Filomat 34:11 (2020), 3871–3884
https://doi.org/10.2298/FIL2011871T

Published by Faculty of Sciences and Mathematics, University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

On Expected Error of Randomized Nyström Kernel Regression

Aleksandar Trokicić[a], Branimir Todorović[a]

[a]*Department of Computer Science, Faculty of Sciences and Mathematics, University of Niš, Serbia*
Višegradska 33, 18000 Niš

Abstract. Kernel methods are a class of machine learning algorithms which learn and discover patterns in a high (possibly infinite) dimensional feature space obtained by often nonlinear, possibly infinite mapping of an input space. A major problem with kernel methods is their time complexity. For a data set with n input points a time complexity of a kernel method is $O(n^3)$, which is intractable for a large data set. A method based on a random Nyström features is an approximation method that is able to reduce the time complexity to $O(np^2 + p^3)$ where p is the number of randomly selected input data points. A time complexity of $O(p^3)$ comes from the fact that a spectral decomposition needs to be performed on a $p \times p$ Gram matrix, and if p is a large number even an approximate algorithm is time consuming. In this paper we will apply the randomized SVD method instead of the spectral decomposition and further reduce the time complexity. An input parameters of a randomized SVD algorithm are $p \times p$ Gram matrix and a number $m < p$. In this case time complexity is $O(nm^2 + p^2m + m^3)$, and linear regression is performed on a m-dimensional random features. We will prove that the error of a predictor, learned via this method is almost the same in expectation as the error of a kernel predictor. Aditionally, we will empirically show that this predictor is better than the ONE that uses only Nyström method.

1. Introduction

Kernel methods [14, 16, 19] are a class of machine learning algorithms that learn patterns in a high (possibly infinite) dimensional feature spaces obtained by usually nonlinear mapping of the input space: $\phi : R^D \to F$. This mapping is often unknown and computationally intractable due to the infinite dimensionality. Kernel methods produce machine learning models defined as $f(x, \lambda, b) = \sum_i \lambda k(x_i, x) + b$, which do not depend explicitly on unknown mapping, but are a linear combination of kernel functions defined on the input space which actually represent and inner product in the feature space: $k(x, y) = \phi(x)^T \phi(y)$. In this way, computationally intractable computation of the inner product in a high or infinite dimensional feature space, is substituted with the computation of a kernel function in the input space. An intrinsic part of every kernel method is the kernel matrix containing values of a kernel function, a positive semi definite matrix of size $n \times n$, when n is the number of examples. Another positive side of kernel methods is the separation of learning algorithm and the space in which the learning is performed because the choice of kernel is the

2010 *Mathematics Subject Classification*. Primary 68T10; Secondary 68Q32
Keywords. kernel regression, kernel matrix, Nyström method, randomized svd, random features
Received: 31 December 2019; Revised: 24 September 2020; Accepted: 31 December 2020
Communicated by Marko Petković
This research was supported by Ministry of Education, Science and Technological Development, Republic of Serbia, Grant No. 174013.
Email addresses: aleksandar.trokicic@pmf.edu.rs (Aleksandar Trokicić), branimir.todorovic@pmf.edu.rs (Branimir Todorović)

choice of the vector space in which the feature vectors belong. That means that we can design good kernels on input examples independently of designing good learning algorithm. Additionally, the input examples do not need to be vectors, we can define kernels on graphs, on images, on strings etc [15, 22]. However, on the negative side kernel methods usually require time complexity that is cubic in the number of data points which is too expensive for large data sets. Time complexity required just for computation of a kernel matrix is squared. Also space complexity is $O(n^2)$ which is intractable for a number of applications. Well known solution to this problem involve an approximation of a kernel matrix [2, 10].

Randomization methods which approximate kernel matrix using a random subset of an input set, represent a popular solution to both time and space complexity of kernel methods. Of these methods Nyström method achieves good results both in practice and theoretically. In Nyström method [3, 6, 20] we randomly select m (where $m < n$) columns of a kernel matrix and approximate the entire matrix based on this columns. Main advantage of this algorithm is its time complexity which is reduced to $O(nm^2 + m^3)$. Space complexity is also reduced to $O(nm)$ because it does not require the computation of the entire matrix.

The authors in [12] is proposed a fast method of computing random features which in turn gives rise to the fast learning algorithms. Specifically, an input space is mapped using some random function and its elements are called random features. This approach appears with diferent formulations in [13] [21] [7]. For example, a consequence of a Nyström method applied on a kernel matrix is an m-dimensional random feature vector computed for each input vector, and this random vectors are called random Nyström features [21]. An arbitrary number $m < n$ represents both the size of a random subset of an input set and the dimension of random feature vectors.

The main idea is that using p (where $p > m$) randomly selected columns of a kernel matrix for a construction of m-dimensional random feature vectors will produce better results then using only m columns, all the while keeping time complexity linear in n. This is in contrast to the Nyström method which uses m selected columns to derive m-dimensional features. Authors from [8] used this idea, and combined the Nyström method with a randomized SVD [5]. Using only Nyström method will require performing SVD on a $p \times p$ symmetric submatrix of a kernel matrix and it will produce p-dimensional random features. However if a randomized SVD is applied as in [8] algorithm will produce m-dimensional random features. This also allows the algorithm to keep the time complexity linear in n. In this paper we will perform theoretical analysis of this approximation method as applied on a least squares regression problem. Additionally, we will show theoretically that this algorithm with sub quadratic complexity exhibits the same predictive performance as the kernel regression. We will show that we can choose p so that the expected error of approximate kernel regression is approximately the same as the kernel regression error. Furthermore, we demonstrate on real world data sets that m-dimensional random features derived from p randomly selected input points produce better results than random Nyström features.

This paper is structured as follows. In the Section II, we review the method of random Nyström features with and without a randomized SVD and its application to linear regression. Proof that the estimator learned on random features defined in the section II is close to the kernel estimator learned on the original input set is presented in the Section III. Finally, experimental results and conclusion are described in the final two sections.

2. Fast kernel regression

In this section we will describe the algorithm for fast approximate kernel regression. Let us assume that the input data set is the following set:

$$T = \{(x_i \in \mathbf{R}^d, y_i \in \mathbf{R})\}_{i=\overline{1,n}} \tag{1}$$

The algorithm consists of two main steps. In the first step each feature vector is mapped into a m-dimensional random feature vector, called random view. In the second step linear regression is applied. Now we will briefly explain a Nyström method, used for a construction of m-dimensional random features and its extension that combines it with a randomized SVD.

2.1. Nyström features

Random features are low dimensional vectors derived from an input set using some random mapping. Every input vector is mapped into its corresponding random feature vector. Several types of random features are present in the literature such as Random Fourier features [12] or random Nyström features [21]. Their main advantage is the usage of kernels with time complexity linear in the number of points.

Assume that we have data set $\{(x_i \in \mathbf{R}^d)_{i=1}^n\}$ and a kernel (positive semi definite function) $k : \mathbf{R}^d \times \mathbf{R}^d \to \mathbf{R}$. Gram matrix $K_{ij} = k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$ represents a $n \times n$ positive semi definite matrix. Function $\Phi(x)$ maps data from $\mathbf{R}^d$ into the high dimensional feature space. In random feature method, every input vector x_i is mapped into the m-dimensional random feature vector r_i so that $r_i^T r_j$ approximates $k(x_i, x_j)$. Nyström method is a random matrix approximation method and when applied on a Gram matrix K its consequence are random features. Specifically for a given $m \ll n$ Nyström algorithm samples m data vectors $\{\hat{x}_i\}_{i=\overline{1,m}}$ from the input set. Approximation matrix $\tilde{K}$ is computed in the following way:

$$K \approx \tilde{K} := k(x_{1:n}, \hat{x}_{1:m}) k(\hat{x}_{1:m}, \hat{x}_{1:m})^+ k(x_{1:n}, \hat{x}_{1:m})^T \tag{2}$$

where $k(\hat{x}_{1:m}, \hat{x}_{1:m})^+ = \hat{V}\hat{D}^{-1}\hat{V}^T$ is a pseudo inverse of $k(\hat{x}_{1:m}, \hat{x}_{1:m})$, where columns of $\hat{V}$ are eigenvectors and diagonal elements of matrix $\hat{D}$ are eigenvalues of the matrix $k(\hat{x}_{1:m}, \hat{x}_{1:m})$. We define a random feature vector r_i in the following way

$$r_i = \hat{D}^{-1/2}\hat{V}^T k(x_{1:n}, \hat{x}_{1:m})^T. \tag{3}$$

According to expression (2) $\tilde{K}_{ij} = r_i^T r_j$. Therefore, random vector r_i approximates $\Phi(x_i)$. Furthermore, authors in [11] called the vector r_i random Nyström feature vector or random Nyström view. Mapping $z(x_i) = r_i$ is called a random Nyström mapping.

Training linear regression on this data set is the same as training it on the kernel regression model, where instead of the kernel matrix its Nyström approximation is used.

Algorithm 1 NF algorithm. [5, 11]

```
1:  procedure SC($T, k, m$)
2:      Input: data set $T = \{(x_i \in \mathbf{R}^d, y_i \in \mathbf{R})\}_{i=\overline{1,n}}$ ;
3:      Input: kernel $k$ ;
4:      Input: number of sampled feature vectors for Nyström method $m$ ;
5:      Sample a random permutation $perm$ from $\overline{1,n}$;
6:      for $i = \overline{1,m}$ do
7:          $\hat{x}_i = x_{perm_i}$; (* Sampling $m$ data vectors $\{\hat{x}_i\}_{i=\overline{1,m}}$; *)
8:      end for
9:      $(\hat{V}, \hat{D}) = eig(k(\hat{x}_{1:m}, \hat{x}_{1:m}))$; (* Perform eigendecomposition of $k(\hat{x}_{1:m}, \hat{x}_{1:m}) = \hat{V}\hat{D}\hat{V}^T$ *)
10:     for $i = \overline{1,n}$ do
11:         $r_i = \hat{D}^{-1/2}\hat{V}^T k(x_{1:n}, \hat{x}_{1:m})^T$; (* Nyström features *)
12:     end for
13:     (* Lines 9–12 will be modified in our algorithm 2 *)
14:     Perform linear regression on $(r_i, y_i)$   $i = \overline{1,n}$;
15:     Output: Linear regressor.
16: end procedure
```

2.2. Randomized eigenvalue decomposition

Now we will briefly explain randomized decomposition that is used in an extension of the Nyström method. Assume that we have a real symmetric matrix $W \in \mathbf{R}^{p \times p}$. In our paper we will apply this algorithm on matrix derived during Nyström method from sampled rows and columns ($W = k(\hat{x}_{1:p}, \hat{x}_{1:p})$ where $\hat{x}_{1:p}$ are p sampled data points). Our goal is to perform an eigenvalue decomposition of W. Time complexity of eigenvalue decomposition is $O(p^3)$. Using the algorithm from [5] approximate decomposition can be computed with time complexity of $O(pm^2 + m^3)$ with the use of a random matrix of dimension $p \times m$ where m is an input parameter. Randomized method for eigenvalue decomposition consists of several steps:

- Generate Gaussian random matrix $\Omega \in \mathbf{R}^{p\times(m+l)}$, where l is a small oversampling parameter (usually 5).
- Construct a matrix $Y = W\Omega \in \mathbf{R}^{p\times(m+l)}$. Oversampling parameter l is chosen because it increases the chances for the matrix Y to span the m-dimensional subspace of W.
- We perform a QR decomposition on a matrix Y. Matrix Q gives us the following approximation $W \approx QQ^TW$ from which follows $W \approx QQ^TWQQ^T$.
- Generate a matrix $B = Q^TWQ$ and perform eigenvalue decomposition on a matrix $B = U\Lambda U^T$.
- Columns of QV are approximate eigenvectors of W and diagonal elements of Λ are approximate eigenvalues of W.
- A matrix V is derived from approximate eigenvectors of W associated with m largest approximate eigenvalues Therefore $W = V\Lambda_m V^T$ where Λ_m contains largest approximate eigenvalues.

In the next section we will show how to apply this algorithms into the Nyström method. Instead of sampling m data vectors from input data set we sample $p > m$ data vectors and perform combination of a Nyström method and a randomized eigenvalue decomposition to derive m-dimensional random feature vectors.

2.3. Random feature vectors using a combination of a Nyström method and a randomized eigenvalue decomposition

Assume that we have data set $\{(x_i \in \mathbf{R}^d)_{i=1}^n\}$ and a kernel (positive semi definite function) $k : \mathbf{R}^d \times \mathbf{R}^d \to \mathbf{R}$. For a given $p \ll n$ Nyström algorithm samples p feature vectors $\{\hat{x}_i\}_{i=\overline{1,l}}$ from the input set. Recall that from a Nyström method, the following matrix is an approximation of a Gram matrix ($K_{ij} = k(x_i, x_j)$):

$$\tilde{K} := k(x_{1:n}, \hat{x}_{1:p})k(\hat{x}_{1:p}, \hat{x}_{1:p})^+ k(x_{1:n}, \hat{x}_{1:p})^T.$$

From it we derive p-dimensional Nyström features $r_i = \hat{D}^{-1/2}\hat{V}^T k(x_{1:n}, \hat{x}_{1:p})^T$. However our goal is to map input vectors into the m-dimensional space while steel using p-sampled data points, where $p > m$. Even though we add another approximation, we wish to keep the main property of random Nyström features, that for each error ϵ there is a large enough p so that the error achieved by the m-dimensional feature vectors (derived from p sampled points) is smaller than ϵ.

In order to produce m-dimensional random vectors we propose to use the randomized eigenvalue decomposition for the computation of the pseudo inverse and therefore kernel matrix approximation as in [8]:

$$K \approx L := k(x_{1:n}, \hat{x}_{1:p})k(\hat{x}_{1:p}, \hat{x}_{1:p})^* k(x_{1:n}, \hat{x}_{1:p})^T$$

where $k(\hat{x}_{1:p}, \hat{x}_{1:p})^* = \hat{V}\hat{D}^{-1}\hat{V}^T$ is an approximate pseudo inverse of $k(\hat{x}_{1:p}, \hat{x}_{1:p})$, where columns of $\hat{V} \in \mathbb{R}^{p\times m}$ are approximate eigenvectors and diagonal elements of matrix $\hat{D} \in \mathbb{R}^{m\times m}$ are approximate eigenvalues of the matrix $k(\hat{x}_{1:p}, \hat{x}_{1:p})$. Approximate eigenvalues and eigenvectors are computed using randomized SVD (2.2). We will call this matrix a RNyström kernel matrix, and this method a RNyström method or more specifically RNyström (p, m) method. Therefore random features which we will call RNyström features are computed as follows:

$$r_i = \hat{D}^{-1/2}\hat{V}^T k(x_{1:n}, \hat{x}_{1:p})^T$$

Finally we apply linear regression algorithm which we will call RNyström kernel regression. Putting it all together we get an algorithm (2).

In the following section, theorem (3.7), we show that the approximate predictor (predictor learned on random features) is close enough to the original one (the best overall predictor).

Algorithm 2 RNF algorithm.

```
1: procedure SC($T, k, p, m, l$)
2:     Input: data set $T = \{(x_i \in \mathbf{R}^d, y_i \in \mathbf{R})\}_{i=\overline{1,n}}$ ;
3:     Input: kernel $k$ ;
4:     Input: a number of sampled feature vectors $p$, and a random feature vector dimension $m$ ;
5:     Sample a random permutation $perm$ from $\overline{1,n}$;
6:     for $i = \overline{1,p}$ do
7:         $\hat{x}_i = x_{perm_i}$; (* Sampling $m$ data vectors $\{\hat{x}_i\}_{i=\overline{1,p}}$; *)
8:     end for
9:     Compute $\Omega \in \mathbb{R}^{p\times(m+l)}$ random Gaussian matrix;
10:    $W = k(\hat{x}_{1:p}, \hat{x}_{1:p})$;
11:    $Y = W\Omega$;
12:    $(Q, R) = qr(Y)$; (* Compute QR decomposition of $Y$ *)
13:    $B = Q^T W Q$;
14:    $(U, \Lambda) = eig(B)$; (* Compute eigendecomposition of a matrix $B = U\Lambda U^T$ *)
15:    $\hat{V} = QU$;
16:    $\hat{D} = \Lambda$;
17:    for $i = \overline{1,n}$ do
18:        $r_i = \hat{D}^{-1/2}\hat{V}^T k(x_{1:n}, \hat{x}_{1:p})^T$; (* RNyström features *)
19:    end for
20:    (* Lines 9–19 are modified lines 9–12 from the algorithm 1 *)
21:    Perform linear regression on $(r_i, y_i)$   $i = \overline{1,n}$;
22:    Output:  Linear regressor.
23: end procedure
```

3. Analysis of an approximate error

In the case of a regression with RNyström features, the original kernel matrix is replaced with the RNyström kernel matrix L. We are not comparing directly the approximation to the original kernel matrix but instead we are analyzing the quality of the regression (i. e. in sample error) that uses this approximation. We will prove that the expected in sample error of the RNyström method (expectation of $\|(L + n\lambda I)^{-1}Lz - z\|^2$,with the respect to the distribution of the approximation) is almost the same as the in-sample error of the kernel regression $\|(K + n\lambda I)^{-1}Kz - z\|^2$, where L is the RNyström approximation of K.

We first list two theorems which we will use to prove the lemma on bound of the approximate matrix product.

Theorem 3.1. *[4] Let $\mathcal{C}$ be a finite set of positive semi-definite matrices of dimension n. Sample $(X_1, \ldots, X_p)$ from $\mathcal{C}$ uniformly at random without replacement and sample $(Y_1, \ldots, Y_p)$ from $\mathcal{C}$ uniformly at random with replacement. Now let f be a convex function on $\mathcal{C}$. Then:*

$$E_X[f(\sum_i X_i)] \le E_Y[f(\sum_i Y_i)]$$

Theorem 3.2. *(**Matrix Bernstein**) [9] Consider an independent sequence $(Y_k)_{k\ge1}$ of symmetric positive semi-definite random matrices that satisfy*

$$EY_k = 0 \text{ and } \|Y_k\| \le R \quad \text{for each index } k$$

Let $\sigma^2 = \|\sum_k EY_k^2\|$. Then for all $t \ge 0$,

$$P(\lambda_{max}(\sum_k Y_k) \ge t) \le d \exp\Big(\frac{-t^2}{3\sigma^2 + 2Rt}\Big)$$

$$E\Big[\lambda_{max}(\sum_k Y_k)\Big] \leq \sigma\sqrt{3\log d} + R\log d$$

Now, we define the lemma that analysis the approximation of the matrix product. For an n-dimensional matrix $\frac{1}{n}\Psi^T\Psi$ and its approximation $\frac{1}{p}\Psi_S^T\Psi_S^T$ where Ψ_S only contains p chosen columns of Ψ, lemma gives the bound on an expectation of the spectral norm of their difference. We will apply this result in the proof of the theorem (3.7), where we will use it to proove the monotonicity of the regularised RNyström approximation of kernel matrix.

The lemma setting is very similar to the [1], however the prof is slightly different, and, in our opinion, simpler. We used the newer version of Bernstein theorem from [9], while in [1] theorem from [17, 18] is used. Also instead of the result from [17, 18] we used the theorem (3.1).

Lemma 3.3. *Let $\Psi \in \mathbb{R}^{n\times r}$ be such that for every row i is $\|\Psi(i,:)\| \leq R$. Let $S \subset \{1, 2, \ldots, n\}$ be a random subset with elements chosen at random without replacement. Then*

$$E[\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T\Psi_S^T] \leq \sqrt{\frac{R^2}{p}\lambda_{max}(\frac{1}{n}\Psi^T\Psi)3\log n} + \frac{1}{p}\lambda_{max}(\frac{1}{n}\Psi^T\Psi)\log n$$

Proof. We consider the matrix $A = \frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T\Psi_S^T = \frac{1}{n}\sum_{i=\overline{1,n}}\psi_i\psi_i^T - \frac{1}{p}\sum_{i\in S}\psi_i\psi_i^T$ where ψ_i represents the i-th row of a matrix Ψ. For the matrix A we know that $E[A] = 0$. Let $B = \frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_J^T\Psi_J^T$ be a random matrix derived in the similar way as matrix A with one difference: $J \subset \{1, 2, \ldots, n\}$ is a random subset with elements chosen at random with replacement. Therefore the matrix B can be expressed in the following way $B = \frac{1}{n}\sum_{i=\overline{1,n}}\psi_i\psi_i^T - \frac{1}{p}\sum_{i\in J}\psi_i\psi_i^T = \frac{1}{n}\sum_{i=\overline{1,n}}\psi_i\psi_i^T - \frac{1}{p}\sum_{i=\overline{1,n}}\sum_{j=\overline{1,p}} z_i^j\psi_i\psi_i^T$ where $(P(z_i^j) = \frac{1}{n})(\forall i, j)$
From the theorem 3.1 and the fact that every norm is a convex function we have:

$$E[\|A\|] \leq E[\|B\|]$$

We use this result because of its simplicity while in [1] they used the result from [17, 18]. We can write the matrix B in the following form:

$$\begin{aligned} B &= \frac{1}{n}\sum_{i=\overline{1,n}}\psi_i\psi_i^T - \frac{1}{p}\sum_{i=\overline{1,n}}\sum_{j=\overline{1,p}} z_i^j\psi_i\psi_i^T \\ &= \frac{1}{n}\frac{1}{p}\sum_{i=\overline{1,n}}\sum_{j=\overline{1,p}}\psi_i\psi_i^T - \frac{1}{p}\sum_{i=\overline{1,n}}\sum_{j=\overline{1,p}} z_i^j\psi_i\psi_i^T \\ &= \sum_{j=\overline{1,p}} M_j \end{aligned} \tag{4}$$

where $M_j = \frac{1}{p}\sum_{i=\overline{1,n}} z_i^j(\frac{1}{n}\Psi^T\Psi - \psi_i\psi_i^T)$ Here we used the newer result from [9], while in [1] the result from [17, 18] is used. According to [1] follows

$$\begin{aligned} &E[M_j] = 0 \\ &\|\sum_{j\in\overline{1,p}} E[M_j^2]\| = \frac{R^2}{p}\lambda_{max}(\frac{1}{n}\Psi^T\Psi) \end{aligned} \tag{5}$$

Now we can apply the theorem (3.2) to obtain:

$$E[\|B\|] \leq \sqrt{\frac{R^2}{p}\lambda_{max}(\frac{1}{n}\Psi^T\Psi)3\log n} + \frac{1}{p}\lambda_{max}(\frac{1}{n}\Psi^T\Psi)\log n$$

from which follows the desired bound. □

Now, we will prove a lemma about the bound of the eigenvalues of a matrix necessary for the proof of the theorem on bounds of in sample error.This lemma shows that the matrix defined in a special way is smaller than the identity matrix. In the proof of the theorem (3.7) a matrix, derived from a kernel matrix, will have the structure that satisfies the following lemma, so we will use lemma's result to prove the monotonicity of the regularized RNyström approximation.

Lemma 3.4. *Let $\Phi \in \mathbb{R}^{n\times n}$ and $\gamma > 0$. Then for a matrix $\Psi = \Phi(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}$:*

$$\frac{1}{n}\Psi^T\Psi \preceq I$$

Proof. From $\gamma > 0$

$$\frac{1}{n}\Phi^T\Phi \preceq \frac{1}{n}\Phi^T\Phi + \gamma I$$

From the fact that for any matrices X, Y, Z from $X \preceq Y$ follows $Z^TXZ \preceq Z^TYZ$ we have:

$$(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}\frac{1}{n}\Phi^T\Phi(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}T} \preceq (\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}(\frac{1}{n}\Phi^T\Phi + \gamma I)(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}T}$$

$$\frac{1}{n}(\Phi(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}})^T\Phi(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}} \preceq I$$

$$\frac{1}{n}\Psi^T\Psi \preceq I$$

□

Corollary 3.5. *Let $\Phi \in \mathbb{R}^{n\times n}$ and $\gamma > 0$. Then from a matrix $\Psi = \Phi(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}$ and for a random subset $S \subset \{1, 2, \ldots, n\}$ follows that every eigenvalue of a matrix $\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T\Psi_S \preceq I$ is from a segment $[0, 1]$.*

Since the RNyström approximation of the kernel matrix extends the Nyström method with the application of the randomized SVD we will list here a theorem on the bound of the expected error of randomized SVD:

Theorem 3.6. *[8] In a randomized SVD (2.2) of a matrix $W \in \mathbb{R}^{p\times p}$, where we compute k approximate eigenvalues and where l is an oversampling parameter (typically a small arbitrary number), the expected error $||W - QQ^TW||$ (with respect to the randomness in the Gaussian random matrix) is upper bounded by*

$$(\sqrt{\frac{k}{l-1}} + \frac{e\sqrt{k+l}}{l\sqrt{p-k}})||WW^T||$$

Now we have listed all lemmas and theorems that are necessary for the proof of a our main result. We will extend the theorem from [1] because we are extending the Nyström approximation with the randomized SVD. Both Nyström approximation and RNyström approximation use the p randomly sampled columns to approximate the matrix.

he in sample error of the RNystomr kernel regression can be arbitrarily close to the in sample error of the original kernel regression, by selecting proper lower bound on the number of sampled columns. That means that if we substitute kernel matrix in a kernel regerssion with an approximate one we get approximately the same predictor. For example, the bound on the number of selected columns on a data set of 1000 elements (a subset of a calHousing data set, see table (1)) with an error $\delta = 0.1$ is around a 100.

Theorem 3.7. *Let $\lambda > 0$ and let $z \in \mathbb{R}^n$ and $K \in \mathbb{R}^{n\times n}$ be a vector of output observations and a kernel matrix derived from input data points respectively. Assume $d = n||diag(K(K + n\lambda I)^{-1})||_\infty$ and $R^2 = ||K||$. Define the estimate $z_K = (K + n\lambda I)^{-1}Kz$. Assume S is a uniform random subset of $p > k$ indices in $\{1, 2, \ldots, n\}$ and consider L as approximate kernel matrix based on RNyström (p, k) method and an oversampling parameter l, with the approximate estimate $z_L = (L + n\lambda I)^{-1}Lz$. For every $\delta \in (0, 1)$ and $p \geq 2\Big(\log n\sqrt{\frac{6d\log n}{10\delta}} + \frac{tR^2\lambda\delta}{2}\Big)$ for $t = 1 + \sqrt{\frac{k}{l-1}} + \frac{e\sqrt{k+l}}{l}$ the following is true*

$$\frac{1}{n}E[||z_L - z||^2] \leq (1 + 6\delta)\frac{1}{n}||z - z_K||^2$$

Proof. Because K is a kernel matrix there exist a matrix $\Phi \in \mathbb{R}^{n\times n}$ such that $K = \Phi\Phi^T$. Approximate kernel matrix L based on a combination of a Nyström method and randomized eigenvalue decomposition can be written in the following way $L = K(:,S)Q(Q^TK(S,S)Q)^{-1}Q^TK(S,:)$ where Q is derived from a QR decomposition on a matrix $K(S,S)\Omega$ (where Ω is a Gaussian random matrix of $p \times (l+k)$ dimension) as in RSVD 2.2. Matrix $K(:,S)$ can be written in the following way $K(:,S) = \Phi\Phi_S^T$ where $\Phi_S = \Phi(S,:)$. Let

$$Ł_\gamma = \Phi\Phi_S^TQ(Q^T\Phi_S\Phi_S^TQ + p\gamma I)^{-1}Q^T\Phi_S\Phi^T$$

be a regularized kernel matrix approximation. We can write $L_\gamma = \Phi N_\gamma \Phi^T$ where $N_\gamma = \Phi_S^TQ(Q^T\Phi_S\Phi_S^TQ + p\gamma I)^{-1}Q^T\Phi_S$. Using Sherman—Morrison—Woodbury identity we get:

$$\begin{aligned} N_\gamma &= \Phi_S^TQ(Q^T\Phi_S\Phi_S^TQ + p\gamma I)^{-1}Q^T\Phi_S \\ &= \Phi_S^TQ(Q^T\Phi_S(Q^T\Phi_S)^T + p\gamma I)^{-1}Q^T\Phi_S \\ &= (Q^T\Phi_S)^T(Q^T\Phi_S)((Q^T\Phi_S)^T(Q^T\Phi_S) + p\gamma I)^{-1} \\ &= \Phi_S^TQQ^T\Phi_S(\Phi_S^TQQ^T\Phi_S + p\gamma I)^{-1} \\ &= I - \gamma(\frac{1}{p}\Phi_S^TQQ^T\Phi_S + \gamma I)^{-1} \end{aligned} \tag{6}$$

Using Sherman–-Morrison—–Woodbury identity approximate in sample error can be computed in the following way:

$$\begin{aligned} \frac{1}{n}\|z - z_{L_\gamma}\| &= \frac{1}{n}\|z - (L_\gamma + n\lambda I)^{-1}L_\gamma z\| \\ &= n\lambda^2\|(L_\gamma + n\lambda I)^{-1}z\| \\ &= n\lambda^2 z^T(\Phi N_\gamma \Phi^T + n\lambda I)^{-2}z \end{aligned} \tag{7}$$

Both function $\gamma \to N_\gamma$ and in sample prediction error are matrix non decreasing functions. Therefore in order to find an upper bound for on error $\|z - z_L\|^2$ it is enough to find an upper bound for $\|z - z_{L_\gamma}\|$ for any $\gamma > 0$ because $L = L_0$. Furthermore in order to find an upper bound for $\|z - z_{L_\gamma}\|$ for any $\gamma > 0$ it is enough to find a matrix lower bound for N_γ.

Let $\Psi = \Phi(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}$ and matrix derived by sampling rows of Ψ is $\Psi_S = \Phi_S(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}$. This implies:

$$\begin{aligned} \Psi^T\Psi &= (\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}\Phi^T\Phi(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}} \\ \Psi_S^T\Psi_S &= (\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}\Phi_S^T\Phi_S(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}} \\ \Psi_S^TQQ^T\Psi_S &= (\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}\Phi_S^TQQ^T\Phi_S(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}} \end{aligned} \tag{8}$$

Applying this to the expression of N_γ we get for $X = (\frac{1}{n}\Phi^T\Phi + \gamma I)^{\frac{1}{2}}$

$$\begin{aligned}
N_\gamma &= I - \gamma(\frac{1}{p}\Phi_S^T QQ^T\Phi_S + \gamma I)^{-1} \\
&= I - \gamma(\frac{1}{p}\Phi_S^T QQ^T\Phi_S + \gamma I + \frac{1}{n}\Phi^T\Phi - \frac{1}{n}\Phi^T\Phi)^{-1} \\
&= I - \gamma(XX - \frac{1}{n}XX^{-1}\Phi^T\Phi X^{-1}X + \frac{1}{p}XX^{-1}\Phi_S^T QQ^T\Phi_S X^{-1}X)^{-1} \\
&= I - \gamma(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}(I - \frac{1}{n}\Psi^T\Psi + \frac{1}{p}\Psi_S^T QQ^T\Psi_S)^{-1}(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}} \qquad (9)
\end{aligned}$$

We define $t_{S,Q} = \lambda_{max}(\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T QQ^T\Psi_S)$. According to the corollary (3.5) we know that $0 \le t_{S,Q} \le 1$. This implies:

$$\begin{aligned}
\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T QQ^T\Psi_S &\le t_{S,Q}I \\
(1 - t_{S,Q})I &\le I - \frac{1}{n}\Psi^T\Psi + \frac{1}{p}\Psi_S^T QQ^T\Psi_S \\
(I - \frac{1}{n}\Psi^T\Psi + \frac{1}{p}\Psi_S^T QQ^T\Psi_S)^{-1} &\le \frac{1}{1 - t_{S,Q}}I \qquad (10)
\end{aligned}$$

From this it follows that

$$\begin{aligned}
I - N_\gamma &= \gamma(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}}(I - \frac{1}{n}\Psi^T\Psi + \frac{1}{p}\Psi_S^T QQ^T\Psi_S)^{-1}(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-\frac{1}{2}} \\
&\le \frac{\gamma}{1 - t_{S,Q}}(\frac{1}{n}\Phi^T\Phi + \gamma I)^{-1} \qquad (11)
\end{aligned}$$

From this we obtain that $K - L_\gamma = \Phi(I - N_\gamma)\Phi^T \le \frac{n\gamma}{1-t_{S,Q}}I$. Assume that $\frac{\gamma}{\lambda(1-t_{S,Q})} \le 1$, and we get

$$(L_\gamma + n\lambda I)^{-1} \le (1 - \frac{\gamma}{\lambda(1 - t_{S,Q})})^{-1}(K + n\lambda I)^{-1}$$

Applying this to the in sample approximate error we get

$$\begin{aligned}
\frac{1}{n}\|z - z_{L_\gamma}\|^2 &= \lambda^2 z^T(L_\gamma + n\lambda I)^{-2}z \\
&\le n\lambda^2(1 - \frac{\gamma}{\lambda(1 - t_{S,Q})})^{-2}z^T(K + n\lambda I)^{-2}z \\
&= (1 - \frac{\gamma}{\lambda(1 - t_{S,Q})})^{-2}\frac{1}{n}\|z - z_K\|^2 \qquad (12)
\end{aligned}$$

Using the fact that $(1 - \frac{\epsilon}{2})^{-2} \le 1 + 3\epsilon$ where $\epsilon \in (0, 1)$ we have

$$\frac{1}{n}\|z - z_{L_\gamma}\|^2 \le (1 + 6\frac{\gamma}{\lambda(1 - t_{S,Q})})\frac{1}{n}\|z - z_K\|^2$$

This implies:

$$\frac{1}{n}E[\|z - z_{L_\gamma}\|^2] \leq (1 + 6\frac{\gamma}{\lambda(1 - E[t_{S,Q}])})\frac{1}{n}\|z - z_K\|^2$$

Now we need to find an upper bound for $E[t_{S,Q}]$.

$$\begin{aligned} E[t_{S,Q}] &= E[\|\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T QQ^T\Psi_S\|] \\ &= E[\|\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T\Psi_S + \frac{1}{p}\Psi_S^T\Psi_S - \frac{1}{p}\Psi_S^T QQ^T\Psi_S\|] \\ &\leq E[\|\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T\Psi_S\|] + \frac{1}{p}E[\|\Psi_S^T\Psi_S - \Psi_S^T QQ^T\Psi_S\|] \end{aligned} \tag{13}$$

Now find an upper bound on $E[\|\Psi_S^T\Psi_S - \Psi_S^T QQ^T\Psi_S\|]$:

$$\begin{aligned} E[\|\Psi_S^T\Psi_S - \Psi_S^T QQ^T\Psi_S\|] &\leq E[\|(I - QQ^T)\Psi_S\Psi_S^T\|] \\ &= E_S[E_Q[(I - QQ^T)\Psi_S\Psi_S^T]] \\ &\leq E_S[(1 + \sqrt{\frac{k}{l-1}} + \frac{e\sqrt{k+l}}{l\sqrt{p-k}})\frac{1}{\gamma}\|\Phi_S\Phi_S^T\|] \end{aligned} \tag{14}$$

The last inequality is derived using theorem (3.6).

$$\begin{aligned} E[\|\Psi_S^T\Psi_S - \Psi_S^T QQ^T\Psi_S\|] &\leq \frac{1}{\gamma}(1 + \sqrt{\frac{k}{l-1}} + \frac{e\sqrt{k+l}}{l\sqrt{p-k}})E_S[\|\Phi_S\Phi_S^T\|] \\ &\leq \frac{1}{\gamma}(1 + \sqrt{\frac{k}{l-1}} + \frac{e\sqrt{k+l}}{l})\|\Phi\Phi^T\| \end{aligned} \tag{15}$$

Applying lemma (3.3) and using ([1]) for $\gamma \leq \lambda$ we get:

$$E[\|\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T\Psi_S\|] \leq \sqrt{\frac{1}{p}\lambda\gamma^{-1}d3\log n\frac{1}{p}\log n}$$

$$E[\|\frac{1}{n}\Psi^T\Psi - \frac{1}{p}\Psi_S^T\Psi_S\|] \leq \sqrt{\frac{3}{k+1}\lambda\gamma^{-1}d\log n\frac{1}{p}\log n}$$

Combining final two results we have

$$E[t_{S,Q}] \leq \sqrt{\frac{3}{k+1}\lambda\gamma^{-1}d\log n\frac{1}{p}\log n} + \frac{1}{p\gamma}(1 + \sqrt{\frac{k}{l-1}} + \frac{e\sqrt{k+l}}{l})R^2.$$

Plugging in $t = 1 + \sqrt{\frac{k}{l-1}} + \frac{e\sqrt{k+l}}{l}$ we get

$$E[t_{S,Q}] \leq \sqrt{\frac{3}{k+1}\lambda\gamma^{-1}d\log n\frac{1}{p}\log n} + \frac{tR^2}{p\gamma}.$$

For $p \geq 2\Big(\log n\sqrt{\frac{6d\log n}{10\delta}} + \frac{tR^2\lambda\delta}{2}\Big)$ and for $\gamma = \frac{\lambda\delta}{2}$ (for which is $\gamma \leq \lambda$) we get

$$E[t_{S,Q}] \leq \frac{1}{2}$$

Since $\frac{\gamma}{\lambda(1-E[t_{S,Q}])} \leq \delta \leq 1$ it follows

$$\frac{1}{n}E[||z - z_{L_\gamma}||^2] \leq (1 + 6\frac{\gamma}{\lambda(1 - E[t_{S,Q}])})\frac{1}{n}||z - z_K||^2$$

from which the desired result is derived $\frac{1}{n}E[||z - z_{L_\gamma}||^2] \leq (1 + 6\delta)\frac{1}{n}||z - z_K||^2$. □

4. Experimental results

Table 1: Data sets

Data Set	**Instances**	**Attributes**
cal housing [1)]	7154	40
abalone [2)]	2089	6
bank8 [1)]	4096	8
bank32 [1)]	4096	32
sarcos2 [3)]	44484	21
sarcos3 [3)]	44484	21
sarcos4 [3)]	44484	21
sarcos5[3)]	44484	21
sarcos6 [3)]	44484	21
sarcos7 [3)]	44484	21
red wine quality [2)]	4898	12
white wine quality [2)]	4898	12
Combined Cycle Power Plant Data Set [2)]	9568	4
Facebook Comment Volume Dataset [2)]	40949	54
Gas Turbine CO and NOx Emission [2)]	36733	11
SGEMM GPU kernel performance [2)]	241600	18
Superconductivty Data [2)]	21263	81
census8H [1)]	22784	8
census16H [1)]	22784	16
kin [1)]	8192	33
pumadynH [1)]	8192	33
pumadynM [1)]	8192	33

In this section we evaluate the performance of our **RNF** (derived from a RNyström method, algorithm 2) against **NF** (derived from a Nyström method, algorithm 1).

We compare algorithms on real world data sets [1)] [3)] [2)], see table (1):

- Abalone [1)] data set can be used to train a model for deriving age prediction from the abalon's measurements. The number of rings of the shell's cut that are seen through the microscope, determines the age of the abalone. However, since this is a time consuming process, this dataset contains other measurements, which are easier to obtain, and which are then used in order to predict the age.

- CalHousing [1)] data set is used to train model for predicting the household price in California. It contains the following attributes median house value, median income, housing median age, total rooms, total bedrooms, population, households, latitude, and longitude.

[1)] http://www.dcc.fc.up.pt/ ltorgo/Regression/DataSets.html
[2)] https://archive.ics.uci.edu/ml/datasets.html
[3)] http://www.gaussianprocess.org/gpml/data/

- Bank8 (8 continuous attributes) and bank32 (32 continues attributes) [1)] is a family of datasets synthetically generated from a simulation of how bank customers choose their banks. The model is trained to predict the rejection value, which is the fraction of people that leave the bank because all the open tellers have full queues.

- The sarcos data [3)] relates to an inverse dynamics problem for a locomotion of a SARCOS anthropomorphic robot arm. The model is trained to predict joint torques from a a 21-dimensional input space (7 joint positions, 7 joint velocities, 7 joint accelerations). *Sarcos-i* refers to the prediction of the *i*-the joint torque.

- Wine quality refers to two datasets that are related to the red and white variants of the Portuguese "Vinho Verde" wine. The model is trained to predict wine quality from only physicochemical (inputs) data. Grape types, wine brand, wine selling price, etc. data are not located in the data set.

- The Combined Cycle Power Plant [2)] dataset is formed by collecting data from a Combined Cycle Power Plant over 6 years, when the power plant was set to work with the full load. Attributes are ambient variables Temperature (T), Ambient Pressure (AP), Relative Humidity (RH) and Exhaust Vacuum (V) collected per hour and the predicted value is the net hourly electrical energy output (EP) of the plant.

- The dataset Gas Turbine CO and NOx Emission [2)] comes from the same power plant. However, this data contains instances of 11 attributes which are sensor measures aggregated over an hour and are used to predict the gas emissions.

- Facebook Comment Volume Dataset [2)] contains features extracted from the facebook posts and the predicted attribute is the number of comments the post will receive.

- SGEMM GPU kernel performance [2)] data set contains the measurement of the running time of a computation of a matrix-matrix product $AB = C$. Parameterized SGEMM GPU kernel is used and tested on all feasible parameter combinations. All used matrices are the size 2048 x 2048. For each tested combination, 4 runs were performed and their results are reported in milliseconds as the 4 last columns. We are using parameters as an input and the model is trained to predict the average running time of the 4 reported times.

- Superconductivty Dataset [2)] contains 81 features extracted from 21263 superconductors and in the 82nd column a critical temperature is stored. The model is trained to predict the critical temperature.

- Census [1)] data set refers to two data sets census8(8 continuous attributes) and census16(16 continuous attributes) was designed analysing the data from the US Census Bureau. The model is trained to predict the median price of the house .

- Kin (Kinematics) [1)] data set is collected by simulating the forward kinematics of an 8 link robot arm. There are various variants of this data set and we are using the one with 32 attributes, which is known to be highly non-linear and highly noisy.

- Pumadyn [1)] data set is chosen from the family of datasets synthetically generated from a realistic simulation of the dynamics of a Unimation Puma 560 robot arm. There are various variants of this data set and we are using the one with 32 attributes, with 3 variants: highly non-linear and highly noisy (H), or highly non-linear and medium noisy (M).

Some of the data sets are already divided into a training set and a test set.Those that are not we split in the following way: first 80% of the data are used training and the remaining 20% for testing. We used Gaussian kernel whose hyper parameters are selected using 5-fold cross validation for a 1000 labeled examples of the training set. Dimension of random features is 10. In **NF** algorithm 10-dimensional random feature is derived from 10 randomly selected items (columns in a Gram matrix). However in our algorithm **RNF**

10-dimensional random feature is derived with additional approximation and not just Nyström method from 50 selected items.

We show mean predictive error and its standard deviation computed on the testing sets and report results in a table (2). Across all data sets **RNF** outperforms **NF**. Therefore, we note that better results are achieved when using the combination of a Nyström method with a randomized SVD compared to the Nyström method alone while at the same time the time complexity remains linear in the size of the data set.

Table 2: Performance comparison of **RNF** and **NF** algorithms on real world data sets. We show mean predictive error and its standard deviation. Number of random features is 10, and the number of selected columns is 50.

Data Set	**NF**	**RNF**	**NF runtime (s)**	**RNF runtime (s)**
calHousing	0.672 ± 0.052	0.639 ± 0.032	0.048	0.048
abalone	0.640 ± 0.001	0.639 ± 0.000	0.006	0.007
bank32	0.510 ± 0.027	0.492 ± 0.005	0.014	0.014
bank8	0.059 ± 0.003	0.060 ± 0.003	0.009	0.009
sarcos2	0.307 ± 0.029	0.301 ± 0.020	0.047	0.047
sarcos3	0.248 ± 0.024	0.241 ± 0.025	0.047	0.047
sarcos4	0.083 ± 0.018	0.075 ± 0.008	0.047	0.047
sarcos5	0.498 ± 0.066	0.454 ± 0.037	0.047	0.047
sarcos6	0.694 ± 0.072	0.643 ± 0.038	0.047	0.047
sarcos7	0.101 ± 0.019	0.085 ± 0.007	0.047	0.047
red wine quality	0.913 ± 0.048	0.936 ± 0.057	0.005	0.004
white wine quality	0.931 ± 0.032	0.929 ± 0.014	0.007	0.007
Combined Cycle Power Plant Data Set	0.111 ± 0.042	0.086 ± 0.016	0.008	0.008
Facebook Comment Volume Dataset	1.131 ± 0.001	1.131 ± 0.001	0.082	0.081
Gas Turbine CO and NOx Emission	0.550 ± 0.178	0.393 ± 0.066	0.008	0.008
SGEMM GPU kernel performance	0.982 ± 0.091	0.946 ± 0.042	0.233	0.235
Superconductivty Data	1.630 ± 0.018	1.602 ± 0.018	0.059	0.060
census8H	1.596 ± 0.047	1.495 ± 0.031	0.025	0.024
census16H	1.728 ± 0.032	1.664 ± 0.025	0.031	0.030
kin	0.996 ± 0.003	0.997 ± 0.002	0.015	0.015
pumadynH	0.864 ± 0.017	0.841 ± 0.004	0.017	0.016
pumadynM	0.832 ± 0.016	0.807 ± 0.005	0.015	0.015

5. Conclusion

In this paper we have considered the problem of speeding up kernel regression using random Nyström features. Kernel methods in general have a cubic time complexity, but when used in a combination with the random features that complexity is linear. In the paper [1], it is proven that the in sample error of a regression that uses Nyström features is arbitrarily close to the in sample error of the kernel regression for a large enough number of sampled columns. Here, we analyzed the error of the regression that uses RNyström features (a combination of the Nyström method and a randomized svd). The input set $\{(x_i \in \mathbf{R}^d)_{i=1}^n\}$ is map into the set of m-dimensional random features $\{(r_i \in \mathbf{R}^m)_{i=1}^n\}$ derived from the set of p data vectors $\{\hat{x}_i\}_{i=\overline{1,p}}$ randomly sampled from the input set. RNyström feature is computed as $r_i = \hat{D}^{-1/2}\hat{V}^T k(x_{1:n}, \hat{x}_{1:p})^T$, where columns of $\hat{V} \in \mathbb{R}^{p\times m}$ are approximate eigenvectors and diagonal elements of matrix $\hat{D} \in \mathbb{R}^{m\times m}$ are approximate eigenvalues of the matrix $k(\hat{x}_{1:p}, \hat{x}_{1:p})$, computed from a randomized eigenvalue decomposition. Therefore, we added another approximation to the Nyström method, and we showed that the main property of Nyström features is kept, i. e. that for each error ϵ there is a large enough p so that the error achieved by the regressor learned from r_i is smaller than the ϵ. Additionally, we showed empirically that using a RNyström method is better for the construction of m-dimensional feature vectors than a Nyström method, by comparing their performance on the real world datasets.

References

[1] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory*, pages 185–209, 2013.

[2] Francis R Bach and Michael I Jordan. Predictive low-rank decomposition for kernel methods. In *Proceedings of the 22nd international conference on Machine learning*, pages 33–40. ACM, 2005.
[3] Alex Gittens. The spectral norm error of the naive nystrom extension. *arXiv preprint arXiv:1110.5305*, 2011.
[4] David Gross and Vincent Nesme. Note on sampling without replacing from a finite collection of matrices. *arXiv preprint arXiv:1001.2738*, 2010.
[5] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.
[6] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the nyström method. *Journal of Machine Learning Research*, 13(Apr):981–1006, 2012.
[7] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the international conference on machine learning*, volume 85, 2013.
[8] Mu Li, Wei Bi, James T Kwok, and Bao-Liang Lu. Large-scale nyström kernel matrix approximation using randomized svd. *IEEE transactions on neural networks and learning systems*, 26(1):152–164, 2015.
[9] Lester Mackey, Michael I Jordan, Richard Y Chen, Brendan Farrell, Joel A Tropp, et al. Matrix concentration inequalities via the method of exchangeable pairs. *The Annals of Probability*, 42(3):906–945, 2014.
[10] Michael W Mahoney and Petros Drineas. Cur matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.
[11] Brian McWilliams, David Balduzzi, and Joachim M Buhmann. Correlated random features for fast semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 440–448, 2013.
[12] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.
[13] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pages 1313–1320, 2009.
[14] Bernhard Schölkopf, Alexander J Smola, et al. Learning with kernels: Support vector machines, regularization. *Optimization, and Beyond. MIT press*, 1(2), 2002.
[15] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Support vector machine applications in computational biology*. MIT press, 2004.
[16] John Shawe-Taylor, Nello Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.
[17] Joel A Tropp. Improved analysis of the subsampled randomized hadamard transform. *Advances in Adaptive Data Analysis*, 3(01n02):115–126, 2011.
[18] Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012.
[19] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.
[20] Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.
[21] Tianbao Yang, Yu-Feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In *Advances in neural information processing systems*, pages 476–484, 2012.
[22] Jianguo Zhang, Marcin Marszałek, Svetlana Lazebnik, and Cordelia Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International journal of computer vision*, 73(2):213–238, 2007.