Data Mining and Knowledge Discovery (2021) 35:1622–1653
https://doi.org/10.1007/s10618-021-00764-6

Correlations between random projections and the bivariate normal

Keegan Kang[1]

Received: 4 February 2020 / Accepted: 4 May 2021 / Published online: 18 May 2021

Abstract
Random projections is a technique primarily used in dimension reduction by mapping high dimensional data to a low dimensional space, preserving pairwise distances in expectation, such as the Euclidean distance, inner product, angular distance, and l_p distance for values of p which are even. These estimated pairwise distances between observations in the low dimensional space can be rapidly computed to be used for nearest neighbor searches, clustering, or even classification. This paper highlights how these two disparate topics have a common thread, and expand upon two computational statistical techniques in recent random projection literature to further improve the accuracy of the estimate of the inner product between vectors under random projection by making use of the properties of the respective dataset, as well as limitations of these methods.

Keywords Bayesian inference · Bivariate normal · Control variate · Data mining · Estimating inner products · Multivariate normal · Random projection

1 Introduction

The basic random projection technique assumes a data matrix $X_{n\times p}$ with n observations and p features. Without loss of generality, assume each observation $\mathbf{x}_i$ is normalized to have a length of 1. We generate a random matrix $R_{p\times k}$ where r_{ij} are i.i.d. $N(0, 1)$ and compute the matrix product $V := XR$.

Responsible editor: Fei Wang.

This work is funded by the SUTD Faculty Fellow Grant RGFECA17003 as well as the Singapore Ministry of Education Academic Research Fund Tier 2 Grant MOE2018-T2-2-013.

✉ Keegan Kang
keegan_kang@sutd.edu.sg

[1] Singapore University of Technology and Design, Singapore, Singapore

Suppose we pick $\mathbf{v}_1, \mathbf{v}_2$ the first two rows of V and look at the tuples $\mathbf{w}_s = (v_{1s}, v_{2s})$. The expectation $\mathbb{E}[v_{1s}v_{2s}]$ is given by

$$\mathbb{E}\left[\sum_{i=1}^{p} x_{1i}x_{2i}r_{is}^2 + 2\sum_{1\leq i<j\leq p} x_{1i}x_{2j}r_{is}r_{js}\right] = \langle \mathbf{x}_1, \mathbf{x}_2\rangle \tag{1}$$

which gives us $\sum_{i=1}^{p} x_{1i}x_{2i}$, the inner product between the vectors $\mathbf{x}_1, \mathbf{x}_2$. The Law of Large Numbers guarantees that if k is sufficiently large, then the sum $\frac{\sum_{s=1}^{k} v_{1s}v_{2s}}{k}$ converges to the inner product $\langle \mathbf{x}_1, \mathbf{x}_2\rangle$. Using moment generating functions (Casella and Berger 2001) with some multivariate theory (Mardia et al. 1979), we can show that the variance

$$\text{Var}[v_{1s}v_{2s}] = 1 + \left(\sum_{i=1}^{p} x_{1i}x_{2i}\right)^2 = 1 + \langle \mathbf{x}_1, \mathbf{x}_2\rangle^2 \tag{2}$$

and with k observations, we have

$$\text{Var}\left[\frac{\sum_{s=1}^{k} v_{1s}v_{2s}}{k}\right] = \frac{1 + \langle \mathbf{x}_1, \mathbf{x}_2\rangle^2}{k} \tag{3}$$

We can also compute a Chernoff type bound (Vempala 2004) to find a value of k where the sum is within some $1 \pm \epsilon, \epsilon > 0$ of the true inner product, which we give here without proof

$$\mathbb{P}\left[\left|\frac{\langle \mathbf{v}_i, \mathbf{v}_j\rangle}{k} - \langle \mathbf{x}_i, \mathbf{x}_j\rangle\right| < \epsilon\right] \leq 1 - 4\exp\left\{-(\epsilon^2 - \epsilon^3)k/4\right\} \tag{4}$$

We additionally note that the bound in (4) is independent of p. This implies that regardless of the dimensionality of the original data, the user can project down to the same k dimensions and keep the same relative error. Tighter bounds independent of p can be computed as well (Kaban 2015).

This gives forth a recipe for the use of random projections to pre-process high dimensional data. For some degree of accuracy based on ϵ, the practitioner can generate V which gives the estimates of the inner product with high probability. If we scale the matrix V by $\frac{1}{\sqrt{k}}$, then the matrix can be used as a proxy of X in distance based computational algorithms, since the norms of each row in V, Euclidean distances and inner products of pairwise vectors in V are unbiased estimates of the actual values of the norms of each row in X, and Euclidean distances and inner products of pairwise vectors in X respectively. Table 1 shows the respective common estimates of distances, with variances given without proof.

Table 1 Estimations using random projections

Distance	Actual value	Estimate	Variance
Norm of vector	$\|\mathbf{x}_i\|_2^2$	v_{is}^2	$2\|\mathbf{x}_i\|_2^4$
Sq. Euclidean Distance	$\|\mathbf{x}_i - \mathbf{x}_j\|_2^2$	$(v_{is} - v_{js})^2$	$2\|\mathbf{x}_i - \mathbf{x}_j\|_2^4$
Inner product	$\langle \mathbf{x}_i, \mathbf{x}_j \rangle$	$v_{is} v_{js}$	$1 + \langle \mathbf{x}_1, \mathbf{x}_2 \rangle^2$

Alternatively, we can also look at the distribution of the tuples $\{(v_{1s}, v_{2s})\}_{s=1}^k$. Both v_{1s}, v_{2s} for $0 \leq s \leq k$ are weighted sums of i.i.d $N(0, 1)$ vectors which have mean 0 and variance of 1. While each tuple (v_{1s}, v_{2s}) are i.i.d., v_{1s} and v_{2s} are correlated with correlation given by $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ as computed in (1). In fact, the tuples are seen as drawn from a bivariate normal

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & a \\ a & 1 \end{pmatrix} \right) \tag{5}$$

where we denote a to be the inner product between $\mathbf{x}_1, \mathbf{x}_2$. Li et al. (2006a) were one of the first who made use of this fact, and used a maximum likelihood estimator to improve upon the estimate of a. Concretely, the likelihood of a is proportional to

$$l(a) \propto (1 - a^2)^{-k/2} \exp\left\{ - \sum_{s=1}^{k} \frac{(v_{1s} \; v_{2s}) \begin{pmatrix} 1 & -a \\ -a & 1 \end{pmatrix} (v_{1s} v_{2s})^T}{2(1 - a^2)} \right\} \tag{6}$$

and the maximum likelihood estimator of $\hat{a}$ is given to be the root of the following cubic

$$f(a) = a^3 - \left(\sum_{s=1}^{k} v_{1s} v_{2s} \right) a^2 + \left(\sum_{s=1}^{k} (v_{1s}^2 + v_{2s}^2) - 1 \right) a - \left(\sum_{s=1}^{k} v_{1s} v_{2s} \right) \tag{7}$$

which could be solved by numerical approximation methods. This maximum likelihood estimator is asymptotically unbiased (Shao 2003), and converges to $N\left(a, \frac{1}{I(a)}\right)$, where $I(a)$ is the Fisher information, with variance given by

$$\text{Var}\left[\hat{a}\right] = \frac{(1 - a^2)^2}{k(1 + a^2)} \tag{8}$$

We now take a look at an unrelated problem in estimating the correlation in bivariate normal data when there is a small sample size n. In 2011, Alkema et al. (2011) looked at projecting the total fertility rate of all countries. However, Fosdick and Raftery (2012) were concerned that there may be correlations in the fertility rate between countries in this study. They modelled each pair of countries as a bivariate normal, and considered estimators which could estimate the correlation ρ between countries. Follow up work

by Fosdick and Perlman (2016) and Fu et al. (2013) include performing inference on the correlation coefficient ρ.

The common thread tying these two disparate topics is simply estimating the value of a (equivalently ρ), when the number of observations is small. However, the number of observations is small in the first case due to dimension reduction, but small in the second case due to constraints on observational data.

The paper will be structured as such. Section 2 will cover similarities and differences between random projections and the bivariate normal. Section 3 will give our main contributions in this manuscript and highlight how they are different from prior work. Section 4 will cover the first computational statistics method, control variates with the multivariate normal. Section 5 will cover the second computational statistics method, using Bayesian inference. Section 6 shows the experimental results of both methods. Section 7 discusses the experimental results. Section 8 concludes this paper with potential future work and applications of the algorithms.

2 Random projections and the bivariate normal—two sides of the same coin

While researchers facing both problems derived similar results independently, they went in different directions. This is not surprising since they had very different goals, despite sharing something in common - the estimation of the inner product (respectively, correlation coefficient).

In applications of random projections, the goal is to quickly compute estimated inner products between any pair of vectors $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^p$, and we briefly describe two of these applications here, similarity search and clustering.

For similarity search, the data matrix $X_{n\times p}$ is usually too large to be stored in memory. Hence, directly computing the inner product between any two $\mathbf{x}_i, \mathbf{x}_j$ involves retrieving the vectors from hardware (which could take some time), as well as computing the inner product (which has time complexity of $O(p)$). Alternatively, we could use random projections to compute the matrix $V_{n\times k}$ which comes with a time complexity of $O(npk)$ and store this in memory. Then $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ can be computed with a time complexity of $O(k)$.

For clustering purposes, distances between points need to be computed. When the number of features p is large, the computational time taken to cluster is proportional to p. But as distances are preserved under random projections, the vectors $\mathbf{v}_i$ in V could be clustered instead, with the computational time taken to cluster being proportional to k. When $k \ll p$, random projections are preferred.

Research in random projections from Achlioptas (2003), Ailon and Chazelle (2009), Li et al. (2006a), Liberty et al. (2008) focuses on the tradeoff between computational cost, storage, and accuracy of the estimated distances.

On the other side of the coin, research in estimating the correlation coefficient places no importance on the computational cost or storage, but more importance on the type of estimator depending on the data and prior knowledge.

We give a few results from random projections and statistical inference here.

Some examples of independent yet identical work include (Li et al. 2006a) re-deriving a maximum likelihood estimator for the correlation a in (5) for random projections, which was already known in Madansky (1965). Fosdick and Raftery (2012) also proposed an empirical estimator for the correlation ρ when there are small sample sizes, which was $\frac{\sum_{s=1}^{k} v_{1s} v_{2s}}{k}$, the original random projection estimate.

Several results from both sides can complement the other as well. Viewing the estimation of the correlation of a bivariate normal as a random projection helps in placing bounds on the estimators suggested by Fosdick and Raftery (2012). We can rewrite (4) as

$$\mathbb{P}\left[\left|\frac{\mathrm{SS}_{xy}}{n} - \rho\right| < \epsilon\right] \leq 1 - 4\exp\left\{-\epsilon^2(1-\epsilon)n/4\right\} \tag{9}$$

in the notation of Fosdick and Raftery. We can further place other bounds on these estimators from random projection literature (Vempala 2004; Li et al. 2006a; Kaban 2015).

Appealing to the Central Limit Theorem allows us to use other i.i.d. r_{ij} in the random matrix R, not just $N(0, 1)$, where the results for the normal distribution hold as p gets large (Li et al. 2006b). This is of particular interest in speeding up the random projection matrix multiplication. Probability distributions where the first moment is zero and the second moment 1 such as the Rademacher distribution (Achlioptas 2003) ($r_{ij} \in \{-1, 1\}$), or sparse distributions (Li et al. 2006b) ($r_{ij} \in \{-1/\sqrt{s}, 0, 1/\sqrt{s}\}$) can be used. Matrix multiplication when entries r_{ij} come from these distributions are faster, since the entries of R are integers (or mostly zeroes).

More could be done to exploit the relationship between random projections and estimation of the correlation coefficient. Datasets which are used for random projection algorithms can be used by statisticians to test the efficacy of their proposed inference on the estimated correlation of the bivariate normal. Computer scientists can look at inference done on the estimated correlation to see if two vectors are mutually orthogonal or not, i.e. testing if $a = 0$. To the best of our knowledge, we know of no such work mentioned in this paragraph.

Finally, we could tie the knot between statistics and computer science for random projections, by using different statistical techniques to improve random projections, based on the type of dataset given.

We highlight the constraints we require for random projections before moving on to the next section. Recall that the purpose of random projections is to speed up the computation of pairwise distances between vectors. Without random projection, the computational complexity to compute all pairwise distances would be $O(n^2 p)$. With random projections, the computational complexity would be $O(npk + n^2 k)$.

In practice, the pre-processing period to compute V (which has a computational complexity of $O(npk)$) could be ignored in some circumstances. For example, if X is too large to store in memory, then the matrix V could be computed, and subsequent computations and analysis done on V in perpetuity.

Any statistical techniques for random projections need to be have at most computational complexity of $O(n^2 k)$, which would imply a computational complexity of $O(k)$

to estimate the distance between a vector pair rather than a computational complexity of $O(p)$. This means that higher order computational complexity in estimating the the correlation coefficient for a bivariate normal is not feasible.

3 Our contributions

In this paper, we synthesize the prior works done (Kang 2017a, b; Kang and Hooker 2017a) to give a rigorous treatment of the control variate estimator of the inner products via random projection, and give insight on when we should use these methods to estimate the inner product between vectors. More explicitly,

1. We give a demonstration of how the method of control variates can be used to estimate any quadratic form in the bivariate normal case with lower variance, as shown in Theorem 1.
2. We explain how to choose a basis of random variables for the control variate technique, using prior work as illustration.
3. We rigorously show that variance reduction always exists for control variates using additional vectors. Moreover, we show that there is an exact form of the control variate correction and the coefficients with an arbitrary amount of extra vectors, given in Theorem 2, Corollary 3 and 4. This allows the user to use these values directly, instead of optimizing over more and more variables with multiple extra vectors.
4. We extend the experiments in our prior works to not just to look at the reduction in overall RMSE across all vector pairs, but also to show the increase in precision and recall in retrieving vector pairs whose true inner product ρ lie between intervals $a \leq \rho \leq b$. This allows our methods to be used widely in more applications.
5. We discuss the performance of the methods in our prior works on several datasets, and give motivation for their performance. This allows users to choose which methods to use based on the distribution of their data.

4 Control variates and the multivariate normal

In this section, we describe how control variates can be used to improve the estimates of the inner product when using random projections. We expand and unify existing work on control variates for the bivariate normal (Kang and Hooker 2017a), and COVFEFE (COntrol Variates For Estimation via First Eigenvectors) for the multivariate normal (Kang 2017b). We show the relationship between control variates and Li et al. (2006a)'s maximum likelihood estimator. We also describe how control variates could be used in getting better estimates of the correlation coefficient for a bivariate normal in observational studies.

4.1 Control variates

Control variates have been used since the late 1970s (Lavenberg and Welch 1981), and is a technique for reducing variance in Monte Carlo simulations, by looking at correlated errors from the same random numbers.

Suppose we have a random number generator, and we use the generator to generate some random variable Y, of which we are interested in getting the estimate $\mathbb{E}[Y]$. Further suppose we use the same random numbers from the generator to generate a random variable Z, but we know the true mean $\mathbb{E}[Z] = \mu_Z$. For any c, we have that $(Y + c(Z - \mu_Z))$ is an unbiased estimator of Y, as

$$\mathbb{E}[Y + c(Z - \mu_Z)] = \mathbb{E}[Y] + 0 \tag{10}$$

The variance of $(Y + c(Z - \mu_Z))$ is given by

$$\text{Var}[Y + c(Z - \mu_Z)] = \text{Var}[Y] + c^2 \text{Var}[Z] + 2c \text{Cov}(Y, Z) \tag{11}$$

The value of $\hat{c}$ which minimizes the variance is

$$\hat{c} = -\frac{\text{Cov}(Y, Z)}{\text{Var}[Z]} \tag{12}$$

and substituting this into (11) gives

$$\text{Var}[Y + c(Z - \mu_Z)] = \text{Var}[Y] - \frac{\text{Cov}(Y, Z)^2}{\text{Var}[Z]} \tag{13}$$

In this case, Z is called a control variate, and $\hat{c}$ a control variate correction. We observe that the theoretical variance of $\{Y + c(Z - \mu_Z)\}$ is always lower than the theoretical variance of Y, with equality if there is no correlation between Y and Z since $\frac{\text{Cov}(Y, Z)^2}{\text{Var}[Z]}$ is always non-negative.

4.2 Control variates with the bivariate normal (CV-BN)

In 2017, Kang and Hooker (2017b) utilized control variates with the bivariate normal. Suppose we are given a data matrix X, and we want to estimate any linear combination of the norms or inner products of any two vectors $\mathbf{x}_i, \mathbf{x}_j$. We compute $V = XR$ as before, look at the rows $\mathbf{v}_i, \mathbf{v}_j$, and consider each tuple $\mathbf{w}_s := (v_{is}, v_{js})$.

We use CV-BN to denote control variates with the bivariate normal.

CV-BN allowed the user to estimate quadratic forms given by $Y = \mathbf{w}^T A \mathbf{w}$ where $\mathbf{w}$ is seen as a draw from the bivariate normal with A is a symmetric matrix.

We summarize and extend the results here with the aim to use these results as a foundation for COVFEFE.

To use a control variate method, we require some random variable Z which we know $\mathbb{E}[Z] = \mu_Z$. Suppose we express

$$Z_s = \mathbf{w}_s^T B \mathbf{w}_s \tag{14}$$

Table 2 Basis vectors for vector space of 2D symmetric matrices

	B_i	$\mathbf{w}_s^T B \mathbf{w}_s$	Expected value of $\mathbf{w}_s^T B \mathbf{w}_s$
B_1	$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$	v_{1s}^2	1
B_2	$\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$	v_{2s}^2	1
B_3	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	$v_{1s} v_{2s}$	$2a$

Table 3 Basis vectors for vector space of symmetric matrices used for 2D control variate

	B_i	$\mathbf{w}_s^T B \mathbf{w}_s$	Expected value of $\mathbf{w}_s^T B \mathbf{w}_s$
B_1	$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$	v_{1s}^2	1
B_2	$\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$	v_{2s}^2	1

where B is some symmetric matrix. Then Z_s can be rewritten as

$$Z_s = \mathbf{w}_s^T \begin{pmatrix} \alpha_1 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{w}_s + \mathbf{w}_s^T \begin{pmatrix} 0 & 0 \\ 0 & \alpha_2 \end{pmatrix} \mathbf{w}_s + \mathbf{w}_s^T \begin{pmatrix} 0 & \alpha_3 \\ \alpha_3 & 0 \end{pmatrix} \mathbf{w}_s \tag{15}$$

$$= \mathbf{w}_s^T \left(\sum_i \alpha_i B_i \right) \mathbf{w}_s \tag{16}$$

where B_i can be seen as basis vectors for the space of symmetric matrices, and α_i the coefficient of B in this basis. Table 2 shows the basis vectors with their corresponding expected value $\mathbf{w}_s^T B \mathbf{w}_s$.

In order to use the control variate method, we need to be able to know the true value of μ_Z. Since we normalized each row of X, we know that the each $\mathbf{x}_i$ is of unit length, and we can use the basis vectors B_1, B_2 for our control variate. However, we do not know the value of $\mathbf{w}_s^T B_3 \mathbf{w}_s$ (as this is what we want to estimate).

Hence, we let $\tilde{B}$ be the subspace of symmetric matrices which we can use for the control variate, and $\tilde{B} = \text{span}\{B_1, B_2\}$. Table 3 shows the basis vectors for this subspace.

By looking at (13), we want to maximize the value of

$$\frac{\text{Cov}\,(Y, Z)^2}{\text{Var}\,[Z]} \tag{17}$$

in order to get the most variance reduction, hence we need to compute

$$\underset{\alpha_1, \alpha_2}{\arg\max} \; \frac{\text{Cov}\left(\mathbf{w}^T A \mathbf{w}, \mathbf{w}^T \left(\alpha_1 B_1 + \alpha_2 B_2\right) \mathbf{w}\right)^2}{\text{Var}\left[\mathbf{w}^T \left(\alpha_1 B_1 + \alpha_2 B_2\right) \mathbf{w}\right]} \tag{18}$$

We state the following lemma of which the proof can be found in Muirhead (2005).

Lemma 1 *Let* $\mathbf{w} \sim N(\mathbf{0}, \Sigma)$*, and* A, B *be symmetric matrices. Then*

$$\mathbb{E}\left[\mathbf{w}^T A\mathbf{w}\right] = Tr[A\Sigma] \tag{19}$$

$$\text{Var}\left[\mathbf{w}^T A\mathbf{w}\right] = 2Tr[A\Sigma A\Sigma] \tag{20}$$

$$\text{Cov}\left(\mathbf{w}^T A\mathbf{w}, \mathbf{w}^T B\mathbf{w}\right) = 2Tr[A\Sigma B\Sigma] \tag{21}$$

In Kang and Hooker (2017b), the optimal coefficients of the basis vectors B_1, B_2 were given without any proof. It was indeed the best α_i which maximized (18), but we now rigorously prove this.

Theorem 1 *In the bivariate normal case, choosing* α_1, α_2 *in the ratio*

$$\frac{\alpha_1}{\alpha_2} = \frac{2aq_{12} + q_{22} + a^2 q_{22}}{q_{11} + a^2 q_{11} + 2aq_{12}} \tag{22}$$

yields the most variance reduction for the estimate of any quadratic form given by

$$\begin{pmatrix} v_{is} & v_{js} \end{pmatrix} \begin{pmatrix} q_{11} & q_{12} \\ q_{12} & q_{22} \end{pmatrix} \begin{pmatrix} v_{is} \\ v_{js} \end{pmatrix}$$

Proof In the general quadratic form case, let $Q = \begin{pmatrix} q_{11} & q_{12} \\ q_{12} & q_{22} \end{pmatrix}$. By applying Lemma 1, (18) simplifies to $\arg\max_{\alpha_1, \alpha_2} f(\alpha_1, \alpha_2)$ where

$$f(\alpha_1, \alpha_2) = \frac{2(\alpha_2(a^2 q_{11} + 2aq_{12} + q_{22}) + \alpha_1(a^2 q_{22} + 2aq_{12} + q_{22}))^2}{\alpha_1^2 + \alpha_2^2 + 2a^2\alpha_1\alpha_2} \tag{23}$$

Suppose α_1, α_2 are non-zero. We can express $\alpha_2 = p\alpha_1$ and thus $f(\alpha_1, \alpha_2)$ becomes

$$f(p) = \frac{2(q_{11} + a^2 pq_{11} + 2a(1+p)q_{12} + a^2 q_{22} + pq_{22})^2}{1 + 2a^2 p + p^2} \tag{24}$$

which is a function of p. Taking its first derivative and solving for the maxima (calculus omitted) yields

$$\hat{p} = \frac{2aq_{12} + q_{22} + a^2 q_{22}}{q_{11} + a^2 q_{11} + 2aq_{12}} \tag{25}$$

□

and hence we choose α_1 and α_2 such that the ratio

$$\frac{\alpha_1}{\alpha_2} = \frac{2aq_{12} + q_{22} + a^2 q_{22}}{q_{11} + a^2 q_{11} + 2aq_{12}} \tag{26}$$

Corollary 1 *In the bivariate normal case, choosing $\alpha_1 = \alpha_2 = 1$ yields the most variance reduction for the estimate of the inner product $\hat{a}$ between the vectors $\mathbf{x}_i, \mathbf{x}_j$. The variance of $\hat{a}$ after using the control variate estimate is the same as the variance reduction for Li's MLE (Li et al. 2006a) where*

$$\text{Var}\left[\hat{a}\right] = \frac{(1-a^2)^2}{1+a^2} \tag{27}$$

The optimal control variate correction $\hat{c}$ is given by

$$\hat{c} = -\frac{a}{1+a^2} \tag{28}$$

The variance of the inner product estimate $\hat{a}$ is lower than the original estimate of the inner product via ordinary random projections, which is $1 + a^2$.

Proof Set $A = \begin{pmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & 0 \end{pmatrix}$ and apply Theorem 1. □

Corollary 2 *In the bivariate normal case, choosing $\alpha_1 = \alpha_2 = 1$ yields the most variance reduction for the estimate of the squared Euclidean distance between the vectors $\mathbf{x}_i, \mathbf{x}_j$.*

The variance of the estimate $\hat{\theta}$ of the Euclidean distance after using the control variate estimate is

$$\text{Var}\left[\hat{\theta}\right] = \frac{4(a^2-1)^2}{a^2+1} \tag{29}$$

The optimal control variate correction $\hat{c}$ is given by

$$\hat{c} = -\frac{2(1-a)^2}{(1+2a^2)} \tag{30}$$

The variance of the Euclidean distance estimate is lower than the original estimate of the Euclidean distance via ordinary random projections, which is $8(a-1)^2$.

Proof Set $A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$ and apply Theorem 1. □

Corollaries 1 and 2 tell us how to get a better estimate for the inner product and Euclidean distance respectively.

Instead of computing $\hat{a} = \frac{\sum_{s=1}^{k} v_{is} v_{js}}{k}$, Corollary 1 gives

$$\hat{a} = \frac{\sum_{s=1}^{k} v_{is} v_{js}}{k} - \frac{a}{1+a^2}\left(\frac{\sum_{s=1}^{k} v_{is}^2}{k} + \frac{\sum_{s=1}^{k} v_{js}^2}{k} - 2\right) \tag{31}$$

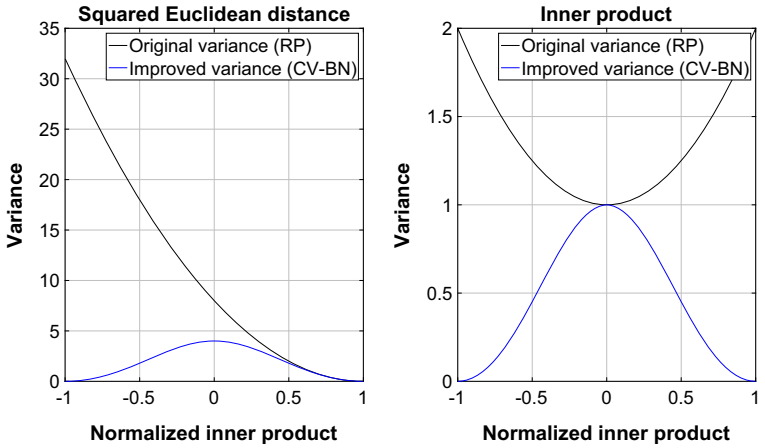

Fig. 1 Plots of the theoretical variances of estimates of squared Euclidean distance and inner product for vector pairs against the normalized inner product. We compare the theoretical variance when using ordinary random projections (RP) and using control variates (CV-BN). With normalized vectors, an inner product of 1 means the vectors are in the same direction, an inner product of 0 means the vectors are orthogonal, and an inner product of -1 means the vectors are in opposite directions

and we can rearrange the terms to get an estimate of a.

Equivalently, we can do something similar with Corollary 2 to get an estimate of the Euclidean distance, but this either necessitates knowing the value of a, or using an empirical estimate of the control variate correction from the data.

Figure 1 gives some intuition on how CV-BN performs by looking at the theoretical variance reduction achieved for the estimate of the Euclidean distance as well as the estimate of the inner product, assuming perfect knowledge of a. As Chernoff bounds for estimates are computed based on second moments, the variance can be used as a proxy to show how tight the probability bounds are. CV-BN achieves good variance reduction for the estimate of the Euclidean distance when the vector pairs get farther apart from each other. On the other hand, CV-BN does not achieve good variance reduction when the vector pairs are nearly orthogonal.

Figures 2 and 3 show the average relative RMSE of the pairwise Euclidean distance estimates as well as the pairwise inner product estimates for all vectors in the *Gisette* dataset (Lichman 2013) with CV-BN, where we have 13,500 observations in $\mathbb{R}^{5000}$.

We see that if we want to get a better estimate of the Euclidean distances, using the empirical value of the control variate correction $\hat{c}$ is better than ordinary random projections, or using a secondary estimate of the inner product. This is due to the estimate of the inner product having greater error compared to the estimate of the Euclidean distance (Vempala 2004; Kaban 2015), and the empirical value of $\hat{c}$ should be used instead for a better estimate.

On the other hand, if we wanted to estimate the inner product, there is little difference whether we use the empirical value or the theoretical value of the control variate correction (substituting an estimate of a) for the estimate of the inner product.

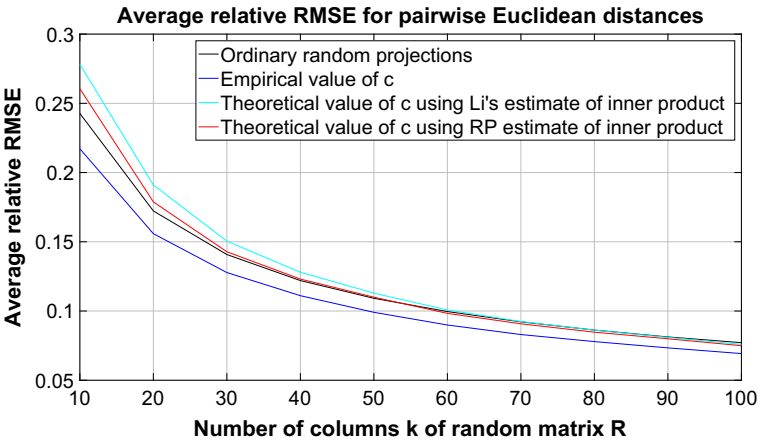

Fig. 2 Plots of average relative RMSE of 91,118,250 pairwise Euclidean distance estimates of the *Gisette* dataset, using training data, testing data, and validation data over 100 iterations. We use the ordinary random projection estimate, the control variate estimate using an empirical value of c, control variate estimate using Li's estimate of the respective inner products, and control variate estimate using the vanilla random projection estimate of the respective inner products

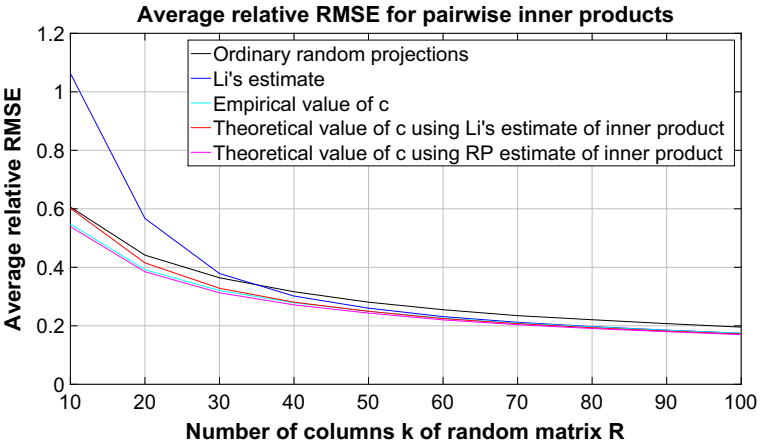

Fig. 3 Plots of average relative RMSE of 91,118,250 pairwise inner product estimates of the *Gisette* dataset, using training data, testing data, and validation data over 100 iterations. We use the ordinary random projection estimate, Li's estimate, the control variate estimate using an empirical value of c, control variate estimate using Li's estimate of the respective inner products, and control variate estimate using the vanilla random projection estimate of the respective inner products

CV-BN can also be applied to the estimation of a bivariate normal when variances are known for observational studies. In fact, control variate estimators (Papamarkou et al. 2014; Oates et al. 2017) which have higher computational costs could be used for this estimation. For example, control functionals are a recent work by Oates et al. (2017), and there is some possibility that a modification of their algorithm could be used for this purpose. A grid search for kernel parameters as well as optimal ρ could

be done to find the control functional which best estimates the (known) variances. The optimal ρ found in grid search would correspond to the correlation coefficient. The computational cost would have a greater time complexity of $O(k)$, but would be acceptable in the context of finding the correlation ρ in observational studies.

CV-BN unfortunately has three shortcomings described below:

1. The control variate correction $\hat{c}$ is in terms of a for both the estimate of the inner product and the Euclidean distance. While this made no difference to the estimate when the number of columns k of the random matrix R is sufficiently large, the value of k should be small under random projections.
2. The estimate of the inner product has a variance reduction when vector pairs are highly correlated (inner product near 1 or -1), and there is little to no variance reduction when the vector pairs are nearly orthogonal (inner product near 0), as can be seen in Fig. 1.
3. The variance in estimating the inner product $\hat{a}$ via CV-BN is equal to the variance in estimating the inner product $\hat{a}$ using Li's MLE. There seems to be no purpose in using a control variate estimate.

The authors next evolved CV-BN by looking at the multivariate normal which would rectify these shortcomings.

4.3 The multivariate normal distribution

In 2017, Kang (2017b) utilized control variates with the multivariate normal, which was called COVFEFE (COntrol Variates For Estimation via First Eigenvectors). Their theory focused on multiple control variates as in Glynn and Szechtman (2002), Portier and Segers (2018).

4.3.1 An example of control variates in three dimensions

Given the matrix $X_{n\times p}$, we compute its first singular vector $\mathbf{e}$. We assume all $\mathbf{x}_i$ are normalized. Suppose we generate the random matrix $R_{p\times k}$, with r_{ij} i.i.d., and we compute $V = XR$, $\mathbf{v}_e = \mathbf{e}^T R$, and scale our $V, \mathbf{v}_e$ by $\frac{1}{\sqrt{k}}$. Further suppose we compute and store the inner products $\langle \mathbf{x}_i, \mathbf{e}\rangle$.

Suppose we want to estimate the inner product $\langle \mathbf{x}_i, \mathbf{x}_j\rangle$, and consider the vectors

$$\mathbf{v}_1 = (v_{i1}, v_{i2}, \ldots, v_{ik}) \tag{32}$$

$$\mathbf{v}_2 = (v_{j1}, v_{j2}, \ldots, v_{jk}) \tag{33}$$

$$\mathbf{v}_e = (v_{e1}, v_{e2}, \ldots, v_{ek}) \tag{34}$$

The 3-tuple $\mathbf{w}_s := \{(v_{is}, v_{js}, v_{es})\}_{s=1}^{k}$ is drawn from the multivariate normal in three dimensions given by

$$\mathbf{w}_s = \begin{pmatrix} v_{is} \\ v_{js} \\ v_{es} \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \Sigma \right) \tag{35}$$

Table 4 Basis vectors for vector space of symmetric matrices used for 3D control variate

	B_i	$\mathbf{w}_s^T B \mathbf{w}_s$	Expected value of $\mathbf{w}_s^T B \mathbf{w}_s$
B_1	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	v_{1s}^2	1
B_2	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	v_{2s}^2	1
B_3	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	v_{3s}^2	1
B_4	$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$	$2v_{1s}v_{3s}$	$2\langle \mathbf{x}_i, \mathbf{e} \rangle$
B_5	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$	$2v_{2s}v_{3s}$	$2\langle \mathbf{x}_j, \mathbf{e} \rangle$

where

$$\Sigma = \begin{pmatrix} 1 & a & a_{13} \\ a & 1 & a_{23} \\ a_{13} & a_{23} & 1 \end{pmatrix} \tag{36}$$

All entries in Σ are known except for a which we want to estimate, with $a_{13} = \langle \mathbf{x}_i, \mathbf{e} \rangle$, and $a_{23} = \langle \mathbf{x}_j, \mathbf{e} \rangle$.

The estimate of the inner product $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$can be written as

$$\mathbf{v}_{1s}\mathbf{v}_{2s} = \mathbf{w}_s^T \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \mathbf{w}_s, \qquad 1 \leq s \leq k \tag{37}$$

and the control variate Z can similarly be written as

$$Z = \mathbf{w}^T \left(\sum_i \alpha_i B_i \right) \mathbf{w} \tag{38}$$

where B_i are the basis vectors for the subspace of symmetric matrices used for the control variate. The B_is are represented in Table 4.

Here is where we rigorously extend (Kang 2017b)'s work.

4.3.2 An example of control variates in $s + 2$ dimensions

For control variates in $s + 2$ dimensions, we can similarly consider the multivariate normal in $s + 2$ dimensions. Given the matrix $X_{n \times p}$, we compute the singular vectors

$\mathbf{e}_1, \ldots, \mathbf{e}_s$. We compute $V = XR$, and all $\mathbf{v}_{ej} = \mathbf{v}_j^T R$, where $1 \leq j \leq s$. We compute and store the $n(s+2) + \frac{s!}{2!(s-2)!}$ inner products $\langle \mathbf{x}_i, \mathbf{e}_j \rangle$, and $\langle \mathbf{e}_i, \mathbf{e}_j \rangle$.

4.3.3 Finding the optimal control variate and its correction

One of the shortcomings of CV-BN is that the optimal control variate correction $\hat{c}$ was in terms of a, which was unknown. This could not be avoided since the covariance matrix Σ was of size 2×2, and a was present in the off diagonal entries. This meant that any basis vector B_i with non-zero terms the first two rows or first two columns contributed to the term a in $\hat{c}$. With multiple extra vectors $\mathbf{e}_s$, basis vectors with non-zero terms in the first two rows and the first two columns can be omitted.

Kang (2017b) gave the coefficients α_i, the basis vectors B_i, and the optimal correction $\hat{c}$ for the multivariate normal in 3 dimensions and in 4 dimensions. We extend the results to give the coefficients α_i and the basis vectors B_i for the multivariate normal in $s+2$ dimensions, provided the vectors $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_s$ are mutually orthogonal.

Theorem 2 *Suppose we generated extra orthogonal vectors* $\mathbf{e}_1, \ldots, \mathbf{e}_s$ *and did all the preliminary computations. Consider the* $(s+2)$*-tuple* $\mathbf{w}_t := \{(v_{it}, v_{jt}, v_{e_1 t}, \ldots, v_{e_s,t})\}_{t=1}^k$. $\mathbf{w}_t$ *must be be distributed MVN* $(\mathbf{0}_{(s+2)\times 1}, \Sigma)$*, where*

$$\Sigma = \left(\begin{array}{c:c} \Sigma_{11} & \tilde{A}_{2\times s} \\ \hdashline \tilde{A}_{2\times s}^T & I_{s\times s} \end{array}\right) \tag{39}$$

with $\Sigma_{11} = \begin{pmatrix} 1 & a \\ a & 1 \end{pmatrix}$, $\tilde{A}_{2\times s} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \end{pmatrix}$ *where* $\mathbf{a}_1 = (a_{13}, a_{14}, \ldots, a_{1,s+2})$, *and* $\mathbf{a}_2 = (a_{23}, a_{24}, \ldots, a_{2,s+2})$.

Suppose we want to estimate the inner product given by

$$\mathbf{w}^T C \mathbf{w} = \mathbf{w}^T \left(\begin{array}{c:c} C_{11} & \mathbf{0}_{2\times s} \\ \hdashline \mathbf{0}_{s\times 2} & \mathbf{0}_{s\times s} \end{array}\right) \mathbf{w} \tag{40}$$

where $C_{11} = \begin{pmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & 0 \end{pmatrix}$. *We write the optimal control variate* Z *as*

$$Z = \mathbf{w}^T B \mathbf{w} = \mathbf{w}^T \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \mathbf{0}_{s\times 2} & \alpha_{s\times s} \end{array}\right) \mathbf{w} \tag{41}$$

where

$$\alpha_{s\times s} = \begin{pmatrix} \alpha_{33} & \alpha_{34} & \dots & \alpha_{3,s+2} \\ \alpha_{43} & \alpha_{44} & \dots & \alpha_{4,s+2} \\ \dots & \ddots & \ddots & \vdots \\ \alpha_{s+2,3} & \alpha_{s+2,4} & \dots & \alpha_{s+2,s+2} \end{pmatrix} \tag{42}$$

The optimal coefficients α_{ij}s are described as

$$\alpha_{ij} = \begin{cases} 0 & \text{if } i \le 2 \text{ or } j \le 2 \\ a_{1i}a_{2j} + a_{1j}a_{2i} & \text{otherwise} \end{cases} \tag{43}$$

with $\hat{c} = -\frac{1}{2}$.

Proof We prove that $\hat{c} = -\frac{1}{2}$. We write

$$\begin{aligned} B\Sigma &= \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \mathbf{0}_{s\times 2} & \alpha_{s\times s} \end{array}\right) \left(\begin{array}{c:c} \Sigma_{11} & \tilde{A}_{2\times s} \\ \hdashline \tilde{A}^T_{s\times 2} & I_{s\times s} \end{array}\right) \\ &= \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \alpha_{s\times s}\tilde{A}^T_{s\times 2} & \alpha_{s\times s} \end{array}\right) \end{aligned} \tag{44}$$

and

$$\begin{aligned} C\Sigma &= \left(\begin{array}{c:c} C_{11} & \mathbf{0}_{2\times s} \\ \hdashline \mathbf{0}_{s\times 2} & \mathbf{0}_{s\times s} \end{array}\right) \left(\begin{array}{c:c} \Sigma_{11} & \tilde{A}_{2\times s} \\ \hdashline \tilde{A}^T_{s\times 2} & I_{s\times s} \end{array}\right) \\ &= \left(\begin{array}{c:c} C_{11}\Sigma_{11} & C_{11}\tilde{A}_{2\times s} \\ \hdashline \mathbf{0}_{s\times s} & \mathbf{0}_{s\times s} \end{array}\right) \end{aligned} \tag{45}$$

We can now write

$$\begin{aligned} B\Sigma B\Sigma &= \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \alpha_{s\times s}\tilde{A}^T_{s\times 2} & \alpha_{s\times s} \end{array}\right) \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \alpha_{s\times s}\tilde{A}^T_{s\times 2} & \alpha_{s\times s} \end{array}\right) \\ &= \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \alpha_{s\times s}\alpha_{s\times s}\tilde{A}^T_{s\times 2} & \alpha_{s\times s}\alpha_{s\times s} \end{array}\right) \end{aligned} \tag{46}$$

and

$$B\Sigma C\Sigma = \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \alpha_{s\times s}\tilde{A}_{s\times 2}^T & \alpha_{s\times s} \end{array}\right)\left(\begin{array}{c:c} C_{11}\Sigma_{11} & C_{11}\tilde{A}_{2\times s} \\ \hdashline \mathbf{0}_{s\times 2} & \mathbf{0}_{s\times s} \end{array}\right)$$
$$= \left(\begin{array}{c:c} \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times s} \\ \hdashline \alpha_{s\times s}\tilde{A}_{s\times 2}^T C_{11}\Sigma_{11} & \alpha_{s\times s}\tilde{A}_{s\times 2}^T C_{11}\tilde{A}_{2\times s} \end{array}\right) \tag{47}$$

We observe that C_{11} can be seen as three elementary row operations, one which swaps the two rows of $\tilde{A}_{2\times s}$, and two which divides each row by 2. But that implies $C_{11}\tilde{A}_{s\times 2}^T\tilde{A}_{2\times s} = \frac{1}{2}\alpha_{s\times s}$, by looking at the cross terms.

Therefore $\mathrm{Tr}[\alpha_{s\times s}\tilde{A}_{s\times 2}^T C_{11}\tilde{A}_{2\times s}] = \frac{1}{2}\mathrm{Tr}[\alpha_{s\times s}\alpha_{s\times s}]$ which leads us to conclude

$$\hat{c} = \frac{-2\mathrm{Tr}[B\Sigma C\Sigma]}{2\mathrm{Tr}[B\Sigma B\Sigma]} = \frac{-2\mathrm{Tr}[C\Sigma B\Sigma]}{2\mathrm{Tr}[B\Sigma B\Sigma]}$$
$$= \frac{-\frac{1}{2}\mathrm{Tr}[\alpha_{s\times s}\alpha_{s\times s}]}{\mathrm{Tr}[\alpha_{s\times s}\alpha_{s\times s}]} = -\frac{1}{2} \tag{48}$$

□

Theorem 2 gives us a quick way to compute the control variates for large n, as well as the optimal control variate correction. We also get the following corollary.

Corollary 3 *The control variate correction in Theorem 2 gives us an improved variance of*

$$\mathrm{Var}\left[Y - \frac{1}{2}(Z - \mu_Z)\right] = \mathrm{Var}[Y] - \frac{\mathrm{Cov}(Y, Z)^2}{\mathrm{Var}[Z]}$$
$$= \mathrm{Var}[Y] - \frac{1}{2}\mathrm{Cov}(Y, Z) \tag{49}$$

Corollary 4 *Adding extra vectors will always give a variance reduction, or do no worse.*

Proof Let $Z^{(s)} = \mathbf{w}^T B\mathbf{w}$ be the case where we use s extra orthogonal vectors. Then $\mathrm{Cov}[Y, Z^{(s)}] = \frac{1}{2}\mathrm{Tr}[\alpha_{s\times s}\alpha_{s\times s}]$ follows by Theorem 2. Let

$$T^{(s)} := \mathrm{Tr}[\alpha_{s\times s}\alpha_{s\times s}] = \sum_{i=1}^{s}\sum_{j=1}^{s}\alpha_{i+2,j+2}^2 = \|\alpha_{s\times s}\|_F^2 \tag{50}$$

It follows that for $s \geq 2$ we have

$$T^{(s)} - T^{(s-1)} = 2\sum_{t=1}^{s}\alpha_{t+2,s+2}^2 - \alpha_{s+2,s+2}^2 \geq 0 \tag{51}$$

Therefore, always adding extra vectors will give us a greater variance reduction as $\mathrm{Cov}\left(Y, Z^{(s)}\right)$ increases monotonically as more vectors are added. □

Corollaries 3 and 4 show that the magnitude of $\mathrm{Cov}[Y, Z^{(s)}]$ is dependent on the magnitude of a_{ij}, and therefore we choose singular vectors $\mathbf{e}_1, \ldots, \mathbf{e}_s$ as a heuristic to maximize the variance reduction.

We note that COVFEFE does not use all additional information. In fact, if we adopted a maximum likelihood approach and compute $\hat{a}$ from the covariance matrix in (39), we could potentially get a greater variance reduction.

However, the resultant cubic equation that needs to be solved has complicated coefficients, and involves computing the determinant of the covariance matrix. For example, if we just add one extra vector and use the covariance matrix in (35) for the 3 dimensional case, the user has to solve for the root of the cubic

$$\begin{aligned} f(a) = a^3 &+ \left(\sum_{t=1}^{k}\left[a_{13}v_{jt}v_{et} - v_{it}v_{jt} + a_{23}v_{it}v_{et} + a_{13}a_{23}v_{et}^2\right] - 3a_{13}a_{23}\right)a^2 \\ &+2\left(\sum_{t=1}^{k}\left[v_{it}^2 + v_{jt}^2 + 2a_{13}a_{23}v_{it}v_{jt} - 2a_{13}v_{it}v_{et} - 2a_{23}v_{jt}v_{et}\right.\right. \\ &\left.\left. -a_{23}^2\left(v_{it}^2 + v_{et}^2\right) + a_{13}^2\left(2a_{23}^2 - v_{jt}^2 + v_{et}^2\right)\right] + a_{13}^2 + a_{23}^2 - 1\right)a \\ &+\left(\sum_{t=1}^{k}\left[(-1 + a_{23}^2)\left(v_{it}(v_{jt} - a_{23}v_{et})\right) + a_{13}^2 v_{it}(v_{jt} - 2a_{23}^2 v_{jt} + a_{23}v_{et})\right.\right. \\ &+a_{13}^3 v_{jt}(a_{23}v_{jt} - v_{et}) + a_{13}(a_{23}^3 v_{it}^2 + v_{jt}v_{et} \\ &\left.\left. +a_{23}^2 v_{jt}v_{et} - a_{23}(v_{it}^2 + v_{jt}^2 + v_{et}^2))\right] - a_{13}^2 a_{23} + a_{13}a_{23}^2 - a_{13}a_{23}\right) \end{aligned} \tag{52}$$

which can get a bit involved. On the other hand, with the multivariate control variate method, Theorem 2 allows the user to compute

$$\hat{a} = \frac{\sum_{t=1}^{k} v_{it}v_{jt}}{k} - \frac{1}{2}\left(\frac{2a_{13}a_{23}\sum_{t=1}^{k} v_{et}^2}{k} - 2a_{13}a_{23}\right) \tag{53}$$

for the 3-dimensional case, and compute

$$\begin{aligned} \hat{a} = &\frac{\sum_{t=1}^{k} v_{1t}v_{2t}}{k} - \sum_{r=3}^{s+2} a_{1r}a_{2r}\left(\frac{\sum_{t=1}^{k} v_{e_{r-2}t}^2}{k} - 1\right) \\ &- \sum_{g,h\geq 3, g>h}^{s+2}\left(\frac{(a_{1g}a_{2h} + a_{1h}a_{2g})\sum_{t=1}^{k} v_{e_{g-2}t}v_{e_{h-2}t}}{k}\right) \end{aligned} \tag{54}$$

for the general $s+2$-dimensional case.

The latter computations are cleaner to implement in a programming language.

4.4 Storage and computational complexity of the algorithm

We necessarily need to compute the first s singular vectors, which can be costly, with computational complexity of $O(\min(np^2, n^2p))$. If the matrix X is sparse, then algorithms like Lanczos can be used to get the first few singular vectors. Probabilistic algorithms can also be used (Halko et al. 2011) to estimate the first few singular vectors with faster computational complexity of $O(n^2 s)$, where s is a parameter chosen based on X.

We also need to compute and store all $\langle \mathbf{x}_i, \mathbf{e}_t \rangle$, for $1 \leq i \leq n$, and $1 \leq t \leq s$. This has computational complexity of $O(snp + s^2 p)$, but can be thought of to be $O(np)$ if the number of extra vectors $\mathbf{e}_1, \ldots, \mathbf{e}_s$ is small. The computational complexity of $O(np)$ in this context is still acceptable since pre-processing data (such as scaling, normalizing, or centering) also costs $O(np)$.

We also need to compute the control variate. This requires the values of each $v_{e_g s} v_{e_h s}, v_{e_g}^2, v_{e_h}^2$ for $1 \leq g, h \leq s$, which has computational complexity $O(s^2 k + sk)$. However, this is a once off computation since the same values are used for all pairs of vectors $\mathbf{v}_i, \mathbf{v}_j$.

If we discount the pre-processing time of generating the first s singular vectors, the computational complexity taken for COVFEFE is of order $O(snp + s^2 p + npk + n^2 k + s^2 k + sk) = O(npk + n^2 k)$ when $s \ll k$. This keeps to the same time as ordinary random projections.

5 Random projections with Bayesian inference

Fosdick and Raftery (2012) motivated this work when they looked at estimating the correlation coefficient of the bivariate normal using the uniform prior, Jeffrey's prior (1961), and the arc-sine prior (Jeffreys 1961), amongst other priors. Their estimate of the correlation coefficient was given by

$$\hat{a} = \frac{\int_{-1}^{1} a\ p(a)\ l(a)\ da}{\int_{-1}^{1} p(a)\ l(a)\ da} \tag{55}$$

where $p(a)$ is the prior chosen, $l(a)$ is the log-likelihood, and the denominator the normalizing constant.

Fosdick and Raftery simulated a million experiments with varying sample sizes and fixed correlation value in order to determine which priors performed well. As expected, their results showed that different priors performed well for correlation coefficients with varying magnitudes.

In the case of estimating the correlation coefficient for one observational study, there is usually no indication of what region the magnitude lies in. The arc-sine prior and the uniform prior were preferred as they performed well for all magnitudes.

On the other hand, there can be improvement from using an arc-sine prior or a uniform prior in the estimation of $\frac{n(n-1)}{2}$ inner products via random projections.

We now discuss the statistical aspects of Kang (2017a) in random projections.

5.1 The Bayesian prior

Given a data matrix $X_{n\times p}$, we can imagine the n observations as being random draws from some arbitrarily large population. We can also imagine the $\frac{n(n-1)}{2}$ pairwise inner products of X as random variables from some continuous probability distribution $q(a)$.

While we do not know this probability distribution $q(a)$, we can get a close enough estimate to this distribution by sampling pairwise inner products $a_1, a_2, \ldots, a_s$ uniformly from the matrix $X_{n\times p}$.

We assume that the $\frac{n(n-1)}{2}$ pairwise inner products in $X_{n\times p}$ are sufficiently large enough to characterize the entire distribution, which is not an unreasonable assumption to make as n is generally large. The sampling distribution of $a_1, \ldots, a_s$ can be thought of as the prior $p(a)$.

5.1.1 Kernel density estimates and numerical integration

We do not necessarily need a closed-form expression for the prior $p(a)$ in order to compute the estimate $\hat{a}$ in (55).

Suppose we let $f(a) = p(a)l(a)$, where $l(a)$ is of the form (6). Then the goal is to numerically integrate the functions $\int_{-1}^{1} a\ f(a)\ da$ and $\int_{-1}^{1} f(a)\ da$ to estimate the inner product as in (55).

Numerical integration algorithms require evaluating $f(a)$ at points $a_1, a_2, \ldots, a_s$, and this requires computing values of $f(a_1), \ldots, f(a_s)$. But if we had the values of $f(a_1), \ldots, f(a_s)$, then we do not need the actual function $f(a)$ itself.

We can use kernel density estimators like the Nadaraya Watson estimator (Nadaraya 1964; Watson 1964) to evaluate $p(a)$ at equally spaced points over the interval $[-1, 1]$. The value of $p(a)$ over these equally spaced points can be passed into a numerical integration algorithm.

We give an example using Simpson's Rule, and consider the interval $[0, 1]$ with equally spaced points at $a_0 := 0, a_1, a_2, \ldots, a_{2s} := 1$, assuming we always have non-negative inner products.

We let α denote the length of the interval $[a_i, a_{i+1}]$, and evaluate $p(a)$ at the points $a_0, a_1, \ldots, a_{2s}$. We hence compute the numerator $\int_0^1 a\ p(a)\ L(a)\ da$ as

$$\int_0^1 a\ p(a)\ L(a)\ da = \frac{2\alpha}{6} \sum_{t=1}^{s} \left[f(a_{2t-2}) + 4f(a_{2t-1}) + f(a_{2t}) \right] \tag{56}$$

and do likewise for the denominator $\int_0^1 p(a)\ L(a)\ da$.

It is straightforward to show that the expected variance of the inner product estimate is smaller than the original variance of the inner product estimate.

However, this does not necessarily mean that the actual variance of the estimate is smaller than the original variance of the inner product estimate.

Unlike COVFEFE where we had to store extra vectors and compute the control variates, we have to sample pairwise inner products $a_1, \ldots, a_s$ and use a kernel density estimator to get a prior $p(a)$. Moreover, after getting the prior $p(a)$, we have to compute the quotient of two integrals

$$\frac{\int_0^1 a\ p(a)\ L(a)\ da}{\int_0^1 p(a)\ L(a)\ da} \tag{57}$$

with numerical integration tools.

We first look at the time taken to compute $p(a)$ at s equally spaced points. This is dependent on the number of actual (pairwise) inner products sampled. Suppose we sample up to $\left\lfloor \sqrt{\frac{n(n-1)}{2}} \right\rfloor$ inner products. This takes time of order $O(np)$. Moreover, evaluating $p(a)$ at $2s$ equally spaced points with Nadaraya-Watson takes time of order $O(ns)$. Altogether, this takes an additional $O(n(p+s))$ of time.

5.2 Computational complexity of the algorithm

We now consider the computational complexity to evaluate (56). For each inner product pair, we have a computational complexity of $O(k)$ to compute the constant terms in (56). We then further take $O(s)$ time to compute the integral for the estimate in (56). The total computational complexity taken for all pairwise estimates is therefore $O(n^2(k+s))$.

Hence the overall computational complexity of this algorithm is $O(npk + np + ns + n^2(k+s)) = O(npk + n^2(k+s))$, which is slower than COVFEFE or ordinary random projections.

However, we show that in certain circumstances, using a Bayesian prior to estimate the inner product gives more accuracy, and this may be an acceptable tradeoff.

6 Experiments

We perform two experiments in this paper. The first experiment is to verify that the algorithms mentioned are better than ordinary random projections and other baseline methods. In these experiments, we compute the average RMSE of all pairwise inner product estimates on the *Arcene* dataset, *Gisette* dataset, and the *MNIST* test images dataset. We look at the average RMSE of all pairwise inner product estimates as we want to demonstrate that the estimators works well on all pairs of "good" and "bad" vectors. We look at both centered and uncentered versions of the datasets, denoting C as centered, and U as uncentered for the plots.

The second experiment measures the precision and the recall in identifying vector pairs that are almost orthogonal from the centered datasets. We look at the average precision and average recall for vector pairs that have an (absolute value of the) inner

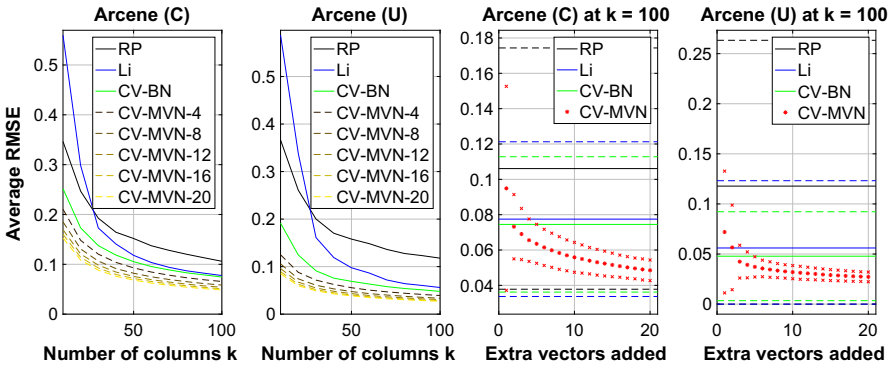

Fig. 4 Plots of average RMSE of all 404,550 pairwise inner product estimates for the *Arcene* dataset, using the multivariate control variate estimates over 100 iterations

product less than $\{0.01, 0.02, \ldots, 0.20\}$, together with their standard deviations. We want to show that these algorithms are an improvement over CV-BN and Li's method, as they do not provide good variance reduction when the vectors are almost orthogonal, as in Fig. 1.

For each experiment, we normalized each observation to have norm 1. We use a random projection matrix $R_{p\times k}$, where r_{ij} are i.i.d. $N(0, 1)$, and k ranges from $10, 20, \ldots, 100$. We repeat the experiments for 100 simulations on all three datasets.

The Matlab code is attached in the supplementary material. The results in this paper come from experiments which have been initialized from a random seed of 0 for comparison purposes.

6.1 Control variates and the multivariate normal

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-BN to be the bivariate control variate estimate of the inner product. We denote CV-MVN-s to be the COVFEFE estimate with extra vectors $\mathbf{e}_1, \ldots, \mathbf{e}_s$ of the inner product.

The first two plots in Figs. 4, 5, and 6 compares the average RMSE of the baseline algorithms denoted as solid lines (RP, Li, CV-BN) with the average RMSE of CV-MVN denoted as dashed lines, where we add extra vectors in multiples of 4 up to twenty extra vectors. We use a log scale for the y axis with the *Gisette* dataset.

The last two plots in Figs. 4, 5, and 6 show the average RMSE (solid horizontal lines, red asterixes) with 3 standard deviations (dotted horizontal lines, red crosses) for all estimators, when the number of columns k is fixed at 100. The x axis of these plots signify the number of extra vectors $\mathbf{e}_i$ added.

For the *Arcene* dataset, it takes up to three extra vectors $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ to get a comparable average RMSE with Li's estimate and the BN estimate for both centered and uncentered data, yet achieve a significantly smaller error. Adding further vectors $\mathbf{e}_4, \mathbf{e}_5, \ldots, \mathbf{e}_{20}$ only give a marginal improvement for the average RMSE.

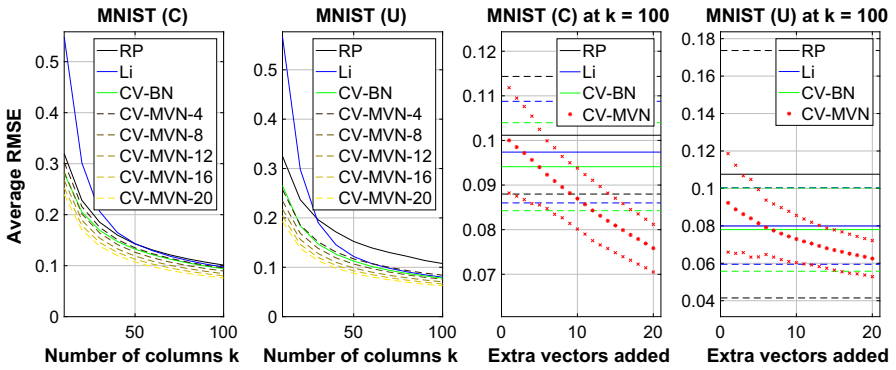

Fig. 5 Plots of average RMSE of all 49,995,000 pairwise inner product estimates for the *MNIST* test dataset, using the multivariate control variate estimates over 100 iterations

For the *MNIST* test dataset, it can take up to seven extra vectors $\mathbf{e}_1, \ldots, \mathbf{e}_7$ before achieving an average RMSE comparable to the BN estimate for the uncentered data, but only five extra vectors for centered data. However, adding further vectors causes the average RMSE to decrease further.

The *Gisette* dataset can be a "bad" dataset for random projections, as the extent of improvement in the estimation of inner products is negligible compared to the *Arcene* and *MNIST* datasets. However, adding extra vectors $\mathbf{e}$ does result in a gradual decrease in the average RMSE.

The empirical results also show that for centered data, we need fewer eigenvectors to achieve a lower average RMSE than the next best estimate (CV-BN or Li's MLE) compared to uncentered data. On the other hand, the average RMSE for all the estimators of centered data is higher than the corresponding average RMSE for all the estimators of centered data.

We further note that while CV-BN and Li's MLE eventually have the same average performance as k increases, the performance of Li's MLE is actually worse on average than the ordinary random projection estimate when k is extremely small (as denoted by the blue lines in Figs. 4, 5, and 6. CV-BN (as well as CV-MVN) on the other hand consistently performs better on average than the ordinary random projection estimate for all values of k. We hypothesize that this is due to the fact that maximum likelihood estimators are only asymptotically unbiased, hence with extremely small k, the estimator can have large errors.

We make the case (empirically) that while CV-BN has the same variance as Li's MLE, CV-BN is easier to implement as there is no need for numerical root finding algorithms, and can improve the ordinary random projection performance with extremely small values of k.

We now describe the plots for the second experiment with COVFEFE. The x-axis of Figs. 7, 8, and 9 correspond to the proportion of inner products which are less than s, where we have $s \in \{0.01, \ldots, 0.20\}$.

We show the plots of the average precision and recall with 3 standard deviations when we use $k = 10, 50, 100$ columns of the random projection matrix. Similar to the

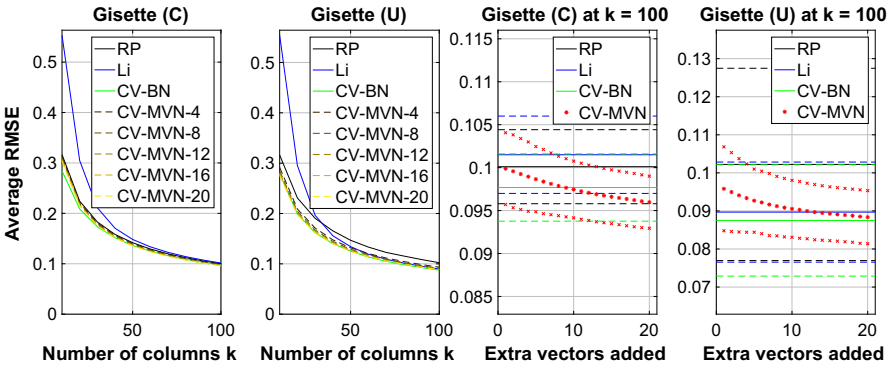

Fig. 6 Plots of average RMSE of all 91,118,250 pairwise inner product estimates for the *Gisette* dataset, using the multivariate control variate estimates over 100 iterations

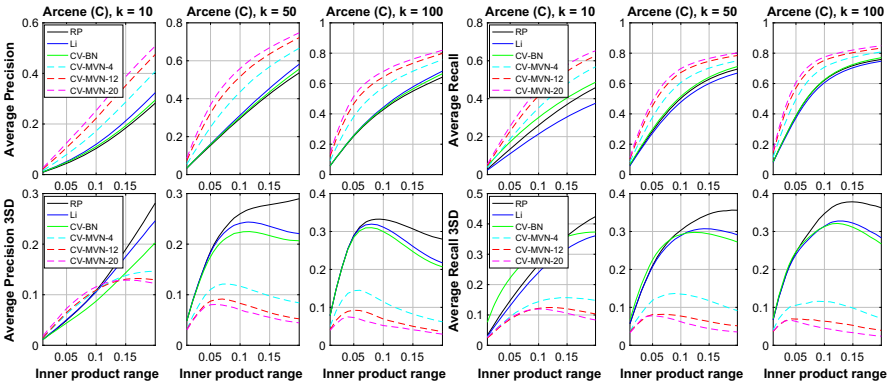

Fig. 7 Plots of average precision, recall with 3 standard deviations of retrieval of inner products less than s, for $s \in \{0.01, 0.02, \ldots, 0.20\}$ of all 404,550 pairwise inner products for the *Arcene* dataset, using the multivariate control variate estimates over 100 iterations

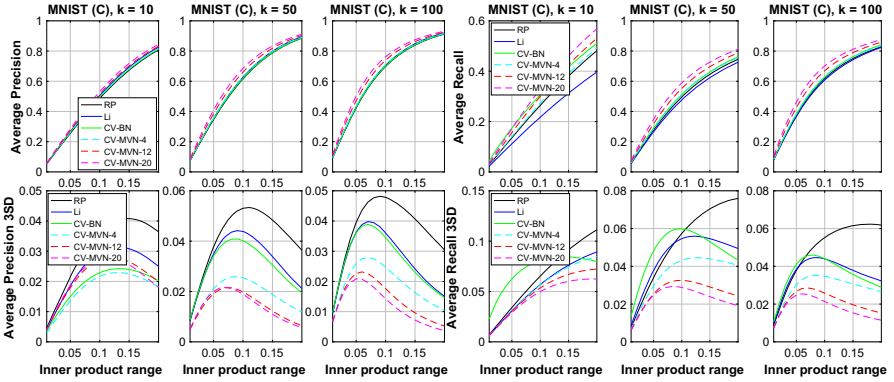

Fig. 8 Plots of average precision, recall with 3 standard deviations of retrieval of inner products less than s, for $s \in \{0.01, 0.02, \ldots, 0.20\}$ of all 49,995,000 pairwise inner products for the *MNIST* test dataset, using the multivariate control variate estimates over 100 iterations

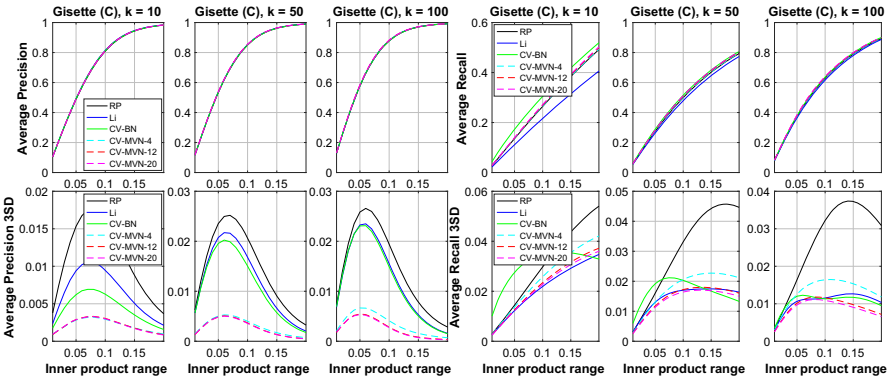

Fig. 9 Plots of average precision, recall with 3 standard deviations of retrieval of inner products less than s, for $s \in \{0.01, 0.02, \ldots, 0.20\}$ of all 91,118,250 pairwise inner products for the *Gisette* dataset, using the multivariate control variate estimates over 100 iterations

first experiment, the baseline algorithms are denoted by solid lines, and CV-MVN as dashed lines. All measurements here are done with the respective centered datasets.

We note that out of the baseline algorithms, CV-BN has lower standard errors for precision in retrieving inner products in most of the cases where $s \in \{0.01, 0.02, \ldots, 0.20\}$. However, CV-BN has higher standard errors for the recall in retrieving inner products.

CV-MVN performs quite well in general with the addition of extra vectors, having greater precision and recall than the baseline algorithms as well as lower standard errors. For example, CV-MVN-4 has much lower standard errors for the precision and recall for the *Arcene* and *MNIST* dataset at $k = 50, 100$, with comparable standard errors at $k = 10$.

6.2 Bayesian prior

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-BN to be the bivariate control variate estimate of the inner product.

Let $\text{rnd}_2(x)$ denote x rounded up to the nearest power of 2, i.e. $\text{rnd}_2(x) \equiv 2^{\lceil \log_2(x) \rceil}$. Let $N \equiv n(n+1)/2$, the total number of pairwise inner products from the respective datasets.

We denote BP-s to be the Bayesian prior estimate where we sample s inner products, BP-LOG to be the Bayesian prior estimate where we sample $\text{rnd}_2(\log_2 N)$ inner products, and BP-SQRT to be the Bayesian prior estimate where we sample $\text{rnd}_2(\sqrt{N})$ inner products.

The first two plots in Figs. 10, 11, and 12 compares the average RMSE of the baseline algorithms (RP, Li, CV-BN) with the average RMSE when we sample up to $\text{rnd}_2(\sqrt{N})$ inner products to estimate the prior.

The last two plots in Figs. 10, 11, and 12 show the average RMSE with 3 standard deviations for all estimators, when the number of columns k is fixed at 100. The x axis

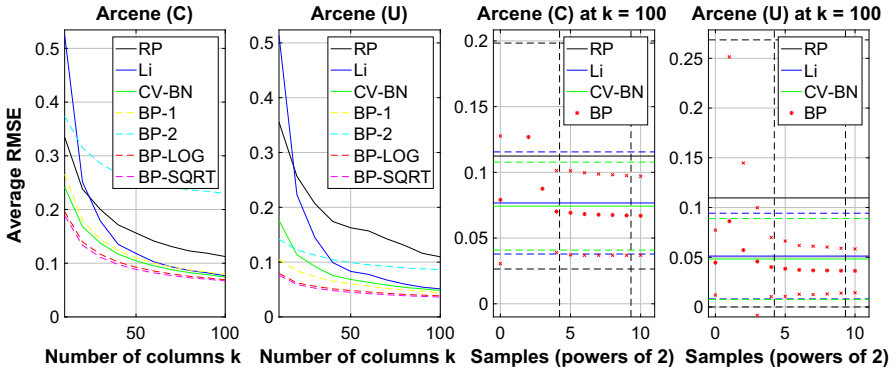

Fig. 10 Plots of average RMSE of all 404,550 pairwise inner product estimates for the *Arcene* dataset, using our Bayesian prior estimates over 100 iterations

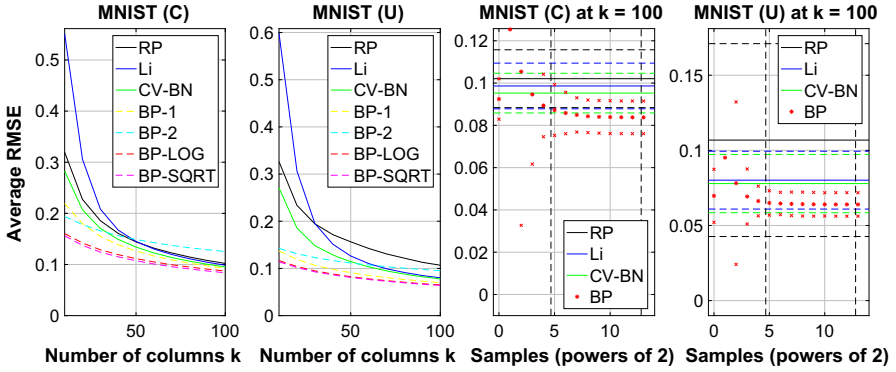

Fig. 11 Plots of average RMSE of all 49,995,000 pairwise inner product estimates for the *Arcene* test dataset, using our Bayesian prior estimates over 100 iterations

of the plots signify the number of inner products sampled, as a power of 2, and can be thought of as a log scale. The vertical dotted lines show the actual values (in terms of the powers of 2) of $\log(N)$ and $\sqrt{N}$.

We focus on $\log(N)$ and $\sqrt{N}$ as the number of inner products to be sampled to get a good estimate of the distribution of the inner products.

For both uncentered and centered data, we see that for all three datasets, the average RMSE decreases when more inner products are sampled. However, the Bayesian prior algorithm does better for the *MNIST* and *Gisette* datasets compared to the *Arcene* dataset, by substantially reducing the overall RMSE.

We can also see that sampling about $\log(N)$ inner products already achieves a low RMSE, and sampling about $\sqrt{N}$ inner products does not yield a substantially better RMSE.

We now describe the plots for the second experiment with the Bayesian prior. The x-axis of Figs. 13, 14, and 15 correspond to the proportion of inner products which are less than s, where we have $s \in \{0.01, \ldots, 0.20\}$.

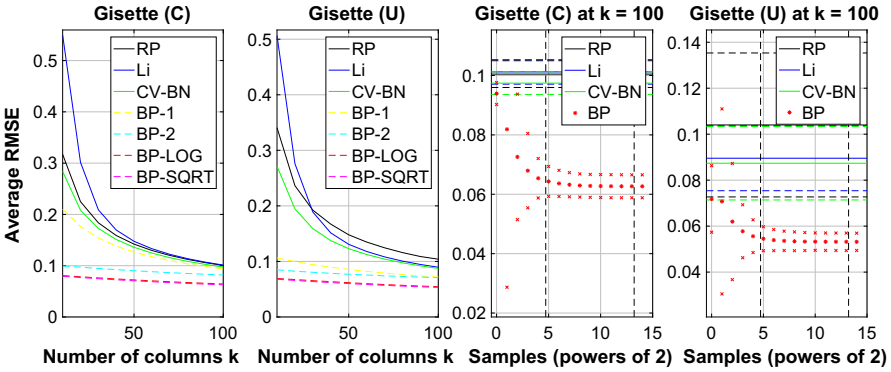

Fig. 12 Plots of average RMSE of all 91,118,250 pairwise inner product estimates for the *Arcene* dataset, using our Bayesian prior estimates over 100 iterations

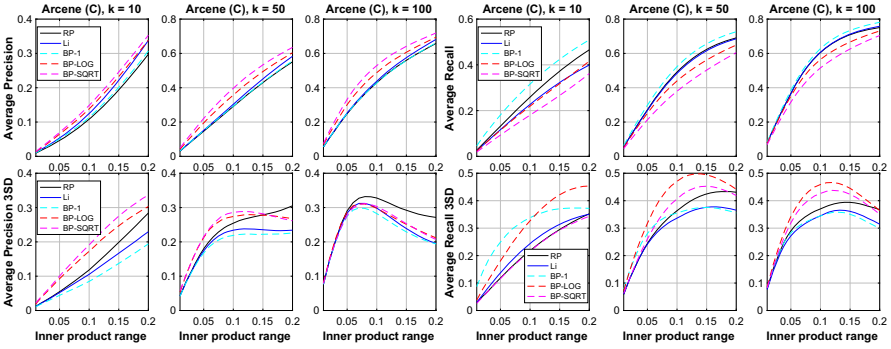

Fig. 13 Plots of average precision, recall with 3 standard deviations of retrieval of inner products less than s, for $s \in \{0.01, 0.02, \ldots, 0.20\}$ of all 404,550 pairwise inner products for the *Arcene* dataset, using the Bayesian prior estimates over 100 iterations

We show the plots of the average precision and recall with 3 standard deviations when we use $k = 10, 50, 100$ columns of the random projection matrix. Similar to the first experiment, the baseline algorithms are denoted by solid lines, and CV-BP as dashed lines. All measurements here are done with the respective centered datasets.

Here, the Bayesian prior algorithm has a higher precision on the *Arcene* dataset, but lower precision on the other two datasets, when compared with baseline algorithms. However, the Bayesian prior achieves a higher recall on the *Gisette* and *MNIST* datasets when compared to other baseline algorithms.

7 Discussion

To understand the performance of COVFEFE and the Bayesian prior on the above datasets, we should look at the singular vectors of the data matrix as well as the distribution of inner products of these data.

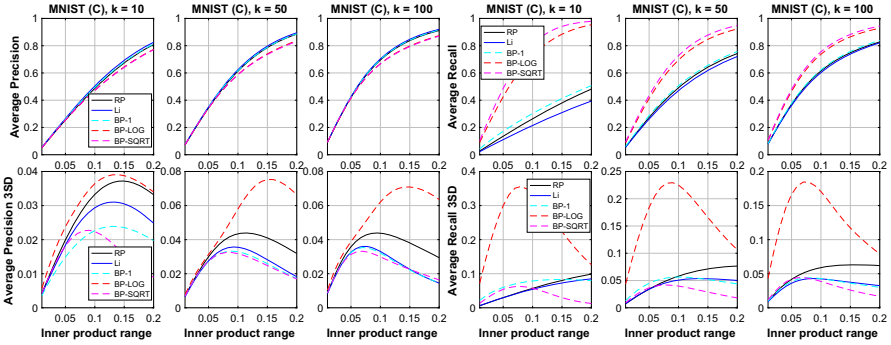

Fig. 14 Plots of average precision, recall with 3 standard deviations of retrieval of inner products less than s, for $s \in \{0.01, 0.02, \ldots, 0.20\}$ of all 49,995,000 pairwise inner products for the *MNIST* test dataset, using the Bayesian prior estimates over 100 iterations

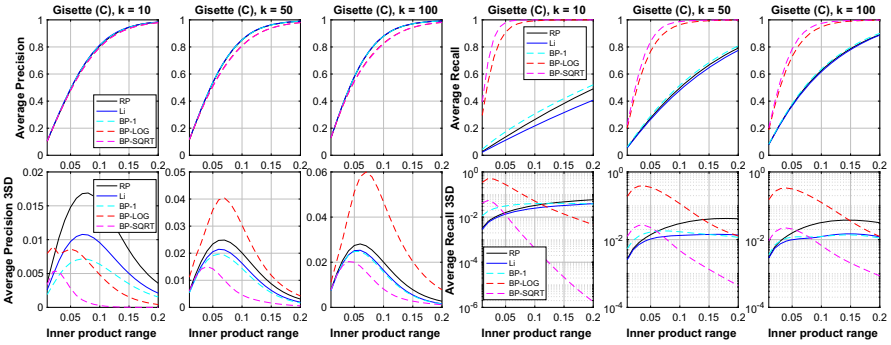

Fig. 15 Plots of average precision, recall with 3 standard deviations of retrieval of inner products less than s, for $s \in \{0.01, 0.02, \ldots, 0.20\}$ of all 91,118,250 pairwise inner products for the *Gisette* dataset, using the Bayesian prior estimates over 100 iterations

Figure 16 shows the cumulative proportion of variation in the data explained by the singular vectors of the respective three datasets. In all of these datasets, we see that the singular values of uncentered data explain more of the variation as compared to centered data. Moreover, for the *Arcene* and *MNIST* dataset, the proportion of variation explained by the first few singular vectors are extremely high, compared to the *Gisette* dataset.

Therefore, the performance of COVFEFE continually improves in Figs. 4 and 5 as more of the singular vectors are added as extra information. On the other hand, we see that the performance of COVFEFE in Fig. 6 has negligible performance as more singular vectors are added, since the first few singular vectors of the *Gisette* dataset does not account for much of the variation.

To summarize, if a few singular vectors account for most of the proportion of variation in the data, COVFEFE works well in reducing the RMSE of the inner product estimates. In fact, we are actually incorporating the information provided by the singular vectors in the data with respect to the distance. Mathematically, we can think of

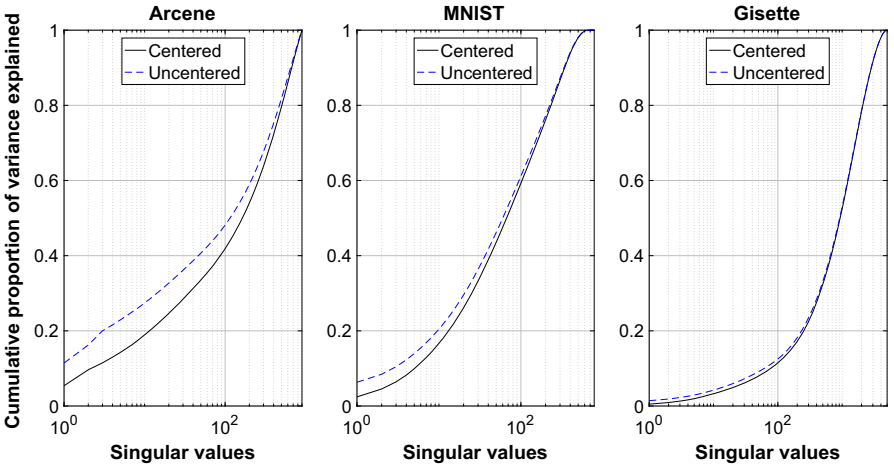

Fig. 16 Cumulative proportion of variation explained by singular values of the *Arcene* dataset, *MNIST* test dataset, and *Gisette* dataset. We use a log scale for the x axis

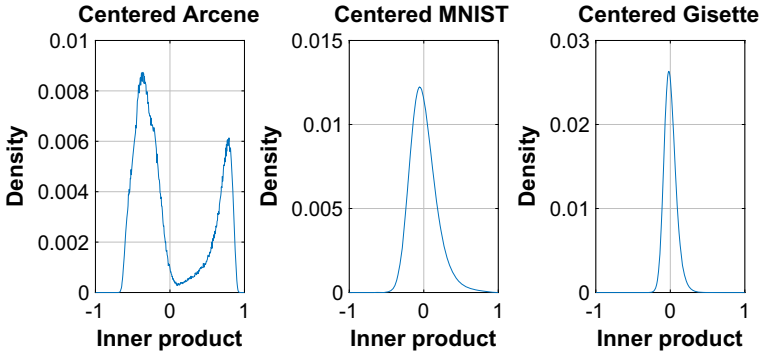

Fig. 17 Distribution of pairwise inner products from the centered *Arcene* dataset, centered *MNIST* test dataset, and centered *Gisette* dataset

the improvement as storing the first few coordinates of a "better" basis to represent the data.

In fact, COVFEFE unites the strengths of random projections (preserving distances in expectation) and PCA (finding a different coordinate system where variation is maximal along these axes), which to the best of our knowledge is unpresidented.

We now consider the distribution of inner products for these datasets. Figure 17 shows the distribution of inner products for the centered and uncentered datasets.

Recall that the ordinary estimate of the inner product between pairwise vectors under random projections has a higher variance (see Fig. 1) when the vectors are nearly orthogonal to each other. We can see that for centered *Gisette* and centered *MNIST* data, the distribution of inner products is almost symmetrical about zero, with the *Gisette* data having a very narrow peak. Hence the Bayesian prior algorithm per-

forms better in terms of recall, since observations far from zero (due to high variance) would be pulled towards the center.

On the other hand, if the distribution of the inner products is not "nice" (for example, bimodal distribution like the *Arcene* dataset), then the Bayesian prior algorithm can perform badly compared to baseline algorithms.

To summarize, the Bayesian prior algorithm has a better performance if our distribution of inner products is near to zero.

8 Conclusion

As high dimensional data becomes more ubiquitous, common learning algorithms perform badly due to the curse of dimensionality. Random projections can be used to mitigate this, by projecting data down to a lower dimensional space, making it more tractable for analysis.

On the other hand, there will always be small sample estimation when more data is expensive or infeasible to get, e.g. getting data in medical trials.

Given that control variates, numerical integration, bayesian inference, kernel density estimators are common in the modern statistician's toolbox, there is no reason why they should not be applied to random projections or small sample estimation to improve their estimates.

These algorithms can also further be improved on. For example, while the method of control variates (CV-BN, COVFEFE) is only useful with random projections, we believe that zero variance control variate techniques or control functionals may work well in estimating the correlation coefficient of the bivariate normal in such observational studies or reducing the generalization error in classification algorithms when used with random projections.

The time taken for the Bayesian prior algorithm can also be reduced if we choose a model for the distribution of inner products, rather than constructing the distribution via kernel density estimators.

We have shown examples of how these tools could be used to improve estimates, and we hope that our paper would lead to increased cross-disciplinary work and collaboration between computer scientists and statisticians.

Acknowledgements We would like to thank the reviewers for their comments and suggestions for improvement, which has helped to enhance the quality of the paper. We also want to thank the following people: Wong Wei Pin and Sergey Kushnarev for fruitful and productive discussions. We thank Omar Ortiz for his technical assistance.

References

Achlioptas D (2003) Database-friendly random projections: Johnson–Lindenstrauss with binary coins. J Comput Syst Sci 66(4):671–687

Ailon N, Chazelle B (2009) The fast Johnson–Lindenstrauss Transform and approximate nearest neighbors. SIAM J Comput 39(1):302–322

Alkema L, Raftery A, Gerland P, Clark S, Pelletier F, Buettner T, Heilig G (2011) Probabilistic projections of the total fertility rate for all countries. Demography 48(3):815–839

Cai D, He X, Han J (2005) Document clustering using locality preserving indexing. IEEE Trans Knowl Data Eng 17(12):1624–1637
Casella G, Berger R (2001) Statistical inference. Duxbury Resource Center
Charikar MS (2002) Similarity estimation techniques from rounding algorithms. In: Proceedings of the thiry-fourth annual ACM symposium on theory of computing. ACM, pp 380–388
Dasgupta S (2000) Experiments with Random Projection. In: Proceedings of the 16th conference on uncertainty in artificial intelligence, UAI '00, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc, pp 143–151
Durrant R, Kaban A (2013) Random projections as regularizers: learning a linear discriminant ensemble from fewer observations than dimensions. In: Asian conference on machine learning, pp 17–32
Fosdick BK, Perlman MD (2016) Variance-stabilizing and confidence-stabilizing transformations for the normal correlation coefficient with known variances. Commun Stat Simul Comput 45(6):1918–1935
Fosdick BK, Raftery AE (2012) Estimating the correlation in bivariate normal data with known variances and small sample sizes. Am Stat 66(1):34–41
Fu Y, Wang H, Wong A (2013) Small sample inference for the correlation in bivariate normal with known variances. Far East J Theor Stat 45(2):147
Glynn PW, Szechtman R (2002) Some new perspectives on the method of control variates. In: Monte Carlo and Quasi-Monte Carlo Methods 2000. Springer, pp 27–49
Halko N, Martinsson PG, Tropp JA (2011) Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev 53(2):217–288
Indyk P, Motwani R (1998) Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the thirtieth annual ACM symposium on theory of computing, STOC '98, New York, NY, USA. ACM, pp 604–613
Jeffreys H (1961) Theory of probability, 3rd edn. Oxford
Kaban A (2015) Improved bounds on the dot product under random projection and random sign projection. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 487–496
Kang K (2017a) Random projections with Bayesian priors. In: Natural Language Processing and Chinese Computing - 6th CCF International Conference, NLPCC 2017, Dalian, China, November 8-12, 2017, Proceedings, pp 170–182
Kang K (2017b) Using the multivariate normal to improve random projections. In: Intelligent data engineering and automated learning—IDEAL 2017: 18th international conference, Guilin, China, October 30–November 1, 2017, Proceedings. Springer, Cham, pp 397–405
Kang K, Hooker G (2017a) Control variates as a variance reduction technique for random projections. In: Pattern recognition applications and methods - 6th international conference, ICPRAM 2017, Porto, Portugal, February 24-26, 2017, Revised Selected Papers, pp 1–20
Kang K, Hooker G (2017b) Random projections with control variates. In: Proceedings of the 6th international conference on pattern recognition applications and methods - volume 1: ICPRAM. INSTICC, ScitePress, pp 138–147
Lavenberg SS, Welch PD (1981) A perspective on the use of control variables to increase the efficiency of Monte Carlo simulations. Manage Sci 27(3):322–335
Li P, Hastie T, Church KW (2006a) Improving random projections using marginal information. In: Lugosi G, Simon H-U (eds) COLT, volume 4005 of Lecture Notes in Computer Science. Springer, pp 635–649
Li P, Hastie TJ, Church KW (2006b) Very Sparse Random Projections. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '06, New York, NY, USA. ACM, pp 287–296
Li P, Mahoney MW, She Y (2010) Approximating higher-order distances using random projections. In: Proceedings of the twenty-sixth conference on uncertainty in artificial intelligence. AUAI Press, pp 312–321
Liberty E, Ailon N, Singer A (2008) Dense fast random projections and lean walsh transforms. In: Goel A, Jansen K, Rolim JDP, Rubinfeld R (eds) APPROX-RANDOM, volume 5171 of Lecture Notes in Computer Science. Springer, pp 512–522
Lichman M (2013) UCI machine learning repository
Madansky A (1965) On the maximum likelihood estimate of the correlation coefficient. Defense Technical Information Center
Mardia KV, Kent JT, Bibby JM (1979) Multivariate analysis. Academic Press, London
Muirhead RJ (2005) Aspects of multivariate statistical theory. Wiley-Interscience, Hoboken

Nadaraya EA (1964) On estimating regression. Theory Probab Appl 9(1):141–142
Oates CJ, Girolami M, Chopin N (2017) Control functionals for Monte Carlo integration. J R Stat Soc: Ser B (Stat Methodol) 79(3):695–718
Papamarkou T, Mira A, Girolami M (2014) Zero variance differential geometric Markov chain Monte Carlo algorithms. Bayesian Anal 9(1):97–128
Paul S, Boutsidis C, Magdon-Ismail M, Drineas P (2013) Random projections for support vector machines. In: Artificial intelligence and statistics, pp 498–506
Portier F, Segers J (2018) Monte carlo integration with a growing number of control variates. arXiv preprint arXiv:1801.01797
Shao J (2003) Mathematical statistics. Springer Texts in Statistics. Springer
Vempala SS (2004) The random projection method, volume 65 of DIMACS series in discrete mathematics and theoretical computer science. Providence, R.I. American Mathematical Society. Appendice, pp 101–105
Watson GS (1964) Smooth regression analysis. Sankhyā: Indian J Stat Ser A 359–372