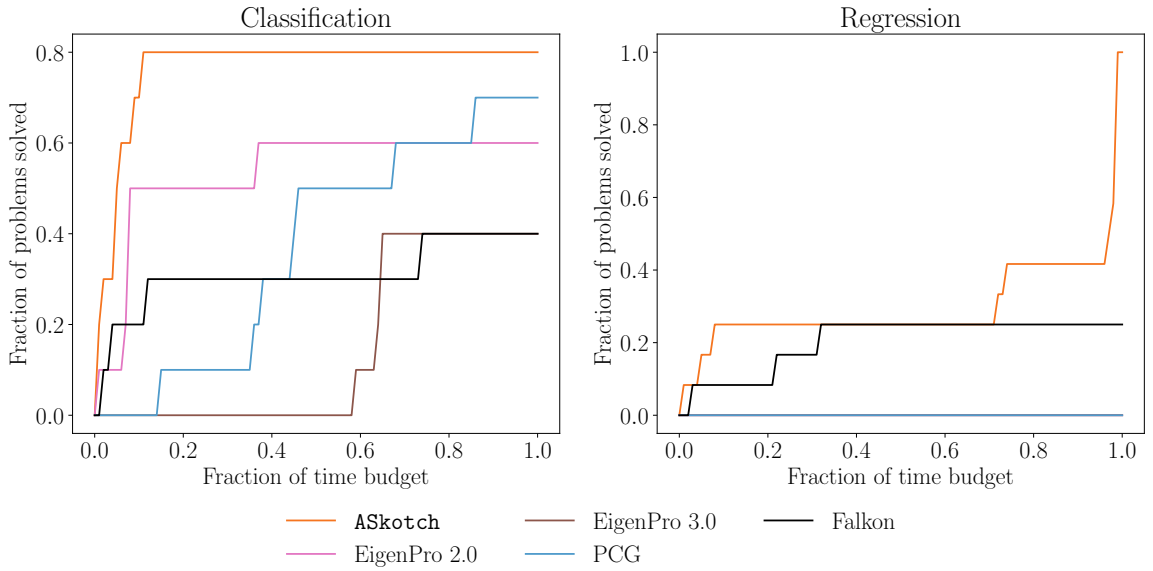

Figure 3: Performance comparison between `ASkotch` and competitors on 10 classification and 13 regression tasks. We designate a classification problem as "solved" when the method reaches within 0.001 of the highest classification accuracy found across all the optimizer + hyperparameter combinations. We designate a regression problem as "solved" when the method reaches within 1% of the lowest MAE (in a relative sense) found across all the optimizer + hyperparameter combinations. PCG and Falkon are run in double precision. EigenPro 2.0, EigenPro 3.0, and PCG do not solve any of the regression problems within the tolerance. `ASkotch` outperforms the competition on both classification and regression.

in Appendix C of our arXiv report that `ASkotch` still outperforms Falkon and PCG when all methods are run in single precision.

Time limits. Each optimizer is run until it reaches a specified time limit, which depends on the size of the dataset. These time limits range between 0.5 to 24 hours. We implement all of the competing methods ourselves to ensure fair runtime comparisons with `ASkotch`. The most expensive computations in all methods are performed using KeOps (Charlier et al., 2021), which ensures our runtime comparisons are fair.

7.1 Performance Comparisons

We evaluate predictive performance on 10 classification and 13 regression tasks from established benchmarks. The classification tasks are from computer vision, particle physics, ecological modeling, and online advertising, while the regression tasks are from computational chemistry, music analysis, and socioeconomics. We assess predictive performance on classification and regression tasks by computing classification accuracy and mean-absolute error (MAE) between the predictions and targets, respectively, on the test set.

We summarize the results of our performance comparisons in Fig. 3. `ASkotch` outperforms the competition on both classification and regression. Notably, `ASkotch` outperforms

the state-of-the-art inducing point methods EigenPro 3.0 and Falkon, demonstrating the value of using full KRR over inducing points KRR.

7.2 Showcase: Huge-Scale Transportation Data Analysis

We apply KRR to a subsample of the taxi dataset to predict taxi ride durations in New York City. Following Meanti et al. (2020), we use an RBF kernel. Due to hardware limitations, we set the blocksize b in `ASkotch` at $n/2{,}000 = 5 \cdot 10^4$ and vary the rank $r \in \{50, 100, 200, 500\}$. We set the rank for PCG as low as $r = 50$, but none of the PCG methods complete a single iteration in the time limit. Following Meanti et al. (2020), we use the root mean square error (RMSE) between the predictions and targets.

The results are shown in Fig. 1. `ASkotch` outperforms Falkon for each value of r. Both EigenPro 2.0 and EigenPro 3.0 (not shown) diverge. Once again, our findings demonstrate the value of using full KRR over inducing points KRR for large-scale regression tasks.

7.3 `ASkotch` Converges Linearly to the Optimum

We demonstrate that `ASkotch` obtains linear convergence to the optimum for large-scale full KRR. To do so, we plot the relative residual

$$\|K_\lambda w - y\|/\|y\|$$

obtained by `ASkotch`. We run `ASkotch` in double precision.

Fig. 4 shows the relative residual obtained by `ASkotch` on three large-scale KRR problems. `ASkotch` converges linearly for all selections of the rank r. Excitingly, `ASkotch` reaches machine precision on all three problems, showing its potential as a high-precision linear system solver.

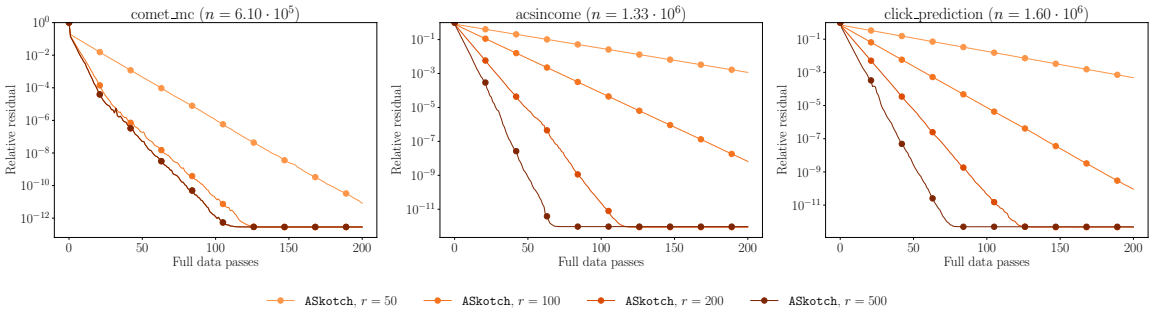

Figure 4: `ASkotch` converges linearly on large-scale full KRR. "Full data passes" indicates the number of passes through K_λ: since $b = n/100$, one full data pass is equivalent to 100 iterations of `ASkotch`. `ASkotch` tends to converge faster as the rank r increases, which matches our theoretical results. Interestingly, $r = 200$ and $r = 500$ yield similar results on comet_mc.

8. Conclusion

We introduce `ASkotch`, an approximate sketch-and-project method for large-scale full KRR. Our theoretical analysis and experiments demonstrate that `ASkotch` is a promising replacement for existing KRR solvers. Looking ahead, we aim to develop methods for automatically selecting the acceleration parameters $\hat{\mu}$ and $\hat{\nu}$ in `ASkotch`, create a distributed implementation of `ASkotch` that scales to datasets with $n \gtrsim 10^9$ training points (such as the full taxi dataset), and develop a mixed-precision version of `ASkotch` that delivers high-quality results with less memory and compute.

Our work, along with previous studies (Wang et al., 2019; Frangella et al., 2023; Díaz et al., 2023), shows that full KRR allows better predictive performance than inducing points KRR. `ASkotch` scales full KRR to datasets orders of magnitude larger than previously possible. In future work, we look forward to seeing how a fast solver for full KRR enables new applications. Moreover, following our methodology, it should be possible to solve general pd linear systems within the (approximate) sketch-and-project framework.

Acknowledgments

We thank Rob Webber for helpful discussions.

PR, ZF, and MU gratefully acknowledge support from the National Science Foundation (NSF) Award IIS-2233762, the Office of Naval Research (ONR) Awards N000142212825, N000142412306, and N000142312203, the Alfred P. Sloan Foundation, and from IBM Research as a founding member of Stanford Institute for Human-centered Artificial Intelligence (HAI). MD and JY gratefully acknowledge support from NSF Award CCF-2338655.

Appendix A. Additional Algorithm Details

We provide additional details for the algorithms proposed in this paper. Appendix A.1 describes the practical implementation of the randomized Nyström approximation and provides pseudocode for `Nyström`. Appendix A.2 describes how we compute preconditioned smoothness constants and provides pseudocode for `get_L`.

A.1 Randomized Nyström Approximation: Implementation

Here, we present a practical implementation of the Nyström approximation from Section 2.3 in `Nyström` (Algorithm 2). `Nyström` is based on Tropp et al. (2017, Algorithm 3). $\text{eps}(x)$ is defined as the positive distance between x and the next largest floating point number of the same precision as x. The resulting Nyström approximation $\hat{M}$ is given by $\hat{U}\,\text{diag}(\hat{\Lambda})\hat{U}^T$, where $\hat{U} \in \mathbb{R}^{p\times r}$ is an orthogonal matrix that contains the approximate top-r eigenvectors of M, and $\hat{\Lambda} \in \mathbb{R}^r$ contains the top-r eigenvalues of M. The Nyström approximation is psd but may have eigenvalues that are equal to 0. In our algorithms, this approximation is always used in conjunction with a regularizer to ensure positive definiteness.

Algorithm 2 `Nyström`

Require: psd matrix $M \in \mathbb{R}^{p\times p}$, approximation rank $r \leq p$

$\Omega \leftarrow \text{randn}(p, r)$ ▷ Test matrix
$\Omega \leftarrow \text{thin\_qr}(\Omega)$ ▷ Orthogonalize test matrix
$\Delta \leftarrow \text{eps}(\Omega.\text{dtype}) \cdot \text{Tr}(M)$ ▷ Compute shift
$Y_\Delta \leftarrow (M + \Delta I)\Omega$ ▷ Compute sketch, adding shift for stability
$C \leftarrow \text{chol}(\Omega^T Y_\Delta)$ ▷ Cholesky decomposition: $C^T C = \Omega^T Y_\Delta$
$B \leftarrow Y_\Delta C^{-1}$ ▷ Triangular solve
$[\hat{U}, \Sigma, \sim] \leftarrow \text{svd}(B, 0)$ ▷ Thin SVD
$\hat{\Lambda} \leftarrow \max\{0, \text{diag}(\Sigma^2 - \Delta I)\}$ ▷ Compute eigs, and remove shift with element-wise max
return $\hat{U}\,\text{diag}(\hat{\Lambda})\hat{U}^T$

The dominant costs are in computing the (shifted) sketch Y_Δ, which has complexity $\mathcal{O}(p^2 r)$, and computing an SVD of B, which has complexity $\mathcal{O}(pr^2)$. In total, the overall complexity of the algorithm is $\mathcal{O}(p^2 r + pr^2)$.

A.1.1 Applying the Nyström Approximation to a Vector

In our algorithms, we often perform computations of the form $(\hat{M}+\rho I)^{-1}g = (\hat{U}\,\text{diag}(\hat{\Lambda})\hat{U}^T + \rho I)^{-1}g$. This computation can be performed in $\mathcal{O}(rp)$ time using the Woodbury formula (Higham, 2002):

$$(\hat{U}\,\text{diag}(\hat{\Lambda})\hat{U}^T + \rho I)^{-1}g = \hat{U}\left(\text{diag}(\hat{\Lambda}) + \rho I\right)^{-1}\hat{U}^T g + \frac{1}{\rho}(g - \hat{U}\hat{U}^T g). \tag{16}$$

We also use the randomized Nyström approximation to compute preconditioned smoothness constants in `get_L` (Algorithm 3). This computation requires the calculation $(P + \rho I)^{-1/2}v$ for some $v \in \mathbb{R}^p$, which can also be performed in $\mathcal{O}(pr)$ time using the Woodbury formula:

$$(\hat{U}\,\text{diag}(\hat{\Lambda})\hat{U}^T + \rho I)^{-1/2}v = \hat{U}\left(\text{diag}(\hat{\Lambda}) + \rho I\right)^{-1/2}\hat{U}^T v + \frac{1}{\sqrt{\rho}}(v - \hat{U}\hat{U}^T v). \tag{17}$$

In single precision, (16) is unreliable for computing $(P + \rho I)^{-1} g$. This instability arises due to roundoff error: the derivation of (16) assumes that $\hat{U}^T \hat{U} = I$, but we empirically observe that orthogonality does not hold in single precision. To improve stability, we compute a Cholesky decomposition LL^T of $\rho \operatorname{diag}(\hat{\Lambda}^{-1}) + \hat{U}^T \hat{U}$, which takes $\mathcal{O}(pr^2)$ time since we form $\hat{U}^T \hat{U}$. Using the Woodbury formula and Cholesky factors,

$$\begin{aligned}(\hat{U} \operatorname{diag}(\hat{\Lambda}) \hat{U}^T + \rho I)^{-1} g &= \frac{1}{\rho} g - \frac{1}{\rho} \hat{U} (\rho \operatorname{diag}(\hat{\Lambda}^{-1}) + \hat{U}^T \hat{U})^{-1} \hat{U}^T g \\ &= \frac{1}{\rho} g - \frac{1}{\rho} \hat{U} L^{-T} L^{-1} \hat{U}^T g.\end{aligned}$$

This computation can be performed in $\mathcal{O}(pr)$ time, since the $\mathcal{O}(r^2)$ cost of triangular solves with L^T and L is negligible compared to the $\mathcal{O}(pr)$ cost of multiplication with $\hat{U}^T$ and $\hat{U}$.

Even in single precision, we find that using (17) in `get_L` works well in practice.

A.2 Computing Preconditioned Smoothness Constants

Here, we provide the details on the randomized powering procedure (`get_L`, Algorithm 3) from Section 2.4 for automatically computing the preconditioned smoothness constant. Given a symmetric matrix H, preconditioner P, and damping ρ, `get_L` uses randomized powering (Martinsson and Tropp, 2020) to compute

$$\lambda_1((P + \rho I)^{-1/2} H (P + \rho I)^{-1/2}).$$

The algorithm only requires matrix-vector products with the matrices H and $(P + \rho I)^{-1/2}$. When P is calculated using `Nyström`, we can efficiently compute a matrix-vector product with $(P + \rho I)^{-1/2}$ using (17). In practice, we find that 10 iterations of randomized powering are sufficient for estimating the preconditioned smoothness constant.

Algorithm 3 `get_L`

Require: symmetric matrix H, preconditioner P, damping ρ, maximum iterations $N \leftarrow 10$
  $v^0 \leftarrow \text{randn}(P.\text{shape}[0])$
  $v^0 \leftarrow v^0 / \|v^0\|_2$ ▷ Normalize
  for $i = 0, 1, \ldots, N-1$ **do**
    $v^{i+1} \leftarrow (P + \rho I)^{-1/2} v^i$
    $v^{i+1} \leftarrow H v^{i+1}$
    $v^{i+1} \leftarrow (P + \rho I)^{-1/2} v^{i+1}$
    $v^{i+1} \leftarrow v^{i+1} / \|v^{i+1}\|_2$ ▷ Normalize
  end for
  $\lambda \leftarrow (v^{N-1})^T v^N$
  return λ

Appendix B. Proofs of Results Appearing in the Main Paper

This appendix provides proofs for every result in the main paper whose proof was omitted. In particular, we provide detailed arguments for the RLS-to-DPP reduction in Section 5.2 and the convergence of `ASkotch` (Theorem 17).

B.1 Proof of Lemma 3

Proof Consider the matrix

$$K_\lambda^{1/2} I_{\mathcal{B}}^T \left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1} I_{\mathcal{B}} K_\lambda^{1/2}.$$

This can be rewritten as

$$K_\lambda^{1/2} I_{\mathcal{B}}^T (K_{\mathcal{B}\mathcal{B}}+\lambda I)^{-1/2} \left[(K_{\mathcal{B}\mathcal{B}} + \lambda I)^{1/2} \left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1} (K_{\mathcal{B}\mathcal{B}} + \lambda I)^{1/2}\right] (K_{\mathcal{B}\mathcal{B}}+\lambda I)^{-1/2} I_{\mathcal{B}} K_\lambda^{1/2}.$$

From this, we deduce

$$\sigma_{P_{\mathcal{B}}} \Pi_{\mathcal{B}} \preceq K_\lambda^{1/2} I_{\mathcal{B}}^T \left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1} I_{\mathcal{B}} K_\lambda^{1/2} \preceq L_{P_B} \Pi_{\mathcal{B}}.$$

By definition, $\hat{L}_{P_{\mathcal{B}}} \geq L_{P_{\mathcal{B}}}$, so

$$\sigma_{P_{\mathcal{B}}} \Pi_{\mathcal{B}} \preceq K_\lambda^{1/2} I_{\mathcal{B}}^T \left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right)^{-1} I_{\mathcal{B}} K_\lambda^{1/2} \preceq \hat{L}_{P_B} \Pi_{\mathcal{B}}.$$

Multiplying by $\hat{L}_{P_{\mathcal{B}}}^{-1}$, we establish the claim. ■

B.2 ARLS-to-DPP Reduction

Here we prove the ARLS-to-DPP reduction in Lemma 12.

Proof idea. To prove Lemma 12, we first establish that the leverage score estimates $\{\tilde{\ell}_i\}_{i=1}^n$ are rough approximations for the marginal probabilities (i.e., *marginal overestimates*) of any given element $i \in [n]$ being sampled into a set distributed according to j-DPP(A) (Lemma 20). We must then combine this guarantee with existing results on DPP coupling (Dereziński and Yang, 2024). However, since prior work focuses on uniform sampling rather than ARLS sampling, we construct a *subdivision*, which uses the marginal overestimates to create an equivalent DPP for which our ARLS sampling maps to uniform sampling (Lemma 25 and Lemma 27). This equivalence allows us to call upon existing guarantees for coupling a DPP with uniform sampling (Lemma 23).

The relationship between leverage score estimates and marginal probabilities is shown in Appendix B.2.1 and the subdivision construction is shown in Appendix B.2.2. The proof of Lemma 12 is given in Appendix B.2.3.

B.2.1 Leverage Score Estimates are Rough Approximations of Marginals

We start by showing that approximate RLSs are marginal overestimates of a j-DPP(A).

Lemma 20 (ARLS are marginal overestimates) *Given $A \in \mathbb{S}_+^n$, $k = \Omega(\log n)$, and $\tilde{\lambda} > 0$ such that $d^{\tilde{\lambda}}(A) \geq 4k$, let $\{\tilde{\ell}_i\}_{i=1}^n$ be c-approximations of $\tilde{\lambda}$-ridge leverage scores of A. For $j \in \mathcal{I} = \left[2k - \sqrt{6k \log\left(\frac{2}{\delta}\right)}, 2k + \sqrt{6k \log\left(\frac{2}{\delta}\right)}\right]$, let $\ell_{i|j} \coloneqq \Pr(i \in \mathcal{B}_{j\text{-DPP}})$ be the i-th marginal probability of $\mathcal{B}_{j\text{-DPP}} \sim j$-DPP$(A/\tilde{\lambda})$. Then, $2\tilde{\ell}_i \geq \ell_{i|j}$ for all i.*

The proof of Lemma 20 requires (i) Lemma 21, which shows that the size of a DPP sample is close to its expected size with high probability and (ii) Lemma 22, which shows that $\ell_{i|j}$ is non-decreasing in j:

Lemma 21 *Given any $A \in \mathbb{S}_+^n$, let $\mathcal{B}_{\mathrm{DPP}} \sim \mathrm{DPP}(A)$ and $\mathbb{E}[|\mathcal{B}_{\mathrm{DPP}}|]$ be its expected size. Then with probability $1-\delta$, we have $\big||\mathcal{B}_{\mathrm{DPP}}| - \mathbb{E}[|\mathcal{B}_{\mathrm{DPP}}|]\big| \leq \sqrt{3\mathbb{E}[|\mathcal{B}_{\mathrm{DPP}}|]\log\left(\frac{2}{\delta}\right)}$.*

Lemma 22 (Monotonicity) *For any $0 < j_1 \leq j_2$, we have $\ell_{i|j_1} \leq \ell_{i|j_2}$.*

The proofs of Lemmas 21 and 22 are deferred to Appendices B.2.4 and B.2.5, respectively.

Proof [Proof of Lemma 20] Define $\tilde{\mathcal{B}}_{\mathrm{DPP}} \sim \mathrm{DPP}(A/\tilde{\lambda})$ with $\mathbb{E}[|\tilde{\mathcal{B}}_{\mathrm{DPP}}|] = d^{\tilde{\lambda}} := d^{\tilde{\lambda}}(A) \geq 4k$, and denote its i-th marginal probabilities as $\ell_i^{\tilde{\lambda}} = \Pr(i \in \tilde{\mathcal{B}}_{\mathrm{DPP}})$. By Lemma 9, $\{\ell_i^{\tilde{\lambda}}\}_{i=1}^n$ are the $\tilde{\lambda}$-ridge leverage scores of A. Also let $w_{\tilde{\lambda},l} \coloneqq \Pr(|\tilde{\mathcal{B}}_{\mathrm{DPP}}| = l)$ be the probability of the set $\tilde{\mathcal{B}}_{\mathrm{DPP}}$ having size l. Then, we can express the marginals of $\tilde{\mathcal{B}}_{\mathrm{DPP}}$ via the law of total probability:

$$\ell_i^{\tilde{\lambda}} = \Pr(i \in \tilde{\mathcal{B}}_{\mathrm{DPP}}) = \sum_l \Pr(i \in \tilde{\mathcal{B}}_{\mathrm{DPP}} \mid |\tilde{\mathcal{B}}_{\mathrm{DPP}}| = l) \cdot \Pr(|\tilde{\mathcal{B}}_{\mathrm{DPP}}| = l) = \sum_l \ell_{i|l} \cdot w_{\tilde{\lambda},l},$$

where we use the fact that $\Pr(i \in \tilde{\mathcal{B}}_{\mathrm{DPP}} \mid |\tilde{\mathcal{B}}_{\mathrm{DPP}}| = l) = \Pr(i \in \mathcal{B}_{l\text{-DPP}}) = \ell_{i|l}$. Defining the interval $\tilde{\mathcal{I}} \coloneqq \left[d^{\tilde{\lambda}} - \sqrt{3d^{\tilde{\lambda}}\log\left(\frac{2}{\delta}\right)}, d^{\tilde{\lambda}} + \sqrt{3d^{\tilde{\lambda}}\log\left(\frac{2}{\delta}\right)}\right]$, by applying Lemma 21 to $\tilde{\mathcal{B}}_{\mathrm{DPP}}$, we have $|\tilde{\mathcal{B}}_{\mathrm{DPP}}| \in \tilde{\mathcal{I}}$ holds with probability $1-\delta$.

Since we assume $\delta = n^{-\mathcal{O}(1)}$ and $d^{\tilde{\lambda}} \geq 4k = \Omega(\log n)$ (with a sufficiently large constant), the infimum of $\tilde{\mathcal{I}}$ can be bounded as $d^{\tilde{\lambda}} - \sqrt{3d^{\tilde{\lambda}}\log\left(\frac{2}{\delta}\right)} \geq \frac{3}{4}d^{\tilde{\lambda}} \geq 3k$. Similarly, the supremum of $\mathcal{I}$ can be bounded as $2k + \sqrt{6k\log\left(\frac{2}{\delta}\right)} \leq 3k$, showing that $\sup\mathcal{I} \leq \inf\tilde{\mathcal{I}}$. By combining this result with Lemma 22, we have the following for any $j \in \mathcal{I}$:

$$\ell_i^{\tilde{\lambda}} = \sum_{l\in\tilde{\mathcal{I}}} \ell_{i|l} \cdot w_{\tilde{\lambda},l} + \sum_{l\notin\tilde{\mathcal{I}}} \ell_{i|l} \cdot w_{\tilde{\lambda},l} \geq \sum_{l\in\tilde{\mathcal{I}}} \ell_{i|l} \cdot w_{\tilde{\lambda},l} \geq (1-\delta)\cdot\ell_{i|3k} \geq (1-\delta)\cdot\ell_{i|j}.$$

Suppose we have RLS c-approximations $\{\tilde{\ell}_i\}_{i=1}^n$ such that $\tilde{\ell}_i \geq \ell_i^{\tilde{\lambda}}$ (see Definition 5). Since $\delta \leq 1/2$, this implies that $2\tilde{\ell}_i \geq \ell_i^{\tilde{\lambda}}/(1-\delta) \geq \ell_{i|j}$ for any $j \in \mathcal{I}$, i.e., the RLS approximations are (up to a factor of 2) marginal overestimates for j-DPP(A). ■

B.2.2 Relating ARLS Sampling to DPPs Using Subdivisions

The remainder of our analysis builds on the following lemma from Dereziński and Yang (2024), which couples a fixed-size DPP, $\mathcal{B}_{j\text{-DPP}} \sim j\text{-DPP}(A)$, with uniform sampling.

Lemma 23 (Dereziński and Yang (2024), Lemma 6.6) *Let $\mathcal{B}_{j\text{-DPP}} \sim j\text{-DPP}(A)$ be a fixed-size DPP sample where $A \in \mathbb{S}_+^n$ and $\log n < j < n$. Suppose the marginal probabilities of j-DPP(A) are near-uniform, that is, $\Pr(i \in \mathcal{B}_{j\text{-DPP}}) \leq cj/n$ holds for all $i \in [n]$ for*

some $c > 1$. Let $\mathcal{B}$ be an i.i.d. uniform sample from $[n]$ of size $\mathcal{O}(cj\log^3 n)$. Then, there is a coupling between $\mathcal{B}_{j\text{-DPP}}$ and $\mathcal{B}$, such that the joint random variable $(\mathcal{B}_{j\text{-DPP}}, \mathcal{B})$ with probability $1 - n^{-\mathcal{O}(1)}$ satisfies $\mathcal{B}_{j\text{-DPP}} \subseteq \mathcal{B}$.

Unfortunately, we cannot use Lemma 23 directly, because the "near-uniform marginals" assumption does not hold in our setting. To address this, we rely on a subdivision process, which transforms a probability distribution (such as a fixed-size DPP) into another distribution which is equivalent (in some sense) and has nearly uniform marginals. This approach, first introduced by Anari and Dereziński (2020), uniformizes the marginals by enlarging the groundset $[n]$ through selective duplication.

Definition 24 (Subdivision of j-DPP, inspired by Anari and Dereziński (2020)) *Given $A \in \mathbb{S}^n_+$, define the distribution $\mu \coloneqq j\text{-DPP}(A)$. Let matrix X be such that $A = XX^\top$, and denote $x_i^\top$ as its i-th row. Assume we have marginal overestimates $\{\tilde{\ell}_i\}_{i=1}^n$ such that $\tilde{\ell}_i \geq \Pr_{\mathcal{S}\sim\mu}(i \in \mathcal{S})$ for all $i \in [n]$. Denote $\tilde{\ell} = \sum_i \tilde{\ell}_i$, let $t_i \coloneqq \left\lceil \frac{n}{\tilde{\ell}}\tilde{\ell}_i \right\rceil$ and $\tilde{n} = \sum_i t_i$. For each $i \in [n]$, we create t_i copies of $\frac{1}{\sqrt{t_i}}x_i^\top$ and let the collection of all these vectors form the new matrix $\tilde{X}$. Let $\tilde{A} = \tilde{X}\tilde{X}^\top \in \mathbb{S}^{\tilde{n}}_+$, we define the subdivision process of μ as $\mu' = j\text{-DPP}(\tilde{A})$.*

We start by stating that the subdivision process produces a distribution with near-uniform marginals (Anari et al., 2024).

Lemma 25 (Anari et al. (2024), Proposition 23) *Let distribution $\mu = j\text{-DPP}(A)$, and $\mu' = j\text{-DPP}(\tilde{A})$ be the subdivision process of μ. Then, μ' has near-uniform marginals: for all $i^{(j)} \in [\tilde{n}]$ we have $\Pr_{\tilde{\mathcal{S}}\sim\mu'}(i^{(j)} \in \tilde{\mathcal{S}}) \leq \frac{\tilde{\ell}}{n} \leq \frac{2\tilde{\ell}}{\tilde{n}}$.*

The following lemma shows an equivalence between $\mu = j\text{-DPP}(A)$ and its subdivision (up to a mapping of the elements), which allows us to transform the problem of sampling from μ to sampling from its subdivision μ'.

Lemma 26 (Equivalence between j-DPP and its subdivision) *Let distribution $\mu = j\text{-DPP}(A)$, and $\mu' = j\text{-DPP}(\tilde{A})$ be its subdivision process. Let $\tilde{\mathcal{B}} \sim j\text{-DPP}(\tilde{A})$ and define a function $\pi : [\tilde{n}] \to [n]$ which maps the t_i duplicates (in $[\tilde{n}]$) of element $i \in [n]$ back to i. Then, $\pi(\tilde{\mathcal{B}}) \sim j\text{-DPP}(A)$.*

Proof By the definition of the subdivision, if two identical rows are sampled from $\tilde{X}$, then the determinant of the corresponding principal submatrix is 0, as is the probability measure μ'. Thus without loss of generality, we can assume that we only sample distinct rows from $\tilde{X}$. Let $\tilde{\mathcal{B}} = \{i_1^{(l_1)}, \ldots, i_j^{(l_j)}\} \subseteq \binom{[\tilde{n}]}{j}$ be a subset sampled from μ', and let $\hat{\mathcal{B}} = \pi(\tilde{\mathcal{B}}) \coloneqq \{\pi(i_1^{(l_1)}), \ldots, \pi(i_j^{(l_j)})\} = \{i_1, \ldots, i_j\} \subseteq \binom{[n]}{j}$. Then, from definition we have

$$\mu'(\tilde{\mathcal{B}}) \propto \det(\tilde{X}_{\tilde{\mathcal{B}}}\tilde{X}_{\tilde{\mathcal{B}}}^\top) = \frac{\det(X_{\hat{\mathcal{B}}}X_{\hat{\mathcal{B}}}^\top)}{t_{i_1}\cdots t_{i_j}} \propto \frac{\mu(\hat{\mathcal{B}})}{t_{i_1}\cdots t_{i_j}}.$$

Note that there are precisely $t_{i_1}\cdots t_{i_j}$ different sets of duplicates that map to the same set $\hat{\mathcal{B}}$, so a random set $\tilde{\mathcal{B}} \sim \mu' = j\text{-DPP}(\tilde{A})$ after mapping with π is distributed exactly according to $\mu = j\text{-DPP}(A)$. ■

Having shown that sampling from μ is equivalent to sampling from its subdivision μ', we need to establish a sampling scheme on $[n]$. Intuitively, importance sampling on $[n]$ based on probabilities proportional to $\{\tilde{\ell}_i\}_{i=1}^n$ should be equivalent to uniform sampling on $[\tilde{n}]$. However, the definition of t_i introduces rounding error. To address this issue, we use the ARLS sampling scheme introduced in Definition 5. In the following result, we show the equivalence between this sampling scheme and uniform sampling on the enlarged subdivision ground set $[\tilde{n}]$.

Lemma 27 (From subdivided uniform sampling to marginal sampling) *Let $\{\tilde{\ell}_i\}_{i=1}^n$ be the marginal overestimates, and define $p_i \coloneqq \frac{\tilde{\ell}}{n} \cdot \lceil \frac{n}{\tilde{\ell}} \tilde{\ell}_i \rceil$ where $\tilde{\ell} = \sum_i \tilde{\ell}_i$. Let $\mathfrak{o}$ be the importance sampling distribution defined on $[n]$ according to probabilities $\{p_i\}_{i=1}^n$, and $\mathfrak{o}'$ be the uniform sampling distribution defined on $[\tilde{n}]$. Let $I \sim \mathfrak{o}$, and let $\hat{I}$ be a uniformly random element in $\pi^{-1}(I)$. Then $\hat{I} \sim \mathfrak{o}'$.*

Proof For any $i \in [n]$ and $l \in [t_i]$, we have the following:

$$\begin{aligned}
\Pr(\hat{I} = i^{(l)}) &= \Pr(I = i) \cdot \Pr(\hat{I} = i^{(l)} \mid I = i) \\
&= \frac{p_i}{\sum_{j=1}^n p_j} \cdot \frac{1}{t_i} \\
&= \frac{\frac{\tilde{\ell}}{n} t_i}{\frac{\tilde{\ell}}{n} \sum_{j=1}^n t_j} \cdot \frac{1}{t_i} \\
&= \frac{1}{\sum_{j=1}^n t_j} \\
&= \frac{1}{\tilde{n}}.
\end{aligned}$$

Therefore, $\hat{I}$ is uniformly random in $[\tilde{n}]$. ■

B.2.3 Proof of Lemma 12

With all the pieces in place, we now prove Lemma 12.

Proof Suppose that $\{\tilde{\ell}_i\}_{i=1}^n$ are c-approximations of $\tilde{\lambda}$-ridge leverage scores of A as in Definition 5. Then, according to Lemma 20, $\{2\tilde{\ell}_i\}_{i=1}^n$ are the marginal overestimates of j-DPP(A) for any $j \in \mathcal{I}$. Thus by Lemma 27, our ARLS$_c^{\tilde{\lambda}}$-sampling is equivalent (up to the mapping π) to uniform sampling on the subdivision ground set $[\tilde{n}]$.

Now for $\mu = j$-DPP(A), we look at its subdivision process $\mu' = j$-DPP$(\tilde{A})$. According to Lemma 25, μ' has near-uniform marginals, i.e., $\Pr_{\mathcal{S}\sim\mu'}(i \in \mathcal{S}) \leq 2cj/n$. Thus by Lemma 23, if we let $\mathcal{B}'$ be a i.i.d. uniform sample from $[\tilde{n}]$ of size $\mathcal{O}(cj \log^3 \tilde{n}) = \mathcal{O}(ck \log^3 n)$, then there is a coupling $(\mathcal{B}', \mathcal{B}'_{j\text{-DPP}})$, such that with probability $1 - n^{-\mathcal{O}(1)}$, we have $\mathcal{B}'_{j\text{-DPP}} \subseteq \mathcal{B}'$, where $\mathcal{B}'_{j\text{-DPP}} \sim \mu'$.

For one side, by Lemma 26 we know that $\pi(\mathcal{B}'_{j\text{-DPP}})$ is distributed identically to the sample $\mathcal{B}_{j\text{-DPP}}$, where π is the mapping from Lemma 26. For the other side, by Lemma 27, we know that $\pi(\mathcal{B}')$ is distributed according to ARLS$_c^{\tilde{\lambda}}$-sampling, i.e., the same as $\mathcal{B}$ from

the statement of Lemma 12. We conclude that for any $j \in \mathcal{I}$, if we do $\text{ARLS}_c^{\tilde{\lambda}}$-sampling with sample size $b = \mathcal{O}(ck\log^3 n)$ and obtain $\mathcal{B}$, then we can couple $\mathcal{B}$ with $\mathcal{B}_{j\text{-DPP}}$ such that $\mathcal{B}_{j\text{-DPP}} \subseteq \mathcal{B}$ with probability $1 - n^{-\mathcal{O}(1)}$. ■

B.2.4 Proof of Lemma 21

To prove Lemma 21, we start with the following result from Kulesza and Taskar (2012), which relates the cardinality of a random-size DPP to a sum of independent Bernoulli random variables.

Lemma 28 (Kulesza and Taskar (2012), Algorithm 1 and Theorem 2.3) *For $A \in \mathbb{S}_+^n$, let $A = \sum_i \lambda_i u_i u_i^\top$ be its eigendecomposition where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Suppose we independently sample $\gamma_i \sim \text{Bernoulli}\left(\frac{\lambda_i}{\lambda_i+1}\right)$ for $i \in [n]$ and we also sample $\mathcal{B}_{\text{DPP}} \sim \text{DPP}(A)$. Then $|\mathcal{B}_{\text{DPP}}| \overset{d}{=} \sum_{i=1}^n \gamma_i$.*

We also require the following Chernoff bound, which is a classic result in probability theory:

Lemma 29 (Chernoff bound) *Let $\bar{Z} = \sum_{i=1}^n Z_i$ where $Z_i \sim \text{Bernoulli}(p_i)$ are independent Bernoulli random variables. Let $\mu = \mathbb{E}[\bar{Z}]$, then for all $\epsilon \in (0,1)$ we have*

$$\Pr(|\bar{Z} - \mu| \geq \epsilon\mu) \leq 2\exp(-\epsilon^2\mu/3).$$

Proof [Proof of Lemma 21] Let $\{\lambda_i\}_{i=1}^n$ be the eigenvalues of A in decreasing order, and let $\gamma_i \sim \text{Bernoulli}\left(\frac{\lambda_i}{\lambda_i+1}\right)$ be n independent random variables. By Lemma 28, $\mathbb{E}[|\mathcal{B}_{\text{DPP}}|] = \sum_i \mathbb{E}[\gamma_i] = \sum_i \frac{\lambda_i}{\lambda_i+1}$. Applying Lemma 29 to $\{\gamma_i\}_{i=1}^n$, we obtain

$$\Pr\left(\left|\sum_i \gamma_i - \sum_i \mathbb{E}[\gamma_i]\right| \geq \epsilon \sum_i \mathbb{E}[\gamma_i]\right) \leq 2\exp\left(-\epsilon^2 \sum_i \mathbb{E}[\gamma_i]/3\right),$$

which gives that with probability $1-\delta$, $\left||\mathcal{B}_{\text{DPP}}| - \mathbb{E}[|\mathcal{B}_{\text{DPP}}|]\right| \leq \sqrt{3\mathbb{E}[|\mathcal{B}_{\text{DPP}}|]\log\left(\frac{2}{\delta}\right)}$. ■

B.2.5 Proof of Lemma 22

The proof of Lemma 22 relies on elementary symmetric polynomials (Definition 30) and Newton's inequalities (Lemma 31).

Definition 30 (Elementary symmetric polynomial) *Given vector $\Lambda = (\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^n$, we define its i-th elementary symmetric polynomial as*

$$s_i(\Lambda) := \sum_{\mathcal{S} \in \binom{[n]}{i}} \prod_{j \in \mathcal{S}} \lambda_j.$$

Lemma 31 (Newton's inequalities) *For non-negative real numbers $\lambda_1, \ldots, \lambda_n$, let s_j be the j-th elementary symmetric polynomial in $\lambda_1, \ldots, \lambda_n$. If we denote $\bar{s}_j = s_j / \binom{n}{j}$, then $\bar{s}_j^2 \geq \bar{s}_{j+1}\bar{s}_{j-1}$.*

Proof [Proof of Lemma 22] Recall that we define $\ell_{i|j} = \Pr(i \in \mathcal{B}_{j\text{-DPP}})$, and it can also be written as $\ell_{i|j} = \Pr(i \in \mathcal{B}_{\text{DPP}} \mid |\mathcal{B}_{\text{DPP}}| = j)$, where $\mathcal{B}_{\text{DPP}} \sim \text{DPP}(A/\bar{\lambda})$ is a random-size DPP sample. Next, we use Kulesza and Taskar (2012, Eq. 5.33):

$$\ell_{i|j} = \Pr(i \in \mathcal{B}_{\text{DPP}} \mid |\mathcal{B}_{\text{DPP}}| = j) = \sum_{p=1}^{n} (e_i^T v_p)^2 \lambda_p \frac{s_{j-1}(\Lambda_{-p})}{s_j(\Lambda)},$$

where $\{e_i\}_{i=1}^n$ are the standard basis vectors, $\{v_p\}_{p=1}^n$ are the eigenvectors of A, and $\lambda_p \coloneqq \lambda_p(A/\bar{\lambda})$.

In order to show that $\ell_{i|j}$ is non-decreasing in j, it suffices to show that $\frac{s_{j-1}(\Lambda_{-p})}{s_j(\Lambda)}$ is non-decreasing in j for any given $p \in [n]$. Denote $\bar{s}_j = s_j(\Lambda)/\binom{[n]}{j}$ as the mean of the j-th elementary symmetric polynomial, then by Lemma 31 we have

$$1 \geq \frac{\bar{s}_{j-1}\bar{s}_{j+1}}{\bar{s}_j^2} = \frac{\binom{[n]}{j}^2}{\binom{[n]}{j-1}\binom{[n]}{j+1}} \frac{s_{j-1}(\Lambda)s_{j+1}(\Lambda)}{s_j^2(\Lambda)} = \frac{(j+1)(n-j+1)}{j(n-j)} \frac{s_{j-1}(\Lambda)s_{j+1}(\Lambda)}{s_j^2(\Lambda)},$$

which gives

$$\frac{s_{j+1}(\Lambda)}{s_j(\Lambda)} \leq \frac{j(n-j)}{(j+1)(n-j+1)} \frac{s_j(\Lambda)}{s_{j-1}(\Lambda)} < \frac{s_j(\Lambda)}{s_{j-1}(\Lambda)}.$$

Notice that $s_{j+1}(\Lambda) = s_{j+1}(\Lambda_{-p}) + \lambda_p \cdot s_j(\Lambda_{-p})$, and by using this fact we have

$$\frac{s_j(\Lambda_{-p})}{s_{j+1}(\Lambda)} = \frac{s_j(\Lambda_{-p})}{s_{j+1}(\Lambda_{-p}) + \lambda_p \cdot s_j(\Lambda_{-p})} = \frac{1}{\lambda_p + \frac{s_{j+1}(\Lambda_{-p})}{s_j(\Lambda_{-p})}}.$$

We have shown that $\frac{s_{j+1}(\Lambda_{-p})}{s_j(\Lambda_{-p})}$ is decreasing in j when p is fixed, so

$$\frac{s_j(\Lambda_{-p})}{s_{j+1}(\Lambda)} = \frac{1}{\lambda_p + \frac{s_{j+1}(\Lambda_{-p})}{s_j(\Lambda_{-p})}} > \frac{1}{\lambda_p + \frac{s_j(\Lambda_{-p})}{s_{j-1}(\Lambda_{-p})}} = \frac{s_{j-1}(\Lambda_{-p})}{s_j(\Lambda)},$$

which shows that $\frac{s_{j-1}(\Lambda_{-p})}{s_j(\Lambda)}$ is increasing in j for any $p \in [n]$. ■

B.3 Proof of Corollary 14

Proof Theorem 13 guarantees that

$$\hat{\mu} \geq \frac{\lambda}{16\rho\bar{\kappa}_{k:n}} \frac{k}{n}.$$

We upper bound $\bar{\kappa}_{k:n}$ to deduce the claim. By definition,

$$\begin{aligned}\bar{\kappa}_{k:n} &= \frac{1}{n-k}\sum_{j>k}\frac{\lambda_j(K_\lambda)}{\lambda_{\min}(K_\lambda)} \\ &= \frac{1}{n-k}\sum_{j>k}\frac{\lambda_j(K)+\lambda}{\lambda_{\min}(K)+\lambda} \\ &\leq \frac{1}{n-k}\sum_{j>k}\frac{2\lambda}{\lambda_{\min}(K)+\lambda} \\ &\leq 2.\end{aligned}$$

Here the third inequality uses $\lambda_j(K) \leq \lambda$ whenever $j \geq 2d^\lambda(K)$ (Frangella et al., 2023, Lemma 5.4). Thus, $\bar{\kappa}_{k:n}^{-1} \geq 1/2$, which yields the claim. ■

B.4 Proof of Proposition 15

We begin with the following preliminary technical result.

Lemma 32 *Let $K \in \mathbb{S}_+^n$, $I_\mathcal{B} \in \mathbb{R}^{b\times n}$ be a row-selection matrix generated via an arbitrary sampling scheme. Set $K_{\mathcal{B}\mathcal{B}} = I_\mathcal{B} K I_\mathcal{B}^T$. Then, for any $\rho > 0$,*

$$d^\rho(K_{\mathcal{B}\mathcal{B}}) \leq \sum_{j=1}^{b}\frac{\lambda_j(K)}{\lambda_j(K)+\rho}.$$

Proof As $K_{\mathcal{B}\mathcal{B}} = I_\mathcal{B} K I_\mathcal{B}^T$, it is a compression of K to the subspace spanned by the columns of $I_\mathcal{B}^T$. Consequently, Cauchy's Interlacing Theorem (Bhatia, 2013, Corollary III.1.5) yields

$$\lambda_j(K_{\mathcal{B}\mathcal{B}}) \leq \lambda_j(K).$$

Now, by definition of $d^\rho(\cdot)$ and the fact that $f(x) = x/(x+\rho)$ is increasing in x for $\rho > 0$, we have

$$d^\rho(K_{\mathcal{B}\mathcal{B}}) = \sum_{j=1}^{b}\frac{\lambda_j(K_{\mathcal{B}\mathcal{B}})}{\lambda_j(K_{\mathcal{B}\mathcal{B}})+\rho} \leq \sum_{j=1}^{b}\frac{\lambda_j(K)}{\lambda_j(K)+\rho}.$$

■

We now commence the proof of Proposition 15.

Proof [Proof of Proposition 15] We begin by observing that Lemma 32 yields $d^\rho(K_{\mathcal{B}\mathcal{B}}) \leq d^\rho(\lfloor K \rfloor_b)$. As $\hat{K}_{\mathcal{B}\mathcal{B}}$ is constructed from a sparse sign embedding with $r = \mathcal{O}\left(d^\rho(\lfloor K\rfloor_b)\log\left(\frac{d^\rho(\lfloor K\rfloor_b)}{\delta}\right)\right)$ columns and $\zeta = \mathcal{O}\left(\log(\frac{d^\rho(\lfloor K\rfloor_b)}{\delta})\right)$ non-zeros per column, Lemma 4.6 in Dereziński et al. (2025b) implies

$$\|K_{\mathcal{B}\mathcal{B}} - \hat{K}_{\mathcal{B}\mathcal{B}}\| \leq 2\lambda_{r+1}(K_{\mathcal{B}\mathcal{B}}) + \frac{1}{r}\sum_{j>r}\lambda_j(K_{\mathcal{B}\mathcal{B}}), \text{ with probability at least } 1-\delta.$$

Now, combining Lemma 5.4 in Frangella et al. (2023) with $r = \mathcal{O}\left(d^{\rho}(\lfloor K \rfloor_b) \log\left(\frac{d^{\rho}(\lfloor K \rfloor_b)}{\delta}\right)\right)$ yields

$$\lambda_{r+1}(K_{\mathcal{B}\mathcal{B}}) \leq \frac{\rho}{4}, \quad \frac{1}{r}\sum_{j>r} \lambda_j(K_{\mathcal{B}\mathcal{B}}) \leq \frac{\rho}{2}.$$

Thus,

$$\|K_{\mathcal{B}\mathcal{B}} - \hat{K}_{\mathcal{B}\mathcal{B}}\| \leq \rho.$$

Combining this last display with the relation that $\hat{K}_{\mathcal{B}\mathcal{B}} \preceq K_{\mathcal{B}\mathcal{B}}$ and our hypothesis that $\rho \geq \lambda$, we find

$$\begin{aligned} K_{\mathcal{B}\mathcal{B}} + \lambda I \preceq K_{\mathcal{B}\mathcal{B}} + \rho I &= \hat{K}_{\mathcal{B}\mathcal{B}} + \rho I + \left(K_{\mathcal{B}\mathcal{B}} - \hat{K}_{\mathcal{B}\mathcal{B}}\right) \\ &\preceq \hat{K}_{\mathcal{B}\mathcal{B}} + \rho I + \rho I \preceq (1+1)\left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right) \\ &= 2(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I). \end{aligned}$$

Moreover,

$$K_{\mathcal{B}\mathcal{B}} + \lambda I = K_{\mathcal{B}\mathcal{B}} + \frac{\lambda}{\rho}\rho I \succeq \frac{\lambda}{\rho}(K_{\mathcal{B}\mathcal{B}} + \rho I).$$

Combining these bounds with $K_{\mathcal{B}\mathcal{B}} \succeq \hat{K}_{\mathcal{B}\mathcal{B}}$, we deduce

$$\frac{\lambda}{\rho}\left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right) \preceq K_{\mathcal{B}\mathcal{B}} + \lambda I \preceq 2\left(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I\right).$$

The preceding display immediately implies

$$\frac{\lambda}{\rho} \leq \sigma_{P_{\mathcal{B}}} \leq L_{P_{\mathcal{B}}} \leq 2.$$

As $\hat{L}_{P_{\mathcal{B}}} = \max\{L_{P_{\mathcal{B}}}, 1\}$, it follows that $\hat{L}_{P_{\mathcal{B}}} \leq 2$. Invoking Lemma 3 we conclude the proof. ■

B.5 Proof of Theorem 17

In this section, we prove Theorem 17 by analyzing the convergence of `ASkotch`. We begin with some preliminaries and notation.

B.5.1 Preliminaries and Notation

The convergence analysis of `ASkotch` is based on a Lyapunov function argument. For NSAP applied to a symmetric psd matrix $A \in \mathbb{R}^n$, Gower et al. (2018) establishes convergence in terms of the Lyapunov function

$$\Delta_t := \|v_t - w_\star\|^2_{B^{1/2}\mathbb{E}[\Pi_{\mathcal{B}}]^+ B^{1/2}} + \frac{1}{\mu}\|w_t - w_\star\|^2_B,$$

where $B \in \mathbb{S}^n_{++}$, $\Pi_{\mathcal{B}} := B^{-1/2}AS^T\left(SAB^{-1}AS^T\right)^+ SAB^{-1/2}$, and $\mu := \lambda^+_{\min}(\mathbb{E}[\Pi_{\mathcal{B}}])$.

In particular, they establish the following result:

Proposition 33 (Gower et al. (2018), Eq. 39) *Suppose that*

$$\operatorname{null}(\mathbb{E}[\Pi_{\mathcal{B}}]) = \operatorname{null}(A).$$

Then at iteration t, NSAP satisfies

$$\mathbb{E}[\Delta_t \mid w_{t-1}, v_{t-1}, z_{t-1}] \leq \left(1 - \sqrt{\frac{\mu}{\nu}}\right) \Delta_{t-1},$$

where

$$\mu = \lambda_{\min}\left(\mathbb{E}[\Pi_{\mathcal{B}}]\right), \quad \nu \coloneqq \lambda_{\max}\left(\mathbb{E}\left[\left(\mathbb{E}[\Pi_{\mathcal{B}}]^{-1/2}\Pi_{\mathcal{B}}\mathbb{E}[\Pi_{\mathcal{B}}]^{-1/2}\right)^2\right]\right).$$

Proposition 33 introduces a new parameter, ν, the largest eigenvalue of the second moment of the normalized projection matrix. Similar to how $\hat{\mu}$ is an analogue μ for analyzing `ASkotch`, there is an analogue to ν for analyzing `ASkotch`:

$$\hat{\nu} \coloneqq \lambda_{\max}\left(\mathbb{E}\left[\left(\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2}\hat{\Pi}_{\mathcal{B},\rho}\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2}\right)^2\right]\right).$$

This new parameter $\hat{\nu}$ corresponds to the second moment of the normalized approximate projection matrix. To establish our convergence result, we need to upper bound $\hat{\nu}$. The following lemma does precisely this.

Lemma 34 (Upper bound on $\hat{\nu}$) *Suppose that $\mathcal{B}$ and $\hat{K}_{\mathcal{B}\mathcal{B}}$ are constructed according to the hypotheses of Theorem 17, then*

$$\hat{\nu} \leq \frac{8(\rho + \bar{\lambda})}{\lambda} \leq 8\left(\frac{\rho}{\lambda} + \frac{n}{k}\right),$$

where $\bar{\lambda}$ is as in Lemma 10.

Proof The proof parallels the proof of Theorem 3.7 in Dereziński et al. (2025c). Our hypothesis on the construction of $\mathcal{B}$ and $\hat{K}_{\mathcal{B}\mathcal{B}}$ allows us to invoke Theorem 13 to reach

$$\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}] \succeq \frac{1}{8\kappa_\rho} K_\lambda (K_\lambda + \bar{\lambda})^{-1},$$

where $\kappa_\rho = \rho/\lambda \geq 1$. The preceding display implies

$$\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1} \preceq 8\kappa_\rho (I + \bar{\lambda} K_\lambda^{-1}).$$

To upper bound $\hat{\nu}$, we observe that it may be rewritten as

$$\hat{\nu} = \left\| \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \mathbb{E}\left[\hat{\Pi}_{\mathcal{B},\rho}\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1}\hat{\Pi}_{\mathcal{B},\rho}\right] \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \right\|.$$

Combining this with the upper bound on $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1}$ and the fact that $\hat{\Pi}_{\mathcal{B},\rho} \preceq I$, we deduce

$$\begin{aligned}
\hat{\nu} &\leq 8\kappa_\rho \left\| \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \mathbb{E}\left[\hat{\Pi}_{\mathcal{B},\rho}^2 + \bar{\lambda}\hat{\Pi}_{\mathcal{B},\rho} K_\lambda^{-1} \hat{\Pi}_{\mathcal{B},\rho}\right] \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \right\| \\
&\leq 8\kappa_\rho \left\| \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \mathbb{E}\left[\hat{\Pi}_{\mathcal{B},\rho} + \bar{\lambda}\hat{\Pi}_{\mathcal{B},\rho} K_\lambda^{-1} \hat{\Pi}_{\mathcal{B},\rho}\right] \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \right\| \\
&\leq 8\kappa_\rho \left(1 + \bar{\lambda} \left\| \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \mathbb{E}\left[\hat{\Pi}_{\mathcal{B},\rho} K_\lambda^{-1} \hat{\Pi}_{\mathcal{B},\rho}\right] \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \right\| \right). \qquad (*)
\end{aligned}$$

Now, using the definition of $\hat{\Pi}_{\mathcal{B},\rho}$, we can express the middle term as follows:

$$\begin{aligned}
\hat{\Pi}_{\mathcal{B},\rho} K_\lambda^{-1} \hat{\Pi}_{\mathcal{B},\rho} &= \frac{1}{\hat{L}_{P_\mathcal{B}}^2} K_\lambda^{1/2} I_\mathcal{B}^T (\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1} I_\mathcal{B} (K_\lambda^{1/2} K_\lambda^{-1} K_\lambda^{1/2}) I_\mathcal{B}^T (\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1} I_\mathcal{B} K_\lambda^{1/2} \\
&= \frac{1}{\hat{L}_{P_\mathcal{B}}^2} K_\lambda^{1/2} I_\mathcal{B}^T (\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1} I_\mathcal{B} I_\mathcal{B}^T (\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1} I_\mathcal{B} K_\lambda^{1/2} \\
&= \frac{1}{\hat{L}_{P_\mathcal{B}}^2} K_\lambda^{1/2} I_\mathcal{B}^T (\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-2} I_\mathcal{B} K_\lambda^{1/2} \\
&\preceq \frac{1}{\rho \hat{L}_{P_\mathcal{B}}^2} K_\lambda^{1/2} I_\mathcal{B}^T (\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1} I_\mathcal{B} K_\lambda^{1/2} \\
&= \frac{1}{\rho \hat{L}_{P_\mathcal{B}}} \hat{\Pi}_{\mathcal{B},\rho} \preceq \frac{1}{\rho} \hat{\Pi}_{\mathcal{B},\rho}.
\end{aligned}$$

Here, the last inequality uses that by definition $\hat{L}_{P_\mathcal{B}} \geq 1$. By plugging the preceding upper bound into $(*)$, we reach

$$\begin{aligned}
\hat{\nu} &\leq 8\kappa_\rho \left(1 + \frac{\bar{\lambda}}{\rho} \left\| \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \mathbb{E}\left[\hat{\Pi}_{\mathcal{B},\rho} K_\lambda^{-1} \hat{\Pi}_{\mathcal{B},\rho} \right] \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1/2} \right\| \right) \\
&= 8\kappa_\rho + \frac{8\kappa_\rho \bar{\lambda}}{\rho} \\
&= \frac{8\left(\rho + \bar{\lambda}\right)}{\lambda}.
\end{aligned}$$

This proves the first inequality. To prove the second inequality, recall that Lemma 10 implies $\bar{\lambda} \leq \frac{1}{k} \sum_{j>k} \lambda_j(K_\lambda)$. Combining this with the preceding display, we conclude

$$\begin{aligned}
\hat{\nu} &\leq \frac{8\left(\rho + \frac{1}{k}\sum_{j>k} \lambda_j(K_\lambda)\right)}{\lambda} \\
&= 8\frac{\rho + \frac{n-k}{k}\lambda + \frac{1}{k}\sum_{j>k} \lambda_j(K)}{\lambda} \\
&\leq 8\frac{\rho + \frac{n-k}{k}\lambda + \lambda}{\lambda} \\
&= 8\left(\frac{\rho}{\lambda} + \frac{n}{k}\right).
\end{aligned}$$

Here, the second inequality follows from the fact that $\frac{1}{k}\sum_{j>k} \lambda_j(K) \leq \lambda$ when $k \geq d^\lambda(K)$ (Frangella et al., 2023, Lemma 5.4). ■

B.5.2 Convergence Proof of ASkotch

To show the convergence of ASkotch, we use the Lyapunov function

$$\tilde{\Delta}_t \coloneqq \|v_t - w_\star\|^2_{K_\lambda^{1/2} \mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1} K_\lambda^{1/2}} + \frac{1}{\hat{\mu}} \|w_t - w_\star\|^2_{K_\lambda}, \tag{18}$$

where $\hat{\mu} = \lambda_{\min}(\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}])$. The Lyapunov function (18) is identical to the one for NSAP except $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$ replaces $\mathbb{E}[\Pi_{\mathcal{B}}]$ and $\hat{\mu}$ replaces μ. The replacement stems from the fact that `ASkotch` computes the search direction with $\frac{1}{\hat{L}_{P_{\mathcal{B}}}}(\hat{K}_{\mathcal{B}\mathcal{B}} + \rho I)^{-1}$ instead of $(K_{\mathcal{B}\mathcal{B}} + \lambda I)^{-1}$.

Proof [Proof of Theorem 17, Item 2] The only difference between `ASkotch` and NSAP is that $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$ replaces $\mathbb{E}[\Pi_{\mathcal{B}}]$, so we can apply the same argument as Gower et al. (2018) to show the convergence for the modified Lyapunov function (18). Thus, we only need to ensure $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$ satisfies the following condition from Proposition 33:

$$\operatorname{null}(\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]) = \operatorname{null}(K_\lambda) = \{0\}.$$

Indeed, this is necessary for the modified Lyapunov function to make sense; otherwise, $\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]$ is singular. Under the hypotheses of Theorem 17, Theorem 13 guarantees $\hat{\mu} > 0$, which immediately implies

$$\operatorname{null}(\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]) = \operatorname{null}(K_\lambda) = \{0\}.$$

Thus, we can invoke arguments in Gower et al. (2018) to arrive at a modified version of Proposition 33 where μ and ν are replaced by $\hat{\mu}$ and $\hat{\nu}$ respectively. Applying this modified version of Proposition 33, we deduce

$$\mathbb{E}[\tilde{\Delta}_t \mid w_{t-1}, v_{t-1}, z_{t-1}] \leq \left(1 - \sqrt{\frac{\hat{\mu}}{\hat{\nu}}}\right) \tilde{\Delta}_{t-1}.$$

Applying the law of total expectation yields

$$\mathbb{E}[\tilde{\Delta}_t] \leq \left(1 - \sqrt{\frac{\hat{\mu}}{\hat{\nu}}}\right)^t \tilde{\Delta}_0.$$

Using the definition of $\tilde{\Delta}_t$, we reach

$$\begin{aligned}
\mathbb{E}[\|w_t - w_\star\|_{K_\lambda}^2] &\leq \left(1 - \sqrt{\frac{\hat{\mu}}{\hat{\nu}}}\right)^t \left(\hat{\mu}\|w_0 - w_\star\|^2_{K_\lambda^{1/2}\mathbb{E}[\hat{\Pi}_{\mathcal{B},\rho}]^{-1}K_\lambda^{1/2}} + \|w_0 - w_\star\|_{K_\lambda}^2\right) \\
&\leq \left(1 - \sqrt{\frac{\hat{\mu}}{\hat{\nu}}}\right)^t \left(\hat{\mu} \cdot \frac{1}{\hat{\mu}}\|w_0 - w_\star\|_{K_\lambda}^2 + \|w_0 - w_\star\|_{K_\lambda}^2\right) \\
&= 2\left(1 - \sqrt{\frac{\hat{\mu}}{\hat{\nu}}}\right)^t \|w_0 - w_\star\|_{K_\lambda}^2. \qquad (*)
\end{aligned}$$

To obtain a fine-grained convergence rate, we apply our bounds on $\hat{\mu}$ and $\hat{\nu}$. Corollary 14 lower bounds $\hat{\mu}$ as

$$\hat{\mu} \geq \frac{\lambda}{32\rho}\frac{k}{n}.$$

On the other hand, Lemma 34 upper bounds $\hat{\nu}$ as

$$\hat{\nu} \leq 8\left(\frac{\rho}{\lambda} + \frac{n}{k}\right) \leq 16 \max\left\{\frac{\rho}{\lambda}, \frac{n}{k}\right\}.$$

Thus,

$$\frac{1}{\hat{\nu}} \geq \frac{1}{16} \min\left\{\frac{\lambda}{\rho}, \frac{k}{n}\right\}.$$

Combining these lower bounds, we deduce

$$\sqrt{\frac{\hat{\mu}}{\hat{\nu}}} \geq \frac{1}{16\sqrt{2}} \min\left\{\sqrt{\frac{\lambda}{\rho}}\frac{k}{n}, \sqrt{\frac{k}{n}}\frac{\lambda}{\rho}\right\}.$$

Combining the preceding display with $(*)$, we conclude the following convergence guarantee:

$$\mathbb{E}[\|w_t - w_\star\|_{K_\lambda}^2] \leq 2\left(1 - \frac{1}{16\sqrt{2}} \min\left\{\sqrt{\frac{\lambda}{\rho}}\frac{k}{n}, \sqrt{\frac{k}{n}}\frac{\lambda}{\rho}\right\}\right)^t \|w_0 - w_\star\|_{K_\lambda}^2.$$

■

B.6 Proof of Corollary 18

Proof Our hypothesis that $\max_{i\in[n]} \ell_i^{\tilde{\lambda}}(K_\lambda) = \Theta\left(\frac{d^{\tilde{\lambda}}(K_\lambda)}{n}\right)$ allows us to invoke Corollary 11 to ensure that the conclusion of Theorem 17 holds when `ASkotch` uses uniform sampling with a blocksize $b = \Theta\left(k \log^3 n\right)$. Moreover, the assumption on ρ guarantees that we are in convergence regime (i). Hence, the number of iterations required to obtain an ϵ-approximate solution is $\mathcal{O}\left(\sqrt{c_\rho}\frac{n}{k}\log\left(\frac{1}{\epsilon}\right)\right)$. When a sparse sign embedding is used to construct $\hat{K}_{\mathcal{B}\mathcal{B}}$, the per-iteration cost of `ASkotch` is $\tilde{\mathcal{O}}(nb + br^2)$. Combining this with the iteration complexity bound and $b = \Theta(k \log^3 n)$, we conclude that the total cost of `ASkotch` is given by

$$\tilde{\mathcal{O}}\left(\sqrt{c_\rho}\left(n^2 + nr^2\right)\log\left(\frac{1}{\epsilon}\right)\right).$$

The claim now follows by observing that $r = \tilde{\mathcal{O}}(\sqrt{n})$ as $d^\rho(\lfloor K \rfloor_b) \leq d^\lambda(K) = \mathcal{O}(\sqrt{n})$. ■

References

Amirhesam Abedsoltan, Mikhail Belkin, and Parthe Pandit. Toward large kernel models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Ahmed Alaoui and Michael W Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems*, 2015.

Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *Journal of Machine Learning Research*, 18(221):1–51, 2018.

Zeyuan Allen-Zhu, Zheng Qu, Peter Richtarik, and Yang Yuan. Even faster accelerated coordinate descent using non-uniform sampling. In *Proceedings of The 33rd International Conference on Machine Learning*, 2016.

Nima Anari and Michał Dereziński. Isotropy and log-concave polynomials: Accelerated sampling and high-precision counting of matroid bases. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, 2020.

Nima Anari, Yang P Liu, and Thuy-Duong Vuong. Optimal sublinear sampling of spanning trees and determinantal point processes via average-case entropic independence. *SIAM Journal on Computing*, pages FOCS22–93, 2024.

Haim Avron, Kenneth L Clarkson, and David P Woodruff. Faster kernel ridge regression using sketching and preconditioning. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1116–1138, 2017.

Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory*, 2013.

Pau Batlle, Matthieu Darcy, Bamdad Hosseini, and Houman Owhadi. Kernel methods are competitive for operator learning. *Journal of Computational Physics*, 496:112549, 2024.

Mikhail Belkin. Approximation beats concentration? An approximation view on inference with smooth radial kernels. In *Conference On Learning Theory*, 2018.

Rajendra Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.

Stefan Blücher, Klaus-Robert Müller, and Stefan Chmiela. Reconstructing kernel-based machine learning force fields with superlinear convergence. *Journal of Chemical Theory and Computation*, 19(14):4619–4630, 2023.

Andrea Caponnetto and Ernesto DeVito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7:331–368, 2007.

Benjamin Charlier, Jean Feydy, Joan Alexis Glaunes, François-David Collin, and Ghislain Durif. Kernel operations on the GPU, with autodiff, without memory overflows. *Journal of Machine Learning Research*, 22(74):1–6, 2021.

Yifan Chen, Ethan N. Epperly, Joel A. Tropp, and Robert J. Webber. Randomly pivoted cholesky: Practical approximation of a kernel matrix with few entry evaluations. *Communications on Pure and Applied Mathematics*, 78(5):995–1041, 2025.

Li-Fang Cheng, Bianca Dumitrascu, Gregory Darnell, Corey Chivers, Michael Draugelis, Kai Li, and Barbara E. Engelhardt. Sparse multi-output Gaussian processes for online medical time series prediction. *BMC Medical Informatics and Decision Making*, 20(1): 152, 2020.

Kurt Cutajar, Michael Osborne, John Cunningham, and Maurizio Filippone. Preconditioning kernel matrices. In *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

Michał Dereziński and Michael W Mahoney. Determinantal point processes in randomized numerical linear algebra. *Notices of the American Mathematical Society*, 68(1):34–45, 2021.

Michał Dereziński and Jiaming Yang. Solving dense linear systems faster than via preconditioning. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1118–1129, 2024.

Michał Dereziński, Daniele Calandriello, and Michal Valko. Exact sampling of determinantal point processes with sublinear time preprocessing. In *Advances in Neural Information Processing Systems*, 2019.

Michał Dereziński, Rajiv Khanna, and Michael W Mahoney. Improved guarantees and a multiple-descent curve for column subset selection and the Nyström method. In *Advances in Neural Information Processing Systems*, 2020.

Michał Dereziński, Daniel LeJeune, Deanna Needell, and Elizaveta Rebrova. Fine-grained analysis and faster algorithms for iteratively solving linear systems. *Journal of Machine Learning Research*, 26(144):1–49, 2025a.

Michał Dereziński, Christopher Musco, and Jiaming Yang. Faster linear systems and matrix norm approximation via multi-level sketched preconditioning. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2025b.

Michał Dereziński, Deanna Needell, Elizaveta Rebrova, and Jiaming Yang. Randomized Kaczmarz methods with beyond-Krylov convergence. *arXiv preprint arXiv:2501.11673*, 2025c.

Mateo Díaz, Ethan N Epperly, Zachary Frangella, Joel A Tropp, and Robert J Webber. Robust, randomized preconditioning for kernel ridge regression. *arXiv preprint arXiv:2304.12465*, 2023.

Ethan N. Epperly, Joel A. Tropp, and Robert J. Webber. Embrace rejection: Kernel matrix approximation by accelerated randomly pivoted cholesky. *SIAM Journal on Matrix Analysis and Applications*, 46(4):2527–2557, 2025.

Zachary Frangella, Joel A. Tropp, and Madeleine Udell. Randomized Nyström preconditioning. *SIAM Journal on Matrix Analysis and Applications*, 44(2):718–752, 2023.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. In *Advances in Neural Information Processing Systems*, 2018.

Guillaume Gautier, Guillermo Polito, Rémi Bardenet, and Michal Valko. DPPy: DPP Sampling with Python. *Journal of Machine Learning Research - Machine Learning Open Source Software (JMLR-MLOSS)*, 2019.

Nidham Gazagnadou, Mark Ibrahim, and Robert M. Gower. RidgeSketch: A fast sketching based solver for large scale ridge regression. *SIAM Journal on Matrix Analysis and Applications*, 43(3):1440–1468, 2022.

Robert Gower, Filip Hanzely, Peter Richtarik, and Sebastian U Stich. Accelerated stochastic matrix inversion: General theory and speeding up BFGS rules for faster second-order optimization. In *Advances in Neural Information Processing Systems*, 2018.

Robert M Gower and Peter Richtárik. Randomized iterative methods for linear systems. *SIAM Journal on Matrix Analysis and Applications*, 36(4):1660–1690, 2015.

Helmut Harbrecht, Michael Peters, and Reinhold Schneider. On the low-rank approximation by the pivoted Cholesky decomposition. *Applied Numerical Mathematics*, 62(4):428–440, 2012.

Nicholas J Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

George S Kimeldorf and Grace Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 41(2):495–502, 1970.

Jacek Kuczyński and Henryk Woźniakowski. Estimating the largest eigenvalue by the power and lanczos algorithms with a random start. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1094–1122, 1992.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

Jihao Andreas Lin, Shreyas Padhy, Javier Antoran, Austin Tripp, Alexander Terenin, Csaba Szepesvari, José Miguel Hernández-Lobato, and David Janz. Stochastic gradient descent for Gaussian processes done right. In *International Conference on Learning Representations*, 2024.

Ji Liu and Stephen J. Wright. An accelerated randomized Kaczmarz algorithm. *Mathematics of Computation*, 85(297):153–178, 2016.

Siyuan Ma and Mikhail Belkin. Diving into the shallows: A computational perspective on large-scale shallow learning. In *Advances in Neural Information Processing Systems*, 2017.

Siyuan Ma and Mikhail Belkin. Kernel machines that adapt to GPUs for effective large batch training. In *Proceedings of Machine Learning and Systems*, 2019.

Per-Gunnar Martinsson and Joel A Tropp. Randomized numerical linear algebra: Foundations and algorithms. *Acta Numerica*, 29:403–572, 2020.

Giacomo Meanti, Luigi Carratino, Lorenzo Rosasco, and Alessandro Rudi. Kernel methods through the roof: Handling billions of points efficiently. In *Advances in Neural Information Processing Systems*, 2020.

Giacomo Meanti, Antoine Chatalic, Vladimir Kostic, Pietro Novelli, Massimiliano Pontil, and Lorenzo Rosasco. Estimating Koopman operators with sketching to provably learn large scale dynamical systems. In *Advances in Neural Information Processing Systems*, 2024.

Cameron Musco and Christopher Musco. Recursive sampling for the Nyström method. In *Advances in Neural Information Processing Systems*, 2017.

Mojmir Mutny, Michał Dereziński, and Andreas Krause. Convergence analysis of block coordinate algorithms with determinantal sampling. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

EJ Nyström. Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben. *Acta Mathematica*, 54(1):185–204, 1930.

Jonathan Parkinson and Wei Wang. Linear-scaling kernels for protein sequences and small molecules outperform deep learning while providing uncertainty quantitation and improved interpretability. *Journal of Chemical Information and Modeling*, 63(15):4589–4601, 2023.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using Gaussian processes. *Journal of Computational Physics*, 348: 683–693, 2017.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. The MIT Press, 11 2005.

Peter Richtárik and Martin Takáč. Stochastic reformulations of linear systems: Algorithms and convergence theory. *SIAM Journal on Matrix Analysis and Applications*, 41(2):487–524, 2020.

Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, 2015.

Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco. Falkon: An optimal large scale kernel method. In *Advances in Neural Information Processing Systems*, 2017.

Alessandro Rudi, Daniele Calandriello, Luigi Carratino, and Lorenzo Rosasco. On fast leverage score sampling and optimal learning. In *Advances in Neural Information Processing Systems*, 2018.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.

Annika Stuke, Milica Todorović, Matthias Rupp, Christian Kunkel, Kunal Ghosh, Lauri Himanen, and Patrick Rinke. Chemical diversity in molecular orbital energy predictions with kernel ridge regression. *The Journal of Chemical Physics*, 150(20):204121, 2019.

F. William Townes and Barbara E. Engelhardt. Nonnegative spatial factorization applied to spatial genomics. *Nature Methods*, 20(2):229–238, 2023.

Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher. Fixed-rank approximation of a positive-semidefinite matrix from streaming data. In *Advances in Neural Information Processing Systems*, 2017.

Stephen Tu, Rebecca Roelofs, Shivaram Venkataraman, and Benjamin Recht. Large scale kernel learning using block coordinate descent. *arXiv preprint arXiv:1602.05310*, 2016.

Stephen Tu, Shivaram Venkataraman, Ashia C. Wilson, Alex Gittens, Michael I. Jordan, and Benjamin Recht. Breaking locality accelerates block Gauss-Seidel. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Ke Wang, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. Exact Gaussian processes on a million data points. In *Advances in Neural Information Processing Systems*, 2019.

Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, 2000.

Anqi Wu, Samuel A. Nastase, Christopher A. Baldassano, Nicholas B. Turk-Browne, Kenneth A. Norman, Barbara E. Engelhardt, and Jonathan W. Pillow. Brain kernel: A new spatial covariance function for fMRI data. *NeuroImage*, 245:118580, 2021.

Haishan Ye, Luo Luo, and Zhihua Zhang. Nesterov's acceleration for approximate Newton. *Journal of Machine Learning Research*, 21(142):1–37, 2020.

Shipu Zhao, Zachary Frangella, and Madeleine Udell. NysADMM: Faster composite convex optimization via low-rank approximation. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.