Improving Random Projections With Extra Vectors to Approximate Inner Products

YULONG LI, ZHIHAO KUANG, JIANG YAN LI, AND KEEGAN KANG, (Member, IEEE)

Science, Maths, and Technology (SMT) Cluster, Singapore University of Technology and Design, Singapore 487372

Corresponding author: Keegan Kang (keegan_kang@sutd.edu.sg)

This work was supported in part by the Ministry of Education, Singapore, through an Academic Research Fund Tier 2, under Grant MOE2018-T2-2-013, and in part by the MOE SUTD Faculty Early Career Award under Grant RGFEC17003.

ABSTRACT This research concerns itself with increasing the accuracy of random projections used to quickly approximate the inner products of data vectors from a given dataset by adding additional information, namely, adding and storing more extra known vectors to the given dataset and associated information. We show how the variance of estimated inner products is reduced as more vectors are added, how variance reduction is related to the geometry of the dataset and moreover, the asymptotic behaviour of the variance as the number of extra vectors added goes to infinity. We provide the formulae governing the estimate of inner products for adding arbitrarily many extra vectors. Lastly, we demonstrate how to efficiently implement the computations of the estimates by showing we can use pre-computed and stored values for most of the computations. Numerical simulations are conducted to support the analytical results.

INDEX TERMS Computational statistics, information retrieval, maximum likelihood estimation, random projections, variance reduction.

I. INTRODUCTION

As a popular and useful dimension reduction technique in high dimensional learning in statistics, random projections have appeared in machine learning [1], data mining [2], information retrieval [3], classification [4], clustering [5], hypothesis testing [6], regression [7], and similarity search [8], etc. Many papers have been devoted to the discussion of various aspects of random projections, to name a few, [9]–[17].

Particularly in machine learning, distances between pairs of data points are compared against each other, which could be inner products, $L^p$ norms, Euclidean distances, etc. Today, machine learning is used for a variety of complex applications (e.g. image processing). As such, pairwise distances for large numbers of high dimensional data points are being compared against each other. The direct computation of such distances would take too much time, so there must be methods to quickly estimate the distances between such data points with minimal compromises in accuracy. Such estimations can be done with random projections and this research is concerned with the improvement of the accuracy of random projections.

Suppose that we have $q$ observations with $p$ features and stored in matrix $M \in \mathbb{R}^{q \times p}$, and that we are interested in calculating the inner products of all pairs of row vectors in $M$, i.e. $MM^T$. With $q$ observations, there are $\frac{q(q-1)}{2}$ pairs. Let each $i^{th}$ row in $M$ be denoted by $\tilde{x}_i$. Since we have $p$ features, computing the inner product $\sum_{t=1}^p x_{i,t}x_{j,t}$ between any two pairs $\tilde{x}_i, \tilde{x}_j$ would take $O(p)$ time for each pair, hence the total time taken would be $O(q^2p)$. When $q$ and $p$ are large, the direct computation of $MM^T$ would be computationally costly. Random projections compute $V = \frac{1}{\sqrt{k}}RM^T$ where $R \in \mathbb{R}^{k \times p}$ is a random matrix with each component $r_{i,j}$ i.i.d. from $\mathcal{N}(0, 1)$. After this, the estimation of $MM^T$ is given by

$$ MM^T \approx V^TV \quad (1.1) $$

To understand why this approximation holds, consider the form of the $(i, j)^{th}$ entry of $V$, given by

$$ v_{i,j} = \frac{1}{\sqrt{k}} \sum_{t=1}^p r_{i,t}x_{j,t} \quad (1.2) $$
Suppose we wanted to approximate the inner product of $\tilde{x}_1^T \tilde{x}_2$ in our original matrix $M$. If we considered the expectation of
\[
E[v_{s,1}v_{s,2}] \quad (I.3)
\]
for some fixed $s$, $1 \leq s \leq k$, we have that
\[
E[v_{s,1}v_{s,2}] = E \left[ \frac{1}{k} \sum_{t=1}^{p} r_{s,t} x_{1,t} \left( \sum_{t=1}^{p} r_{s,t} x_{2,t} \right) \right] \quad (I.4)
\]
\[
= \frac{1}{k} \left( E \left[ \sum_{t=1}^{p} r_{s,t}^2 \sum_{t=1}^{p} r_{s,t} x_{1,t} x_{2,t} \right] \right) \quad (I.5)
\]
\[
= \frac{1}{k} \sum_{t=1}^{p} x_{1,t} x_{2,t} \quad (I.6)
\]
since $r_{i,j}$ is i.i.d. $\mathcal{N}(0, 1)$, and hence
\[
E \left[ \sum_{s=1}^{k} v_{s,1}v_{s,2} \right] = \sum_{s=1}^{k} x_{1,s} x_{2,s} \quad (I.7)
\]
Hence each $(i,j)$ entry of $V^T V$ in (1.1) is an estimate of the inner product between $\tilde{x}_1$ and $\tilde{x}_2$.

When $k$ is much smaller than $q, p$, i.e. $k \ll (q, p)$, the saving in computational complexity from $O(q^2)$ of calculation for $MM^T$ to $O(qpk + q^2k)$ of calculation for $\frac{1}{\sqrt{k}} MR$ plus $V^T V$ is substantial. A second way to think about the computational cost is this: Computing $V$ takes $O(qpk)$ time which is a one-off cost, and can be stored in $O(qk)$ space. Subsequent estimates of any pairs of inner products would take $O(k)$ time, rather than $O(p)$ time, which is a drastic speedup when $k \ll p$, and the values of $V^T V$ could be used in applications such as ANN (approximate nearest neighbors).

**A. RELATED LITERATURE**

From the above example, we can see that random projections focus on projecting the data to a lower dimensional space, and then quickly estimating the distance between vectors. There has been research which focused on improving the speed of random projections. For example, [18] and [19] looked at changing the distribution of the random projection matrix. Achlioptas [18] considered using i.i.d. entries from the Rademacher distribution for the random projection matrix, while Li et al. [19] considered using i.i.d. entries of the the Sparse Bernoulli distribution for the random projection matrix. Since the support of the Rademacher distribution consisted of $\{-1, 1\}$, and the support of the Sparse Bernoulli distribution consisted of $\{-1, 0, 1\}$, matrix multiplication with the random projection matrix became much faster compared to ordinary random projections. References [20] and [21] also looked at the cases where the entries of the random projection matrix were dependent on each other and could be generated recursively. All of these methods ended up speeding random projections.

On the other hand, there is also research in reducing the variance of the random projection estimate via various computational statistical techniques. Techniques such as anti-thetic sampling [22], Bayesian priors [23], control variates [16], [24], maximum likelihood estimation [10], [17] have all been used to improve the estimate of random projections.

Our work mainly focuses on improving the estimate of random projections, and builds upon the ideas in [10], [17].

**B. IMPROVING RANDOM PROJECTIONS BY USING MARGIN NORMS**

By adding additional information, namely, pre-calculating and storing partial information beforehand, one can improve the accuracy of random projections to obtain a better approximation $V^T V$ for $MM^T$; and some work (e.g. [10], [17]) have specifically addressed this topic. For example, by using a maximum likelihood estimator (MLE) and taking advantage of marginal norms, Li et al. in [10] have shown that margin-constrained random projections can improve estimation accuracy remarkably.

More specifically, consider the data matrix $M \in \mathbb{R}^{q \times p}$ that is given. Pre-calculate and store the norm of each row vector in $M$, from which margin norms come at a cost of $O(qp)$ complexity. To approximate $MM^T$, let $\tilde{x}_1^T = (x_1, \ldots, x_p)$, $\tilde{x}_2^T = (x_2, \ldots, x_p) \in \mathbb{R}^p$ represent two arbitrary different row vectors of $M$. Instead of directly computing $\tilde{x}_1^T \tilde{x}_2$ for all possible pairs of $\tilde{x}_1, \tilde{x}_2$ in $M^T$, one can adopt a random matrix $R \in \mathbb{R}^{k \times p}$ with each entry $r_{i,j}$ i.i.d. $\mathcal{N}(0, 1)$, to generate new random vectors $\tilde{v}_1 = (v_1, \ldots, v_{k,1})^T$ and $\tilde{v}_2 = (v_2, \ldots, v_{2,k})^T$ by the map $T \equiv \frac{1}{\sqrt{k}} R$, with $\tilde{v}_1 = T \tilde{x}_1$ and $\tilde{v}_2 = T \tilde{x}_2$ respectively. Denote the vector $(\tilde{u}_j^{(2)}) = [v_{1,j}, v_{2,j}]^T$ which comprises the $j$th element of $\tilde{v}_1$ and $\tilde{v}_2, j = 1, \ldots, k$.

Then it was shown in [10] that
\[
(\tilde{u}_j^{(2)}) \sim \mathcal{N} \left( \bar{0}, \Sigma \right), \quad j = 1, \ldots, k \quad (I.8)
\]
with
\[
\sigma_{i,j} = \tilde{x}_i^T \tilde{x}_j / k, \quad 1 \leq i, j \leq 2 \quad (I.9)
\]
where $a := \tilde{x}_1^T \tilde{x}_2$ is to be estimated; and the associated joint likelihood function for tuples $\{(v_{1,j}, v_{2,j})\}_{j=1}^{k}$ is
\[
\text{lik} \left( \{v_{1,j}, v_{2,j}\}_{j=1}^{k} \right) \propto |\Sigma|^{-\frac{k}{2}} \exp \left(-\frac{1}{2} \sum_{j=1}^{k} (\tilde{u}_j^{(2)})^T \Sigma^{-1} (\tilde{u}_j^{(2)}) \right) \quad (I.10)
\]
\[1\] By pre-computing and storing norms of row vectors of $M$, the computation of $MM^T$ is therefore the same as the computation of $\tilde{x}_1^T \tilde{x}_2$, with $\tilde{x}_1 \neq \tilde{x}_2$ and letting $\tilde{x}_1^T$ and $\tilde{x}_2^T$ go over all different pairs of row vectors in $M$. This way, we only need to focus on approximating each $\tilde{x}_1^T \tilde{x}_2$, which is convenient to unfold the theory.
By writing down the log likelihood function, \( l(a) \), from (I.10), the following estimation result is obtained:

**Proposition 1 ([10], Lemma 2):** Suppose the margins, \( m_1 = x_1^T \bar{x}_1 \) and \( m_2 = x_2^T \bar{x}_2 \) are known; the maximum likelihood estimator (MLE), denoted as \( \hat{a} \), is a solution to:

\[
a^3 - \bar{v}_1^T \bar{v}_2 a^2 + (-m_1 m_2 + m_1 \| \bar{v}_2 \|^2 + m_2 \| \bar{v}_1 \|^2) \times a - m_1 m_2 \bar{v}_1^T \bar{v}_2 = 0 \tag{I.11}
\]

**Furthermore, the (asymptotic) variance of \( \hat{a} \) is**

\[
\text{Var}(\hat{a}) = \frac{\left( m_1 m_2 - a^2 \right)^2}{m_1 m_2 + a^2} \tag{I.12}
\]

We can see that the approximation of \( a \) is just one of the roots of the cubic equation, which can be carried out without difficulty by a root-finding algorithm such as Newton Raphson once the coefficients are determined; and it was also pointed out that this (asymptotic) variance of MLE, namely, \( \text{Var}(\hat{a}) \) in Proposition 1 is smaller than the ones obtained by other methods (see [10]), and therefore is an ideal candidate to use in applications.

**C. FROM ADDING MARGINAL NORM TO ADDING MORE ADDITIONAL INFORMATION (VECTORS)**

The results in Proposition 1 suggest that it is seemingly true that adding more known information will yield a further variance reduction and therefore obtaining a better estimate for each inner product \( \bar{x}_1^T \bar{x}_2 \). So, it is natural to ask: "can we continue to reduce the variance \( \text{Var}(\hat{a}) \) with more additional information?"

In this work, based on results in 1, we endeavor to the investigation of improvement of random projection by adding more additional information (extra known vectors) in the context of maximum likelihood estimation.

More precisely, for data matrix \( M \in \mathbb{R}^{q \times p} \), we not only pre-compute and store norms of each row in \( M \), but also pick \( n \) extra known vectors denoted as \( \bar{x}_3, \cdots, \bar{x}_{n+2} \in \mathbb{R}^p \) and further calculate and store their norms and the inner products between each of these \( n \) vectors and each row vector of \( M \) as well. The goal is still to approximate \( \bar{x}_1^T \bar{x}_2 \), where \( \bar{x}_1^T, \bar{x}_2^T \) are two arbitrary different vectors of dataset \( M \). However, with these more known information, we expect to obtain a much smaller variance \( \text{Var}(\hat{a}) \) than the one in Proposition 1.

To unfold the work, let us do setting up and go to a little details in the following.

**D. OUR RESULTS**

Towards the goal of estimating \( \hat{a} = \bar{x}_1^T \bar{x}_2 \), let us denote these \( n \) extra vectors as

\[
\bar{x}_3 = \begin{bmatrix} x_{3,1} \\ \vdots \\ x_{3,n} \end{bmatrix}, \cdots, \bar{x}_{n+2} = \begin{bmatrix} x_{n+2,1} \\ \vdots \\ x_{n+2,n} \end{bmatrix}, \tag{I.13}
\]

where \( n \) is a non-negative integer with the convention that we add nothing if \( n = 0 \) (leading back to the case in Proposition 1). Under random projection mappings, accordingly we have additional vectors \( \bar{v}_3 = T \bar{x}_3, \ldots, \bar{v}_{n+2} = T \bar{x}_{n+2} \).

We similarly let the vector \( \bar{u}^{(n+2)}_j = [v_{1,j}, \ldots, v_{n+2,j}]^T \) and have

\[
[\bar{u}^{(n+2)}_j] \sim \mathcal{N}(0, \Sigma), \quad j = 1, \ldots, k \tag{I.14}
\]

with

\[
\sigma_{i,j} = \bar{x}_i^T \bar{x}_j / k, \quad 1 \leq i, j \leq (n + 2), \tag{I.15}
\]

and the according joint likelihood function for tuples \( \{v_{1,j}, \ldots, v_{n+2,j}\}_{j=1}^k \) is

\[
\text{lik}(\{v_{1,j}, \ldots, v_{n+2,j}\}_{j=1}^k) \propto |\Sigma|^{-\frac{k}{2}} \exp\left\{-\frac{1}{2} \sum_{j=1}^k (\bar{u}^{(n+2)}_j)^T \Sigma^{-1} (\bar{u}^{(n+2)}_j)\right\} \tag{I.16}
\]

Under this setting, our main results are Theorems 6 and 9, which refine the two results in Lemma 2 and establish the formulae governing MLE \( \hat{a} \) and variance \( \text{Var}(\hat{a}) \) separately, and therefore generalizing Proposition 1.

We shall discuss analysis questions include how the variance reduction works as we add \( n \) extra vectors, whether there exists an optimal \( n \) that minimize the variance \( \text{Var}(\hat{a}) \), and what is the asymptotic behaviour of \( \text{Var}(\hat{a}) \) as \( n \) goes to infinity. As a byproduct, we also point out how the variance reduction is related to the geometry of dataset \( M \). Meanwhile, in the numerical part, computational cost of \( \hat{a} \) for adding extra vectors is discussed; we provide simulations for various comparisons to support the analytical results and propose effective algorithms for evaluating the MLE \( \hat{a} \).

**E. NOTATION**

Let us first summarize the notation and convention that will be adopted throughout the material.

- Notation \( \bar{\cdot} \) is a column vector by default. And the subscript of a matrix denoting the size will be often omitted when there is no chance of misunderstanding.
- Matrix \( M \in \mathbb{R}^{q \times p} \) represents a dataset, \( q, p \) represent observations and features of data points.
- \( \bar{x}_1, \bar{x}_2 \) will be called testing vectors and denote an arbitrary pair of different column vectors in our dataset \( M^T \). Therefore, instead of directly estimating \( MM^T \), we can focus on estimating \( \bar{x}_1^T \bar{x}_2 \), letting \( \bar{x}_1, \bar{x}_2 \) vary over all possible pairs.
- \( \bar{x}_3, \ldots, \bar{x}_{n+2} \in \mathbb{R}^p \) are extra vectors added.
- Let \( A \equiv k \Sigma \), where \( \Sigma \) is the covariance matrix in (I.15).

\[
(i,j) \text{ entry of } A = \bar{x}_i^T \bar{x}_j, \quad 1 \leq i, j \leq (n + 2) \tag{I.17}
\]

and in particular, \( a := \bar{x}_1^T \bar{x}_j \) is what we want to approximate in this work.

- \( R \in \mathbb{R}^{k \times p} \) represents the random matrix and its \( (i,j) \)-th entry is denoted by \( r_{i,j} \), where \( r_{i,j} \) are i.i.d. \( \mathcal{N}(0, 1) \).
- \( \bar{v}_3 = T \bar{x}_3, \ldots, \bar{v}_{n+2} = T \bar{x}_{n+2} \), where \( T \equiv \frac{1}{\sqrt{k}} R \). The \( j \)th entry of \( \bar{v}_i \) is denoted by \( v_{i,j} \), for \( i = 3, \ldots, 2 + n \).
- \( \{\bar{u}^{(a)}_{i,j}\} \) will be shorthand for the vector \( [v_{1,j}, \ldots, v_{n,j}]^T \).
• det (·) and |·| will be used interchangeably to denote the determinant of a matrix.

• \( \alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3 \) are terms that appear in the final cubic equation given by Theorem 9, and are described fully in the theorem itself.

II. MAXIMUM LIKELIHOOD ESTIMATION

Now we are in the position to calculate the maximum likelihood estimator \( \hat{a} \) by utilizing the joint likelihood function (I.16) for adding \( n \) extra vectors. We expect to obtain a generalization of Proposition 1, where only marginal norms are added, namely \( n = 0 \). As \( n \) increases, the pattern for finding \( \hat{a} \) and \( \text{Var} (\hat{a}) \) becomes obscure. The first challenge is to find the uniform equations involving \( \hat{a} \) and \( \text{Var} (\hat{a}) \). We first state the result.

**Lemma 2:** Assume \( \det (\Sigma) \neq 0 \).

1. The maximum likelihood estimator \( \hat{a} \) (corresponding to testing vectors \( \vec{x}_1, \vec{x}_2 \)) is a solution to equation

\[
|A'| |A| + B' |A| - B |A'| = 0, \tag{II.18}
\]

where

\[
B = \sum_{i=1}^{n} \sum_{j=1}^{n+2} A_{i,j} \vec{v}_i^T \vec{v}_j \tag{II.19}
\]

and \( A_{i,j} \) is the \((i, j)\)-th entry of \( \text{adj}(A) \).

2. The asymptotic variance of \( \hat{a} \) is given by

\[
\text{Var} (\hat{a}) = \frac{2}{k} \left( |A|'^2 - (4A_{1,2})^2 |A| + (4A_{1,2} - |A'|) |A'| \right), \tag{II.20}
\]

The usefulness of this lemma lies in that it confirms the existence of equations governing \( \hat{a} \) and variance \( \text{Var} (\hat{a}) \) for adding any \( n \) extra vectors and that it provides general formulæ for finding \( \hat{a} \) and \( \text{Var} (\hat{a}) \).

In view of Proposition 1, it is natural to expect that equation (II.18) is supposed to be a cubic equation in terms of \( a \) for arbitrary \( n \) and \( \text{Var} (\hat{a}) \) decreases as \( n \) increases. Unfortunately, this is not straightforward at this stage from Lemma 2. As \( n \) increases, on one hand, the order of polynomial in the left-hand side of (II.18) with respect to \( a \) is not immediately clear and thus solving for the roots of equation (II.18) might be intimidating; on the other hand, the right-hand side of equation (II.20) does not involve \( n \) explicitly, which blurs the relationship between \( \text{Var} (\hat{a}) \) and the number of extra vectors \( n \). We need to seek alternate equivalent expressions that are more convenient for analysis.

A. FURTHER ANALYSIS OF \( \hat{a} \) AND \( \text{VAR}(\hat{a}) \)

We intend to refine equations (II.18) and (II.20) separately. For analytical reason, let us first focus on analysis of (II.20), and turn to (II.18) after.

B. THE BEHAVIOR OF \( \text{VAR}(\hat{a}) \)

Observe that \( \text{Var} (\hat{a}) \) in equation (II.20) does not explicitly depend on \( n \), but does depend on \( |A| \). To build the connection between \( \text{Var} (\hat{a}) \) and \( n \), we first investigate the relations between \( \text{Var} (\hat{a}) \) and \( |A| \), and then the relations between \( |A| \) and \( n \). By combining this two pieces together, the relationship between \( \text{Var} (\hat{a}) \) and \( n \) will be clear. The main result will be established after several lemmas that are crucial.

The first lemma produces a nice expression of \( |A| \).

**Lemma 3:** Partition the matrix \( A \) in (I.17) as

\[
A = \begin{bmatrix}
\vec{x}_1^T \vec{x}_1 & a & \vec{x}_1^T \vec{x}_2 & B_1^{n \times 2} \\
a & \vec{x}_2^T \vec{x}_2 & B_2^{n \times 2} \\
& & & D_{n \times n}
\end{bmatrix} \tag{II.21}
\]

then the determinant of \( A \) can be expressed, with respect to \( a \), as

\[
|A| = |D| a^2 + 2a A_{1,2} + A_0, \tag{II.22}
\]

where \( A_{1,2} \) is the \((1, 2)\)-th entry of \( \text{adj}(A) \), namely, \((-1)^{1+2} |W| \) and \( W \) is the \((n + 1) \times (n + 1) \) matrix generated by eliminating the 1st row and the 2nd column of \( A \), and \( A_0 \) denotes the value of \( |A| \) evaluated at \( a = 0 \).

By utilizing this lemma, we now can rewrite \( \text{Var} (\hat{a}) \) in Lemma (2) into concise expressions in terms of \( |A| \).

**Lemma 4:** The variance \( \text{Var} (\hat{a}) \) satisfies

\[
\frac{1}{\text{Var} (\hat{a})} = \frac{k}{2} \left( \frac{2|D|}{|A|} + (\log |A|)^2 \right), \tag{II.23}
\]

and

\[
\frac{1}{\text{Var} (\hat{a})} = \frac{k}{2} \left( 1 - \frac{A_0}{|A|} \right)^2 + \frac{1}{a^2} \left( \frac{|D|}{|A|} \right)^2 a^2 + \frac{2|D| A_0}{|A|^2}, \tag{II.24}
\]

where \( D \) and \( A_0 \) are defined as in Lemma 3.

\( \frac{2|D|}{|A|} \) appears in equation (II.23) and \( \left( \frac{|D|}{|A|} \right)^2 a^2 \) appears in equation (II.24), and all other terms are non-negative. Therefore, both equations (II.23) and (II.24) imply that \( \text{Var} (\hat{a}) \) decreases as \( |A| \) decreases (with a fixed \( |D| \), which is determined by our \( n \) extra vectors). However, the subtle difference between these two equations is: if \( |A| \) is decreasing and very close to \( |D| \), then we are supposed to look at equation (II.23) instead of equation (II.24) as \( \text{Var} (\hat{a}) \) decreases in order \( \frac{2|D|}{|A|} \) that is quicker than order \( \left( \frac{|D|}{|A|} \right)^2 a^2 \); otherwise, if \( |A| \) is decreasing and very close to \( 0 \), then we are supposed to look at equation (II.24) instead of equation (II.23) as \( \text{Var} (\hat{a}) \) decreases in order \( \left( \frac{|D|}{|A|} \right)^2 a^2 \) that is quicker than order \( \frac{2|D|}{|A|} \).

Indeed, other alternate equivalent expressions of \( \text{Var} (\hat{a}) \) can also be derived, which provide insight for the behavior of variance from different angles and serve as convenient tools for various purposes.

Now from above, we already see that \( \text{Var} (\hat{a}) \) closely depends on the behavior of \( |A| \), we therefore turn to investigation of the relationship between \( |A| \) and the number of additional vectors \( n \). Namely, as \( n \) increases, how the determinant \( |A| \) behaves, which will eventually help connect \( \text{Var} (\hat{a}) \) and \( n \).

**Lemma 5:** Let \( m \) be a positive integer and inductively define the sequence of symmetrical and positive definite
matrices, denoted as \( \{T_{m+1}\} \), by

\[
T_{m+1} = \begin{bmatrix}
1_{1,1} & \cdots & 1_{1,m+1} \\
\vdots & \ddots & \vdots \\
1_{m+1,m} & \cdots & 1_{m+1,m+1}
\end{bmatrix},
\]

(II.25)

(they have overlapping entries). It is valid that

\[
|T_{m+1}| \leq c|T_m|,
\]

where

\[
c = t_{m+1,m+1} - \max\{t_{1,1}^2, \ldots, t_{m,m}^2\}.
\]

Based on this lemma, we arrive at the following:

**Theorem 6:** Suppose we add \( n \) extra known vectors \( \tilde{x}_3, \ldots, \tilde{x}_{n+2} \) that are orthogonal to each other and \( \det(A) \neq 0 \).

Then

- If \( \tilde{x}_3, \ldots, \tilde{x}_{n+2} \) are all perpendicular to both testing vectors \( \tilde{x}_1 \) and \( \tilde{x}_2 \), then \( \text{Var}(\hat{\alpha}) \) always remain the same no matter what \( n \) is. Namely,

\[
\text{Var}(\hat{\alpha}, n) = \text{Var}(\hat{\alpha}) .
\]

- Suppose each of \( \tilde{x}_3, \ldots, \tilde{x}_{n+2} \) is not perpendicular to at least one of the testing vectors \( \tilde{x}_1, \tilde{x}_2 \) and denote

\[
s_i = \max\{-\frac{1}{\|\tilde{x}_j\|^2\|\tilde{x}_i\|^2}, j = 1, 2, \ldots, n\},
\]

Then

\[
\text{Var}(\hat{\alpha}, n) \leq \frac{1}{k} \left( \frac{2}{(\|\tilde{x}_1\|^2\|\tilde{x}_2\|^2 - \alpha^2)^2} \prod_{i=3}^n (1 - s_i) + (\log|A|^2)^2 \right).
\]

(II.28)

(II.29)

Notice one of the constrains imposed on extra vectors added in Theorem 6 is orthogonality, which is always possible by the Gram-Schmidt process.

The first part points out that we can reduce variance \( \text{Var}(\hat{\alpha}) \) corresponding to the testing pair \( \tilde{x}_1, \tilde{x}_2 \) only when additional vectors added are not all orthogonal to \( \tilde{x}_1 \) and \( \tilde{x}_2 \), which is a good reason for the assumption in the second part.

Observe from equation II.28 and II.29 that \( s_i \) essentially measures the angles between extra vectors \( \tilde{x}_3, \ldots, \tilde{x}_{n+2} \) and testing vectors \( \tilde{x}_1, \tilde{x}_2 \) and that \( \text{Var}(\hat{\alpha}, n) \) goes to zero if any \( s_i \) is close to 1.

Hence, the second part tells us that there does not actually exists an “optimal” \( n \) that minimizes the variance \( \text{Var}(\hat{\alpha}) \); to lower the variance, we should on one hand add more additional vectors, and on the other hand, we should try to make extra vectors have small angles with all data points (all possible testing vectors \( \tilde{x}_1, \tilde{x}_2 \)). Since the later requires one to know the geometry information of dataset \( M \) in advance, keeping adding additional vectors seems to be the simplest way to reduce the variance. However, as we will see soon, the computational cost will truly be the concern that prevents one from choosing \( n \) too large.

### C. FURTHER ANALYSIS OF \( \hat{\alpha} \)

In this section, we show that the highest order of \( a \) in equation (II.18) is 3 for any \( n \). The following definition provides a convenient tool to describe the coefficients of the cubic equation.

**Definition 7:** Let \( m \) be a positive integer, \( M \) be a matrix of size \( m \times m \) and \( \{\tilde{l}_i\}_{i = 1}^m \) be a sequence of \( m \) vectors with \( m \) entries for each vector, we define the function \( \mathcal{N} \) from \( M \) and \( \{\tilde{l}_i\} \) to \( \mathbb{R} \) as:

\[
\mathcal{N}(M, \{\tilde{l}_i\}) := \det (M_1) + \cdots + \det (M_m),
\]

where the matrix \( M_i(i = 1, \ldots, m) \) is generated by replacing the \( i \)-th row of \( M \) with \( \tilde{l}_i^T \).

**Lemma 8:** Let \( D \) be the \( n \times n \) matrix defined in (II.21),

\[
\tilde{w}_1 = [\tilde{x}_1^T \tilde{x}_3, \ldots, \tilde{x}_1^T \tilde{x}_{n+2}]^T \quad \text{and} \quad \tilde{w}_2 = [\tilde{x}_2^T \tilde{x}_3, \ldots, \tilde{x}_2^T \tilde{x}_{n+2}]^T.
\]

Denote the matrices \( A, A, \hat{A}, \hat{A} \) of size \((n + 2) \times (n + 2)\) as

\[
\begin{align*}
1 & A = \begin{bmatrix} 0 & 0 & 0 & 0 & \tilde{v}_1^T & \cdots & \tilde{v}_{n+2}^T \end{bmatrix} \\
2 & A = \begin{bmatrix} 0 & 0 & 0 & 0 & \tilde{v}_1^T & \cdots & \tilde{v}_{n+2}^T \end{bmatrix} \\
3 & A = \begin{bmatrix} 0 & 0 & 0 & 0 & \tilde{v}_1^T & \cdots & \tilde{v}_{n+2}^T \end{bmatrix} \\
4 & A = \begin{bmatrix} 0 & 0 & 0 & 0 & \tilde{v}_1^T & \cdots & \tilde{v}_{n+2}^T \end{bmatrix}.
\end{align*}
\]

(II.30)

Further denote sequences of vectors as

\[
\{\tilde{l}_1\} = (0, 0, \tilde{v}_1^T \tilde{v}_3, \ldots, \tilde{v}_1^T \tilde{v}_{n+2})^T
\]

\[
\{\tilde{l}_1\} = (0, 0, \tilde{v}_1^T \tilde{v}_3, \ldots, \tilde{v}_1^T \tilde{v}_{n+2})^T
\]

\[
\{\tilde{l}_1\} = (0, 0, \tilde{v}_1^T \tilde{v}_3, \ldots, \tilde{v}_1^T \tilde{v}_{n+2})^T
\]

\[
\{\tilde{l}_1\} = (0, 0, \tilde{v}_1^T \tilde{v}_3, \ldots, \tilde{v}_1^T \tilde{v}_{n+2})^T
\]

\[
\{\tilde{l}_1\} = (0, 0, \tilde{v}_1^T \tilde{v}_3, \ldots, \tilde{v}_1^T \tilde{v}_{n+2})^T
\]

\[
\{\tilde{l}_1\} = (0, 0, \tilde{v}_1^T \tilde{v}_3, \ldots, \tilde{v}_1^T \tilde{v}_{n+2})^T
\]

\[
\{\tilde{l}_1\} = (0, 0, \tilde{v}_1^T \tilde{v}_3, \ldots, \tilde{v}_1^T \tilde{v}_{n+2})^T
\]

(i = 1, \ldots, n + 2). Then, for matrix \( A \) and number \( B \) in equation (II.18) of Lemma 2, we have

\[
\det (A) = \alpha_1 a^2 + \alpha_2 a + \alpha_3, \quad B = \beta_1 a^2 + \beta_2 a + \beta_3,
\]

where

\[
\alpha_1 = \frac{1}{|A|}, \quad \alpha_2 = 2\frac{1}{|A|}, \quad \alpha_3 = \frac{3}{|A|},
\]

|\[
\beta_1 = \mathcal{N}(A, \{\tilde{l}_1\}) \quad \beta_2 = \mathcal{N}(A, \{\tilde{l}_1\}) \quad \beta_3 = \mathcal{N}(A, \{\tilde{l}_1\}),
\]

(II.37)

The first part of Lemma 2 can be restated as follows.

**Theorem 9:** \( \hat{\alpha} \) is a solution to the cubic equation

\[
2\alpha_1 a^3 + (3\alpha_1 a^2 + \alpha_1 a^2 - \alpha_1 a^2) a^2 + (2\alpha_1 a + 2\alpha_1 a + \alpha_1 a^2 - 2\alpha_1 a) a
\]

(II.30)

\[
+ (\alpha_2 a_1 + \alpha_3 a_1 - \alpha_2 a_1) = 0,
\]

(II.36)
where

\[
\alpha_1 = |A|, \alpha_2 = 2|A|, \alpha_3 = |A|,  \\
\beta_1 = \mathcal{N}(A, \{l_i\}), \beta_2 = \mathcal{N}(A, \{\tilde{l}_i\}) + \mathcal{N}(\{l_i\}),  \\
\beta_3 = \mathcal{N}(A, \{3l_i\}).
\]

(II.39)

Theorem 9 confirms that, for each testing vector pair \(\tilde{x}_1, \tilde{x}_2\), equation (II.18) in Lemma 2 is a cubic equation with respect to \(a\), which is quite general, or use the formula of determinant of \(2 \times 2\) block matrix, which is based on the features of matrices \(A, \tilde{A}, A, \tilde{A}\), we find that efficient implementation is more about how many and how to choose the \(n\) extra vectors.

Therefore, as far as Phase 3 is concerned, it is meaningful to know how quickly the computational cost increases as \(n\) grows, i.e., the power in \(O(n^3)\) with respect to \(n\).

Let us consider this in the following by comparing \(n\) arbitrary vectors against \(n\) orthonormal and normalized vectors.

**Procedure 1 Computing \(\alpha_1, \alpha_2, \alpha_3\)**

1. Utilize the formula of the determinant of block square matrix

\[
\text{det}
\begin{bmatrix}
C_1 & C_2 \\
C_3 & C_4
\end{bmatrix} = \text{det}(C_1) \text{det}(C_3 - C_2 D^{-1} C_4)
\]

(III.40)

to calculate \(1, 2, 3, 4\).

(2). Then use pre-computed and stored information in Phase 2 where needed.

The total cost of Procedure 1 takes at least \(O(n^3)\) using Gaussian elimination to compute \(\text{det}(D)\) as well as \(D^{-1}\).

**Procedure 2 Computing \(\beta_1, \beta_2, \beta_3\)**

1. For each \(\mathcal{N}(A, \{l_i\}) (j = 1, 2, 3, 4)\),

for \(i = 1, 2, \cdots, n + 2\) do

   Calculate \((A, \{l_i\})\) by applying Definition 7 and Gaussian elimination.

end for

The total cost for Procedure 2 takes \(O(n^4)\) for each pair \(\tilde{x}_1, \tilde{x}_2\).

Comparing with the cost for \(\alpha_1, \alpha_2, \alpha_3\), we see that the cost of calculation of \(\beta_1, \beta_2, \beta_3\) dominates for all but very small problems. The increase of the cost of Procedure 2 is quadruplicate with respect to \(n\), i.e., \(O(n^4)\), from which the computation can be too large to be prohibitive as \(n\) increases.

On the other hand, we can reduce the computational cost if we choose extra vectors to be orthonormal and normalized. This will make \(D\) in (III.40) be an identity matrix and drastically simplify calculations, which therefore produces concise formulae for \(\alpha\)’s and \(\beta\)’s using pre-computed and stored values in Phase 2. This way, we can not only lower the number of arithmetic operations but also take advantage of the way in which data was stored to decrease the memory access.

We can hence simplify Procedure 1 to take \(O(n)\) time, which is a one-off cost, and simplify Procedure 2 to take \(O(n^2)\) time for each vector pair. The exact expressions for \(\alpha\)’s and \(\beta\)’s are in Appendix H.
Table 1 provides a comparison of computational cost between naive random projections (RP), random projections with control variates [16], RP with marginal norms (namely the work in [10]) and RP with \( n \) extra vectors, and sign full random projections [25] showing how computational complexity increases as additional information is added. As Newton Raphson has quadratic convergence and converges in about 1-3 steps in finding the root of the cubic polynomial, we can treat this as having complexity of \( O(1) \) time. It can be seen that while we have an increased computational cost at the start, this is a one-off cost, and estimating the inner product of any \( i,j \) pair just requires an additional \( O(n^2) \) time, which is negligible when we generate some small \( n \) number of extra vectors.

### B. SIMULATIONS

We compare the numerical experiments of approximating inner products using maximum likelihood estimate method with \( n = 1, 2, 3 \) extra vectors to the methods of naive random projections, Li’s work [10], the method of control variates (see for example [24]), and sign full random projections [25]. We denote RP to be the ordinary random projection estimator of the inner product, Li to be Li’s estimate of the inner product, CV to be the bivariate control variate estimate of the inner product, and sign RP to denote sign full random projections. We denote MLE-\( n \), \( n = 1, 2, 3 \) to be the maximum likelihood estimate algorithm with \( n \) extra vectors.

We perform experiments on two datasets: Arcene and DrivFace, which are retrieved from the UCI machine learning repository [26], [27].

Before we proceed with our main experiments, we first do a sanity check and look at the computational cost of our algorithm. Table 1 shows the computational complexity of our algorithm, but we want to see how our algorithm performs in practice in terms of speed. We take the first two vectors \( \hat{x}_1, \hat{x}_2 \) from the DrivFace dataset, and find the average time taken (over 1000 simulations) to compute the estimate of the inner product \( \hat{x}_1^T \hat{x}_2 \) for varying \( k \) columns of the random projection matrix for \( k = \{10, 20, \ldots, 100000\} \), before smoothing the results. Figure 1 shows our results. We can see that while our method takes more time than the four baseline methods, it still keeps to the same magnitude of time taken with \( k < 10^3 \). Since we usually look at the random projection performance with \( k < 10^2 \), our method is still comparable with the baseline methods, though at a cost of speed. This is the main limitation of our method.

We next proceed with our main experiments, which are to check whether our new algorithms outperform those we proposed in the previous papers. Specifically, we compute the average root mean squared error (RMSE) of all pairwise inner products on the dataset, and see whether the RMSE of maximum likelihood estimate is lower than that of other algorithms. The RMSE is defined as follows:

\[
\text{RMSE} = \sqrt{\frac{\sum_{i,j}(\hat{a}_{i,j} - x_i^T x_j)^2}{C_q^2}}
\]

(III.41)

where \( \hat{a}_{i,j} \) is the estimated inner product computed by our algorithms, and \( x_i^T x_j \) represents the true value of inner product between \( i \)th and \( j \)th row vectors of dataset \( M \in \mathbb{R}^{q \times p} \), and \( C_q^2 \) denotes the total number of pairs \( x_i, x_j \).

We choose the test set of Arcene which has 100 observations and 10000 features. The dataset DrivFace has 606 observations and 6400 features. Our datasets are preprocessed by centering and normalization. For each dataset, we use a random projection matrix \( R \), where each entry is i.i.d. \( \mathcal{N}(0, 1) \), and \( k \) ranges from 1 to 100. We repeat the experiments for 1000 simulations with different random seeds, and then we plot the average performance as well as 3 standard deviations over the 1000 experiments.

In the simulation, we choose the singular vectors of the dataset \( M \in \mathbb{R}^{q \times p} \) to be the extra vectors in the MLE method. In particular, the singular value decomposition (SVD) of matrix \( M \) is

\[
M = U \Sigma V^T
\]

(III.42)

where the columns of \( V^T \) are orthonormal to each other. We choose the first \( n \) columns of \( V^T \) to be the \( n \) extra vectors of MLE-\( n \). The motivation is that the first \( n \) singular vectors are equivalent to the first \( n \) eigenvectors of the sample covariance matrix of the corresponding dataset, and the first \( n \) eigenvectors tend to form small angles with the majority of the vectors in the dataset.

Figure 2 and 3 are the plots of the performances of the algorithms using mean and 3 standard deviations, which compares the RMSE of RP, Li and CV, with the RMSE of MLE-\( n \), \( n = 1, 2, 3 \), and we use a log scale plot to visualize the differences.

For both of the datasets, we can see that the more extra vectors we add, the better the performance.
Furthermore, we evaluate the precision and recall of the algorithms when they are estimating the inner products. Firstly we define 39 thresholds from −0.95 to 0.95, with 0.05 difference between each two consecutive thresholds, i.e., −0.95, −0.9, −0.85, · · · , 0.95. Then we define the following terms:

- **True Positive (TP)** = the number of (i, j) pair where \( \hat{a}_{i,j} < \text{threshold} \) and \( \vec{x}_i^T \vec{x}_j < \text{threshold} \)
- **True Negative (TN)** = the number of (i, j) pair where \( \hat{a}_{i,j} > \text{threshold} \) and \( \vec{x}_i^T \vec{x}_j > \text{threshold} \)
- **False Positive (FP)** = the number of (i, j) pair where \( \hat{a}_{i,j} < \text{threshold} \) and \( \vec{x}_i^T \vec{x}_j > \text{threshold} \)
- **False Negative (FN)** = the number of (i, j) pair where \( \hat{a}_{i,j} > \text{threshold} \) and \( \vec{x}_i^T \vec{x}_j < \text{threshold} \)

where \( \hat{a}_{i,j} \) is the estimated inner product between \( \vec{x}_i, \vec{x}_j \), and \( \vec{x}_i^T \vec{x}_j \) is the exact inner product.
FIGURE 3. Comparison of RMSE and 3 standard deviations for approximating 183,315 pairs of inner products in DrivFace via random projections (RP), Li’s method (Li), control variates (CV), sign full random projections and our maximum likelihood estimator (MLE).

FIGURE 4. Comparison of average precision for approximating the 4,950 pairs of inner products in Arcene via random projections (RP), Li’s method (Li), control variates (CV), sign full random projections and our maximum likelihood estimator (MLE) at \( k = 10; 50; 100 \).

Based on this we define the two metrics:

\[
\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}. \quad (III.43)
\]

We look at the performance of the precision and recall of both datasets over different values of \( k \in \{10, 50, 100\} \) in Figures 4 to 11.

From the plots we observe that as more vectors are added, we get a higher average precision and recall as well as lower standard deviations.

IV. CONCLUSION AND FUTURE WORK

In this work we have investigated improving random projections using additional vectors. By establishing Theorem 6 and...
Theorem 9, the specific cubic equation that governs MLE is obtained for adding arbitrary \( n \) extra vectors; and the upper bound of variance reduction is obtained as well. As we can see, the variance of MLE will be continuously reduced as \( n \) increases under certain assumptions, meanwhile, the calculation of MLE can be performed at a reasonable cost of computation by choosing additional vectors to be orthonormal to each other. Theorem 9 also shows that the variance can be dramatically decreased provided the extra vectors have small angles with testing vectors of dataset, which suggests that we can pick these vectors by knowing some special characteristics of the dataset (e.g. singular vectors).

These work opens opportunities for studying other aspects of improving random projections using additional...
FIGURE 7. Comparison of 3 standard deviations of recall for approximating the 4,950 pairs of inner products in Arcene via random projections (RP), Li’s method (Li), control variates (CV), sign full random projections and our maximum likelihood estimator (MLE) at $k = 10; 50; 100$.

FIGURE 8. Comparison of average precision for approximating the 183,315 pairs of inner products in DrivFace via random projections (RP), Li’s method (Li), control variates (CV), sign full random projections and our maximum likelihood estimator (MLE) at $k = 10; 50; 100$.

information, and can be used as a step stone to develop parallel results of those in [10], such as various probability bounds. Last but not least, during the process of this work, authors also found that there is a close connection between MLE method and the method of control variates for improving random projections, which deserves special attention and will be dealt with in separate work.

APPENDICES
APPENDIX A
PROOF OF LEMMA 2

We first prove the first part of Lemma 2.

Proof: 1. Recall the notation $A = k \Sigma$ in (I.17), therefore,

$$|\Sigma| = \frac{1}{k^{2+r-n}} |A|; \quad \Sigma^{-1} = \frac{k}{|A|} \text{adj}(A).$$ (A.44)
Also, denote \( X = [\tilde{x}_1, \ldots, \tilde{x}_{2+n}] \), where \( \tilde{x}_i = [x_{i,1}, \ldots, x_{i,p}]^T, i = 1, \ldots, 2+n \). Further recall that we had the map \( T \equiv \frac{1}{\sqrt{k}} R \), with \( \tilde{v}_1 = TX_1, \ldots, \tilde{v}_{n+2} = TX_{n+2} \).

2. Taking the logarithm of the right-hand side of equation (I.16) and utilizing (A.44) gives

\[
l(a) = -\frac{k}{2} \log \frac{1}{k^{n+2}} |A| - \frac{k}{2 |A|} \sum_{j=1}^{k} \tilde{r}_j^T \text{adj}(A) \tilde{r}_j^{(n+2)} \]

\[
= -\frac{k}{2} \log \frac{1}{k^{n+2}} |A| - \frac{k}{2 |A|} \sum_{j=1}^{k} \tilde{r}_j^T X \text{adj}(A) X^T \tilde{r}_j
\]

\[
= -\frac{k}{2} \log \frac{1}{k^{n+2}} |A| - \frac{k}{2 |A|} \sum_{j=1}^{k} \tilde{r}_j^T Q \tilde{r}_j
\]

(denote \( Q = X \text{adj}(A) X^T \))

\[
= -\frac{k}{2} \log \frac{1}{k^{n+2}} |A| - \frac{1}{2 |A|} \sum_{j=1}^{k} \sum_{s=1}^{p} \sum_{t=1}^{p} Q_{s,t} r_{j,s} r_{j,t}
\]
\(Q_{s,t}\) is the \((s, t)\)-th entry of \(Q\), also notice \(Q\) is symmetric.

Differentiating both sides with respect to \(a\) yields
\[
I'(a) = -\frac{k}{2} \frac{|A'|}{|A|} - \frac{|A||\sum_{j=1}^{k} \sum_{p=1}^{P} Q_{s,t,r_j,s_t,r_j,t}'}{2|A|^2} + \frac{|A'||\sum_{j=1}^{k} \sum_{p=1}^{P} Q_{s,t,r_j,s_t,r_j,t}'}{2|A|^2}.
\]  \hspace{1cm} (A.45)

3. Keep in mind that \(Q\) is symmetric. By expanding the expression of \(Q\) and equating the corresponding entries, we obtain
\[
Q_{s,t} = \sum_{l=1}^{2+n} \sum_{i=1}^{2+n} A_{l,i} x_i x_l t, \quad \text{(A.46)}
\]
where \(A_{l,i}\) is the \((l, i)\)-th entry of \(\text{adj}(A)\) (notice that \(\text{adj}(A)\) is also symmetric). Substituting back we have
\[
I'(a) = -\frac{k}{2} \frac{|A'|}{|A|} - \frac{|A||\sum_{j=1}^{k} \sum_{p=1}^{P} \sum_{l=1}^{2+n} \sum_{i=1}^{2+n} A_{l,i} x_i x_l t r_j s_t r_j t}'}{2|A|^2} + \frac{|A'||\sum_{j=1}^{k} \sum_{p=1}^{P} \sum_{l=1}^{2+n} \sum_{i=1}^{2+n} A_{l,i} x_i x_l t r_j s_t r_j t}'}{2|A|^2}.
\]  \hspace{1cm} (A.47)

Observing \(\sum_{j=1}^{k} \sum_{p=1}^{P} Q_{s,t,r_j,s_t,r_j,t} = k \vec{v}_t^T \vec{v}_t\), we continue to simplify to obtain
\[
I'(a) = -\frac{k}{2} \frac{|A'|}{|A|} - \frac{k}{2} \frac{|A||\sum_{j=1}^{k} \sum_{p=1}^{P} \sum_{l=1}^{2+n} \sum_{i=1}^{2+n} A_{l,i} x_i x_l t r_j s_t r_j t}'}{|A|^2} + \frac{k}{2} \frac{|A'||\sum_{j=1}^{k} \sum_{p=1}^{P} \sum_{l=1}^{2+n} \sum_{i=1}^{2+n} A_{l,i} x_i x_l t r_j s_t r_j t}'}{|A|^2}.
\]  \hspace{1cm} (A.48)

where
\[
B = \sum_{l=1}^{2+n} \sum_{i=1}^{2+n} A_{l,i} \vec{v}_t^T \vec{v}_t. \quad \text{(A.49)}
\]

4. Setting \(I'(a) = 0\), we arrive at
\[
|A'||A| + B'|A| - B|A'| = 0. \quad \text{(A.50)}
\]

Now we prove the second part of Lemma 2.

Proof: 1. It is known that \(\text{Var}(\hat{a}) = -\mathbb{E}(i'll'(a))^{-1}\).

2. Utilizing equation (A.45), we compute the second derivative
\[
I''(a) = -\frac{k}{2} \frac{|A'|||A| - |A'|^2}{|A|^2} - \frac{k}{2} \frac{|A'B' - |A'|B|}{|A|^2} + \frac{k}{2} \frac{|A'|^2}{|A|^2} - \frac{k}{2} \frac{|A'B'|}{|A|^2} - \frac{k}{2} \frac{|A'B'|^2}{|A|^2} - \frac{k}{2} \frac{|A'B'|}{|A|^2} - \frac{k}{2} \frac{|A'B'|^2}{|A|^2}.
\]  \hspace{1cm} (A.51)

Therefore,
\[
\mathbb{E}(i'll'(a)) = \frac{k}{2} \frac{|A'|||A| - |A'|^2}{|A|^2} - \frac{k}{2} |A|^4 \left(2|A||A'|^2 - |A'|^2|A|^2\right) \mathbb{E}(B).
\]
To determine $c_2$, the key observation is that all the terms of $|A|$ involving $a$ must come from either $a_1 a_{1,2}$ or $a_{2,2}$ (since $\text{adj}(A)$ is symmetric, $A_{1,2} = A_{2,1}$). To see this, denoting the entries of $A$ as $x_i^a x_j = a_{ij}$ for simplicity and expanding $|A|$ by definition gives

$$|A| = \sum_{j_k \neq 2} (-1)^{1+j_2+\cdots+j_n} a_{1,j_1} a_{2,j_2} a_{3,j_3} \cdots a_{n,j_n}$$

$$= \sum_{j_k \neq 2} (-1)^{1+j_2+\cdots+j_n} a_{1,j_1} a_{2,j_2} a_{3,j_3} \cdots a_{n,j_n}$$

$$= \sum_{j_k \neq 2} (-1)^{1+j_2+\cdots+j_n} a_{1,j_1} a_{2,j_2} a_{3,j_3} \cdots a_{n,j_n}$$

where the symmetry of $A$ is used in the last line.

3. $c_3$ is yielded by evaluating $|A|$ at $a = 0$ in equation (B.55), or equivalently,

$$c_3 = \begin{bmatrix} a_{1,1} & 0 & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ a_{1,3} & a_{2,3} & a_{3,3} & \cdots & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{1,n} & a_{2,n} & a_{3,n} & \cdots & a_{n,n} \end{bmatrix}.$$ .

Substituting $c_1, c_2, c_3$ into (B.55) gives the desired result.

**APPENDIX C**

**PROOF OF LEMMA 4**

Proof: 1. Differentiating $A_{1,2}$ (the (1, 2)-th entry of $\text{adj}(A)$) with respect to $a$, we have $A_{1,2}' = -|D|$. By virtue of Lemma 3, we calculate

$$|A|' = 2a|D| + (2A_{1,2} - 2a|D|)$$

and $|A|'' = -2|D|$. By virtue of Lemma 3, we calculate

$$|A|'' = -2|D|.$$ .

Substituting $A_{1,2}', |A|', |A|''$ into equation (II.20), we obtain

$\frac{1}{\text{Var}(\hat{a})} = k \left( \frac{2|D| + 2A_{1,2}|A| + (4A_{1,2} - 2A_{1,2})|A|'}{|A|^2} \right)$

$= k \left( \frac{2|D|}{|A|} + \frac{2A_{1,2}|A|}{|A|^2} \right)$

$= k \left( \frac{2|D|}{|A|} + \frac{|A|'}{|A|^2} \right)$

$= k \left( \frac{2|D|}{|A|} + \frac{|A|'}{|A|^2} \right)$ .

which is the first desired equation.

**APPENDIX B**

**PROOF OF LEMMA 3**

Proof: 1. First it is clear that the determinant $|A|$ is a quadratic equation with respect to $a$, namely,

$$|A| = c_1 a^2 + c_2 a + c_3.$$ .

And using the definition of determinant of $A$, it can be directly verified $c_1 = -|D|$.

4. Hence we establish

$$\text{Var}(\hat{a}) = -E(ll'(a))^{-1}$$

$$= \frac{|A|^2}{k (|A|'' - 4A_{1,2})|A| + (4A_{1,2} - |A|')|A'|}.$$ .

This completes the whole proof.

**APPENDIX C**

**PROOF OF LEMMA 4**

Proof: 1. Differentiating $A_{1,2}$ (the (1, 2)-th entry of $\text{adj}(A)$) with respect to $a$, we have $A_{1,2}' = -|D|$. By virtue of Lemma 3, we calculate

$$|A|' = 2a|D| + (2A_{1,2} - 2a|D|)$$

and $|A|'' = -2|D|$. By virtue of Lemma 3, we calculate

$$|A|'' = -2|D|.$$ .

Substituting $A_{1,2}', |A|', |A|''$ into equation (II.20), we obtain

$\frac{1}{\text{Var}(\hat{a})} = k \left( \frac{2|D| + 2A_{1,2}|A| + (4A_{1,2} - 2A_{1,2})|A|'}{|A|^2} \right)$

$= k \left( \frac{2|D|}{|A|} + \frac{2A_{1,2}|A|}{|A|^2} \right)$

$= k \left( \frac{2|D|}{|A|} + \frac{|A|'}{|A|^2} \right)$

$= k \left( \frac{2|D|}{|A|} + \frac{|A|'}{|A|^2} \right)$ .

which is the first desired equation.
2. For the second equation, using $|A'| = 2A_{1,2}$ and Lemma 3 we see that
\[
\frac{2A_{1,2}|A'|}{|A|^2} = \left(\frac{2aA_{1,2}}{a|A|}\right)^2 = \frac{1}{a^2} \left(\frac{|A|}{|A|} - |D|a^2 - A_0\right)^2
\]
\[
= \frac{1}{a^2} \left(\frac{1 - A_0}{|A|} - |D|a^2\right)^2 = \frac{1}{a^2} \left(1 - \frac{A_0}{|A|}\right) + \left(\frac{D}{|A|}\right)^2 a^2 - 2\left(1 - \frac{A_0}{|A|}\right)|D|\frac{|A|}{|A|}.
\]
Substituting this into the second line of equation (C.58) and simplifying, we obtain
\[
\frac{1}{\text{Var}(\bar{a})} = k \frac{1}{2} \left(1 - \frac{A_0}{|A|}\right) + \left(\frac{D}{|A|}\right)^2 a^2 + 2\frac{|D|}{|A|}.
\]

**APPENDIX D**

**PROOF OF LEMMA 5**

**Proof:** 1. Without loss of generality, we assume that
\[
\frac{t_{1,m+1}^2}{t_{1,1}} = \max\{t_{1,m+1}^2, \ldots, t_{m,m}^2\}
\]
otherwise, we can always permute the rows and columns suitably to repeat the proof.

2. We now intend to diagonalize matrix $T_{m+1}$ by elementary row and column actions, which do not change the determinant $|T_{m+1}|$.

By the property of positive definite matrix it is clear that $t_{1,1} \neq 0$, we add the first row multiplied by $\frac{t_{m+1}}{t_{1,1}}$ to the last row, then add the first row multiplied by $\frac{t_{m+1}}{t_{1,1}}$ to the $(m-1)$th row, and so on, all the way to adding the first row multiplied by $\frac{t_{m+1}}{t_{1,1}}$ to the 2nd row. By doing so, we obtain
\[
\begin{bmatrix}
t_{1,1} & \cdots & t_{1,m+1} & 0 & \cdots & t_{2,m+1} - \frac{t_{1,m+1}t_{2,1}}{t_{1,1}} \\
0 & \cdots & 0 & \cdots & \cdots & \cdots \\
\vdots & \cdots & \vdots & \cdots & \cdots & \vdots \\
0 & \cdots & t_{m,m+1} & 0 & \cdots & t_{m+1,m+1} - \frac{t_{m+1,m}t_{m+1}}{t_{1,1}}
\end{bmatrix}
\]

(D.60)

Similarly, we do elimination using column actions to obtain
\[
\begin{bmatrix}
t_{1,1} & 0 & \cdots & 0 & 0 \\
0 & t_{2,2} - \frac{t_{1,2}^2}{t_{1,1}} & \cdots & t_{2,m+1} - \frac{t_{1,m+1}t_{2,1}}{t_{1,1}} & \cdots \\
\vdots & \cdots & \vdots & \cdots & \cdots \\
0 & \cdots & 0 & t_{m,m+1} - \frac{t_{m+1,m}t_{m,m}}{t_{1,1}} & \cdots \\
0 & \cdots & 0 & \cdots & t_{m+1,m+1} - \frac{t_{m+1,m+1}t_{m+1}}{t_{1,1}}
\end{bmatrix}
\]

(D.61)

3. Note that $t_{2,2} - \frac{t_{1,2}^2}{t_{1,1}} \neq 0$. We continue to proceed analogously and eventually get a diagonal matrix with diagonal entries:
\[
[t_{1,1}, \ldots, t_{m,m} - \sum_{i=1}^{m} d_i^2, t_{m+1,m+1} - \sum_{i=2}^{m-1} l_i^2]
\]

for suitable $d_i$’s, $l_i$’s and it is guaranteed by the assumption in lemma that each entry has to be positive value. Since elementary row and column actions do not change the determinant,
\[
det(T_{m+1}) = \det(T_m) \left(t_{m+1,m+1} - \frac{t_{m+1}^2}{t_{1,1}} - \sum_{i=2}^{m} l_i^2\right)
\]

\[
\leq c \cdot \det(T_m),
\]

where
\[
c = t_{m+1,m+1} - \max\{t_{m+1,m+1}, \ldots, t_{m,m}^2\}
\]

**APPENDIX E**

**PROOF OF LEMMA 8**

**Proof:** 1. For $|A|$, we directly apply Lemma 3 and note $a^2 = \frac{1}{a} |A| = A_{1,2}$ to obtain
\[
|A| = -\frac{1}{a} |a^2 + 2a(a|A| + |A|)^2 + |A| = a^2 + a_2 a + a_3,
\]
where $a_1, a_2, a_3$ are as in (II.37).

2. To compute $B$, first recall in equation (II.18) of Lemma 2 that
\[
B = \sum_{l=1}^{n+2} \sum_{i=1}^{n+2} \tilde{A}_{l,i,\vec{v}_l} (\vec{v}_l)^T, \quad A_{l,\vec{v}_l} \text{ is } (l, \vec{v}_l)\text{-th entry of } \text{adj}(A).
\]

(E.61)

To visualize $B$, using Definition 7 we can rewrite $B$ as
\[
B = \delta(A, \tilde{A}_{l,\vec{v}}) = |A_1| + \cdots + |A_{n+2}|,
\]

(E.62)

where
\[
\tilde{v}_l = (v_{l1}^T, v_{l2}^T, v_{l3}^T, \cdots, v_{ln+2}^T)^T,
\]

(E.63)

$i = 1, \ldots, n+2$ and matrices $A_i$ ($i = 1, \ldots, n+2$) are generated by replacing the $i$-th row of matrix $A$ with $\tilde{v}_l$.

Let us analyze each $A_i$ separately. It can be directly verified with the aid of Lemma 3 and Definition 7 that
\[
|A_1| = (A, \tilde{A}_{l,\vec{v}})a + (A, \tilde{A}_{l,\vec{v}}),
\]

and
\[
|A_2| = (A, \tilde{A}_{l,\vec{v}})a + (A, \tilde{A}_{l,\vec{v}}),
\]

for $i = 3, \ldots, n+2$. 


Substituting these back into \( B \) in (E.62), we obtain
\[
B = \left( \sum_{i=3}^{n+2} (\bar{A}, \bar{I}_i) \right) a^2
+ \left( 4, \bar{A}_1 \right) + \sum_{i=3}^{n+2} (\bar{A}, \bar{I}_i) + \left( 2, \bar{A}_2 \right) + \sum_{i=3}^{n+2} (\bar{A}, \bar{I}_i)
\]
which is the desired result and completes the whole proof.

**APPENDIX F**

**PROOF OF THEOREM 6**

**Proof:** 1. Proof for the first part of theorem is established first.

Since \( \bar{x}_3, \ldots, \bar{x}_{n+2} \) are all orthogonal to both \( \bar{x}_1 \) and \( \bar{x}_2 \), going back to Lemma 3, the matrix \( B \) is a zero matrix, \( |A| = (\|\bar{x}_1\|^2 - a^2) |D| \) and \( A_{1,2} = -a |D| \). Therefore, by using (II.20) in Lemma 2, we directly calculate
\[
\text{Var}(\hat{a}) = 2 \left( \frac{|A|^2}{k} \right) \left( \frac{|A|^2}{k} \right) = \frac{2}{k} \frac{|A|^2}{|A|^2} = 1, \quad \forall a \geq 1.
\]
which does not depend on \( n \).

2. For the second part of theorem. According to our assumption that \( \bar{x}_3, \ldots, \bar{x}_{n+2} \) are orthogonal to each other, the matrix \( A \) in equation (I.14) can be written as
\[
a_{i,j} = \begin{cases} 0 & i, j \geq 3, i \neq j \\ \bar{x}_i^T \bar{x}_j & \text{otherwise} \end{cases}
\]

Let \( \bar{A}_i \) for \( i = 2 \leq n \leq 2n \) denote the matrix of size \((i - 1) \times (2n - 1)\) generated by eliminating the \( i \)-th all the way to \((n + 2)\)-th rows and eliminating the \( i \)-th all the way to \((n + 2)\)-th columns of matrix \( A \). Applying Lemma 5 inductively to matrix \( A \), we obtain
\[
|A| \leq (n + 2, n+2 - s_n + 2a_{n+2,n+2}) |\bar{A}_{n+2}|
\leq (n + 2, n+2 - s_n + 2a_{n+2,n+2}) |\bar{A}_{n+1}|
\leq (n + 2, n+2 - s_n + 2a_{n+2,n+2}) |\bar{A}_{n+1}|
\leq (3, 3 - a_3, 3) \cdots (n + 2, n+2 - s_n + 2a_{n+2,n+2}) |\bar{A}_{n+1}|
= |\bar{x}_3|^2 \cdots |\bar{x}_{n+2}|^2 (|\bar{x}_1|^2 - a^2)^2 \prod_{i=3}^{2n+2} (1 - s_n).
\]

Substituting back into the first equation of Lemma 4 and noting that
\[
|D| = |\bar{x}_3|^2 \cdots |\bar{x}_{n+2}|^2,
\]
we arrive at the desired result
\[
\text{Var}(\hat{a}, n) \leq \frac{1}{k} \left( \frac{2}{|\bar{x}_1|^2 - a^2} \right) \prod_{i=3}^{2n+2} (1 - s_n) + \log |A|'.
\]

**APPENDIX G**

**PROOF OF THEOREM 9**

**Proof:** Directly substitute equation (II.36) into equation (II.18) for \( |A| \) and \( B \), simplify and expand it, and then apply (II.37).

**APPENDIX H**

**PROCEDURES**

Since the \( n \) extra vectors are chosen to be normalized and orthogonal, \( D \) is an identity matrix in formula
\[
\det \left( \begin{bmatrix} C_1 & C_2 \\ \bar{C}_2 & D \end{bmatrix} \right) = \det(D) \det(C_1 - C_2D^{-1}C_3).
\]

By virtue of this simplified formula, the specific expressions of \( \alpha_1, \alpha_2, \alpha_3 \) and \( \beta_1, \beta_2, \beta_3 \) can be calculated as follows.

**A. PROCEDURE 1**

Using definitions of \( \alpha 's \) in (II.39) and formula (H.66), we directly calculate
\[
\begin{align*}
\alpha_1 &= -1, \\
\alpha_2 &= 2 \sum_{t=3}^{2n+2} (\bar{x}_t^T \bar{x}_t), \\
\alpha_3 &= (1 - \sum_{t=3}^{2n+2} (\bar{x}_t^T \bar{x}_t)^2)(1 - \sum_{t=3}^{2n+2} (\bar{x}_t^T \bar{x}_t)^2) \\
&\quad - \sum_{t=3}^{2n+2} (\bar{x}_t^T \bar{x}_t)(\bar{x}_t^T \bar{x}_t).
\end{align*}
\]

This requires summing up values of the form \((\bar{x}_t^T \bar{x}_t)\) which we would have pre-computed and stored in Phase 2.

**B. PROCEDURE 2**

Using Definition 7 and (II.37) and formula (H.66), we compute the following.
\[
\begin{align*}
(a). \quad & \beta_1 = \mathbb{N}(A, \{\bar{I}_1\}) \\
& \frac{1}{(A, \bar{I}_1)} = (A, \bar{I}_2) = 0 \\
& \frac{1}{(A, \bar{I}_1)} = -\bar{v}_1^T \bar{v}_1, \quad i = 3, \ldots, 2+n.
\end{align*}
\]

Therefore
\[
\beta_1 = - \sum_{i=3}^{2+n} \bar{v}_i^T \bar{v}_i.
\]

(b). \( \beta_2 = \mathbb{N}(A, \{\bar{I}_1\}) + \mathbb{N}(A, \{\bar{I}_1\}) \).
First compute $\text{N}(\hat{A}, \{z_i^l\})$ to obtain:

$$
\begin{align*}
2^n \cdot (\hat{A}, \{z_i^2\}) & = 0 \\
2^n \cdot (\hat{A}, \{z_i^2\}) & = \sum_{i=3}^{2+n} v_i^T v_i x_i^T x_i - v_i^T v_i \\
2^n \cdot (\hat{A}, \{z_i^2\}) & = \sum_{i=3}^{2+n} (x_i^T x_i v_i^T v_i - v_i^T v_i x_i^T x_i) \\
& \quad + \sum_{i=3}^{2+n} v_i^T v_i, \\
& \quad i = 3, \ldots, 2 + n. \quad (\text{H.70})
\end{align*}
$$

We can also compute $\text{N}(\hat{A}, \{z_i^4\})$ to obtain:

$$
\begin{align*}
4^n \cdot (\hat{A}, \{z_i^4\}) & = 0 \\
4^n \cdot (\hat{A}, \{z_i^4\}) & = \sum_{i=3}^{2+n} v_i^T v_i x_i^T x_i - v_i^T v_i \\
4^n \cdot (\hat{A}, \{z_i^4\}) & = \sum_{i=3}^{2+n} (x_i^T x_i v_i^T v_i - v_i^T v_i x_i^T x_i) \\
& \quad + \sum_{i=3}^{2+n} v_i^T v_i, \\
& \quad i = 3, \ldots, 2 + n. \quad (\text{H.71})
\end{align*}
$$

Hence, $\beta_2$ is obtained by summing up all of them.

(c). Last, let us compute $\beta_3 = \text{N}(\hat{A}, \{z_i^3\})$.

$$
\begin{align*}
3^n \cdot (\hat{A}, \{z_i^3\}) & = (v_i^T v_i - \sum_{i=3}^{2+n} v_i^T v_i x_i^T x_i)(1 - \sum_{i=3}^{2+n} x_i^T x_i) \\
& \quad + \sum_{i=3}^{2+n} x_i^T x_i x_i^T x_i (v_i^T v_i - \sum_{i=3}^{2+n} v_i^T v_i x_i^T x_i) \\
3^n \cdot (\hat{A}, \{z_i^3\}) & = (v_i^T v_i - \sum_{i=3}^{2+n} v_i^T v_i x_i^T x_i)(1 - \sum_{i=3}^{2+n} x_i^T x_i) \\
& \quad + \sum_{i=3}^{2+n} x_i^T x_i x_i^T x_i (v_i^T v_i - \sum_{i=3}^{2+n} v_i^T v_i x_i^T x_i) \\
3^n \cdot (\hat{A}, \{z_i^3\}) & = (v_i^T v_i - \sum_{i=3}^{2+n} v_i^T v_i x_i^T x_i)(1 - \sum_{i=3}^{2+n} x_i^T x_i) \\
& \quad - \frac{1}{v_i^T v_i} (\hat{A}, \{z_i^2\}) \times (\hat{A}, \{z_i^2\}) \\
& \quad \times \frac{2}{v_i^T v_i}, \\
& \quad i = 3, \ldots, 2 + n. \quad (\text{H.72})
\end{align*}
$$

Hence, $\beta_3$ is obtained by summing up all of them.

**ACKNOWLEDGMENT**

The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

This work was supported in part by the Ministry of Education, Singapore, through an Academic Research Fund Tier 2, under Grant MOE2018-T2-2-013, and in part by the MOE SUTD Faculty Early Career Award under Grant RGFECA17003.

**REFERENCES**

[1] M.-F. Balcan, A. Blum, and S. Vempala, “Kernels as features: On kernels, margins, and low-dimensional mappings,” *Mach. Learn.*, vol. 65, no. 1, pp. 79–94, Oct. 2006.

[2] K. Liu, H. Kargupta, and J. Ryan, “Random projection-based multiplicative data perturbation for privacy preserving distributed data mining,” *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 1, pp. 92–106, Jan. 2006.

[3] E. Bingham and H. Mannila, “Random projection in dimensionality reduction: Applications to image and text data,” in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: ACM, 2001, pp. 245–250.

[4] S. Paul, C. Boutisidis, M. Magdon-Ismail, and P. Drineas, “Random projections for support vector machines,” in *Proc. 16th Int. Conf. Artif. Intell. Statist.*, 2013, pp. 498–506.

[5] C. Boutisidis, A. Zouraris, and P. Drineas, “Random projections for $k$-means clustering,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 298–306.

[6] R. Friedman, L. Moreno, and S. Vallejo, “Some hypothesis tests based on random projection,” *Comput. Statist.*, vol. 32, no. 3, pp. 1165–1189, Sep. 2017.

[7] L. N. Geppert, K. Ickstadt, A. Manteuau, J. Quedenfeld, and C. Soehler, “Random projections for Bayesian regression,” *Statist. Comput.*, vol. 27, no. 1, pp. 79–101, Jan. 2017.

[8] M. S. Charikar, “Similarity estimation techniques from rounding algorithms,” in *Proc. 34th Ann. ACM Symp. Theory Comput. (STOC)*. New York, NY, USA: ACM, 2002, pp. 380–388.

[9] S. S. Vempala, *The Random Projection Method*, vol. 65. Providence, RI, USA: AMS, 2005.

[10] P. Li, T. J. Hastie, and K. W. Church, “Improving random projections using marginal information,” in *Learning Theory (Lecture Notes in Computer Science)*, vol. 4005. Berlin, Germany: Springer, 2006, pp. 635–649, doi: 10.1007/11776420_46.

[11] G. Blanchard, “Generalization error bounds for aggregate classifiers,” in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 693–701.

[12] A. Kabn, “Improved bounds on the dot product under random projection and random sign projection,” in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 487–496.

[13] A. Kabin, “Dimension-free error bounds from random projections,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 4049–4056.

[14] P. W. Glynn and R. Schezman, “Some new perspectives on the method of control variates,” in *Monte Carlo and Quasi-Monte Carlo Methods 2000*. Berlin, Germany: Springer, 2002, pp. 27–49.

[15] P. Li, K. W. Church, and T. J. Hastie, “Conditional random sampling: A sketch-based sampling technique for sparse data,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 873–880.

[16] K. Kang and G. Hooker, “Random projections with control variates,” in *Proc. 6th Int. Conf. Pattern Recognit. Appl. Methods (ICPRAM)*, 2017, pp. 138–147.

[17] K. Kang and W. Wong, “Improving sign random projections with additional information,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2479–2487.

[18] D. Achlioptas, “Database-friendly random projections: Johnson-Lindenstrauss with binary coins,” *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 671–687, Jun. 2003, doi: 10.1016/S0022-0000(03)00025-4.

[19] P. Li, T. J. Hastie, and K. W. Church, “Very sparse random projections,” in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: ACM 2006, pp. 287–296, doi: 10.1145/1150402.1150436.

[20] E. Liberty, N. Ailon, and A. Singer, “Dense Fast Random Projections and Lean Walsh Transforms,” in *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (Lecture Notes in Computer Science)*, vol. 5171, A. Goel, J. K. Jansen, J. D. P. Rolim, and R. Rubinfield, Eds. Boston, MA, USA: Springer, 2008, pp. 512–522.
[21] C. Boutsidis and A. Gittens, “Improved matrix algorithms via the sub-sampled randomized Hadamard transform,” SIAM J. Matrix Anal. Appl., vol. 34, no. 3, pp. 1301–1340, Jan. 2013.

[22] J. Ji, J. Li, S. Yan, B. Zhang, and Q. Tian, “Super-bit locality-sensitive hashing,” in Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 108–116.

[23] K. Kang, “Random projections with Bayesian priors,” in Proc. Nat. CCF Conf. Natural Lang. Process. Chin. Comput. (NLPCC), Dalian, China, Nov. 2017, pp. 170–182, doi: 10.1007/978-3-319-73618-1_15.

[24] K. Kang, “Using the multivariate normal to improve random projections,” in Proc. Int. Conf. Intell. Data Eng. Automated Learn. Guilin, China: Springer, 2017, pp. 397–405.

[25] P. Li, “Sign-full random projections,” in Proc. AAAI Conf. Artif. Intel., vol. 33, Jul. 2019, pp. 4205–4212.

[26] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, “Result analysis of the nips 2003 feature selection challenge,” in Proc. Adv. Neural Inf. Process. Syst., 2005, pp. 545–552.

[27] K. Diaz-Chito, A. Hernández-Sabaté, and A. M. López, “A reduced feature set for driver head pose estimation,” Appl. Soft Comput., vol. 45, pp. 98–107, Aug. 2016.

YULONG LI received the master’s degree in mathematical physics from Capital Normal University, Beijing, in 2015, and the Ph.D. degree in applied mathematics from the University of Wyoming, USA, in 2019. Since 2019, he has been working as a Research Fellow of the Singapore University of Technology and Design. His research interests include random projections, general fractional calculus theory, and fractional-order differential equations. During the Ph.D. degree, he was awarded the UW Science Initiative Scholarship, in 2018, and he is also nominated for the UW Outstanding Dissertation Award by the Department of Mathematics and Statistics.

ZHIHAO KUANG received the bachelor’s degree in applied mathematics from NUS, through the SM2 Scholarship Program. He possesses statistical knowledge, including regression, classification, and simulation, and computer programming practices, including MATLAB projects, data analytics, and webpage editing. Previously, he also worked as a Research Intern with DNV GL, where the job scope included designing data model for ship fuel, sedimentation, and productivity.

JIANG YAN LI is currently pursuing the bachelor’s degree with the Singapore University of Technology and Design (SUTD). He likes teaching and has been involved as an undergraduate teaching assistant in freshman engineering calculus and ordinary differential equations. His research interests include applied mathematics, mathematical programming, and electronic circuit engineering.

KEEGAN KANG (Member, IEEE) received the B.Sc. (MMORSE) degree from Warwick University, in 2012, and the Ph.D. degree in statistics from Cornell University, Ithaca, NY, USA, in 2017. From 2017 to 2019, he was a Faculty Fellow of the Singapore University of Technology and Design, Singapore. He is currently working on improving hashing algorithms with statistical techniques. He has published in conferences such as the International Conference of Machine Learning (ICML) and the International Conference on Natural Language Processing and Chinese Computing (NLPCC). His research interests are in LSH algorithms, machine learning, statistical theory, and educational pedagogy. More recently, he was invited to give a main talk at the High Dimensional Data Mining (HDM) Workshop at the IEEE International Conference on Data Mining (ICDM), in 2019, although his Postdoctoral researcher (and coauthor Y. Li) ended up giving the talk instead due to unexpected commitments.

* * *