Improving Sign Random Projections With Additional Information

Keegan Kang [*] [1] Wong Wei Pin [*] [1]

Abstract

Sign random projections (SRP) is a technique which allows the user to quickly estimate the angular similarity and inner products between data. We propose using additional information to improve these estimates which is easy to implement and cost efficient. We prove that the variance of our estimator is lower than the variance of SRP. Our proposed method can also be used together with other modifications of SRP, such as Super-Bit LSH (SBLSH). We demonstrate the effectiveness of our method on the MNIST test dataset and the Gisette dataset. We discuss how our proposed method can be extended to random projections or even other hashing algorithms.

1. Introduction

One of the first appearances of SRP was in graph theory (Goemans & Williamson, 1995) where the authors looked at the probability of a random hyperplane separating two vectors. This probability was used to come up with an improved algorithm to approximate the maximum cut in a graph. Today, this probability is also used for distance based learning algorithms given high dimensional data.

Consider a data matrix $X_{n\times p}$ with n observations, and p features. Computing pairwise distances between each observation takes at least $O(n^2p)$ of time which is computationally costly when n, p are large.

It was a few years later when the fact that the probability of the separation was proportional to the angle θ behind the vectors was used in a locality sensitive hashing (LSH) scheme (Charikar, 2002). High dimensional data in $\mathbb{R}^p$ would be mapped to lower dimensional binary vectors in $\mathbb{R}^k$ under SRP, and the computed Hamming distance between the binary pairs would give an estimate of their angular similarity with high probability. The time taken under this technique would be reduced to $O(npk + n^2k)$.

[*]Equal contribution [1]Singapore University Of Technology And Design. Correspondence to: Keegan Kang <keegan_kang@sutd.edu.sg>.

Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

By going one step further and storing the norms of vectors (Li et al., 2006), SRP could also be used to recover the inner products between these vectors. This proved to be competitive with conventional random projections under certain conditions. Binary codes require 1 bit of storage per entry under SRP, but storing real numbers as doubles could take 64 bits per entry under conventional random projections. If the variance of the inner product estimates using SRP is lower than the variance of the inner product estimates using conventional random projections for the same amount of bits, then SRP would be highly preferred.

These results gave rise to an efficient Approximate Nearest Neighbor (ANN) algorithm as computing Hamming distances in the lower dimensional space would be faster than computing angles in the higher dimensional one. An example of this could be image classification, where a 1024×1024 pixel image would mean working with vectors in $\mathbb{R}^p$, $p = 1048576$. Transforming these vectors to binary codes of length k, $k \ll p$ would drastically reduce the time taken.

Moreover, these estimates can be used in conjunction with the "kernel trick", where (estimates of) inner products are used within algorithms such as support vector machines (SVM) to further reduce the computational cost.

1.1. Sign Random Projections

Given a data matrix $X_{n\times p}$, consider a random matrix $R_{p\times k}$ with entries i.i.d from $N(0, 1)$. We compute $V = \texttt{sgn}(XR)$, where we define

$$\texttt{sgn}(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{1}$$

and apply sgn to every element in XR. We set $\texttt{sgn}(x) = 0$ when $x < 0$ for convenience of notation in Figures 2 and 8.

Theorem 1.1. *The estimate for the angle θ between vectors $\mathbf{x}_i, \mathbf{x}_j$ is given by the Hamming distance between $\mathbf{v}_i, \mathbf{v}_j$ multiplied by π/k.*

Proof. This proof is given in (Goemans & Williamson, 1995). □

This estimate of θ is the sum of k Bernoulli observations, which has variance $\text{Var}[\hat{\theta}] = \frac{\theta(\pi-\theta)}{k}$. Figure 1 shows how the theoretical variance of SRP varies with θ.

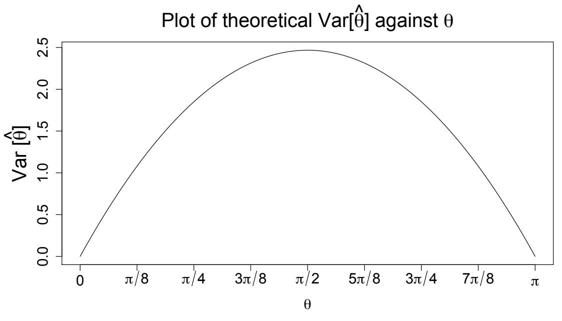

Figure 1. Theoretical variance of SRP based on angle θ between vectors $\mathbf{x}_i, \mathbf{x}_j$.

1.2. Super-Bit LSH

Super-Bit LSH (SBLSH) (Ji et al., 2012) was proposed to reduce the variance of the estimates of angular similarity. SBLSH was proven to have lower variance than SRP, when the angle to estimate is within the interval $[0, \frac{\pi}{2}]$. This algorithm only takes place in the pre-processing stage to get a different random matrix $\tilde{R}$. While SRP only involved generating a random matrix R with k columns, SBLSH divided the k columns into L groups of N columns, and orthogonalized the N vectors within each group. These new vectors become the columns of the new matrix $\tilde{R}$. The subsequent process is then the same as ordinary SRP.

Empirical results with SBLSH showed that given some $K = k$, the best division to ensure a variance reduction is to have $L = 1$ group, with $N = k$ orthogonal vectors.

While SBLSH achieves variance reduction, there are areas where this algorithm could do better. For example, orthogonalizing batches of N vectors can be costly when the dimension of the data p is large, and therefore a longer pre-processing period is needed. Furthermore, geometry theory (Ball, 1997) states that any two normalized vectors sampled in extremely high dimensions would be orthogonal (with high probability) to each other, so SRP would achieve roughly the same performance as SBLSH with large p.

2. Our Contributions

We propose an estimator for SRP by using additional information. Our estimator keeps to the same order of time as SRP, and has a less costly pre-processing period than SBLSH. We demonstrate that our estimator can be used in conjunction with any modifications of SRP such as SBLSH to get a more accurate estimate. We prove that the variance of our estimator is lower than the ordinary SRP estimate. We explain how we can build better and more accurate estimators from our work. Lastly, we demonstrate our results on the MNIST test dataset and the Gisette dataset.

Our work was inspired by Deming (Deming & Stephan, 1940) and Li (Li & Church, 2007). Deming examined the computation of maximum likelihood estimates for a multinomial distribution when the marginal probabilities are known. His result was used by Li to estimate two way associations in word data by making use of information known at the margins. We go one step further by adding information to the dataset to construct our own margins.

2.1. Our Estimator

We denote $\theta_{\mathbf{x}_i,\mathbf{x}_j}$ to be the angle between the vectors $\mathbf{x}_i, \mathbf{x}_j$. Given $X_{n\times p}$, we construct a vector $\mathbf{e} \in \mathbb{R}^p$ and compute and store the angles $\theta_{\mathbf{x}_1,\mathbf{e}}, \ldots, \theta_{\mathbf{x}_n,\mathbf{e}}$. We now proceed the same way as SRP and generate the random matrix $R_{p\times k}$. We store $V = \texttt{sgn}(XR)$ as well as $\mathbf{v}_e = \texttt{sgn}(\mathbf{e}^T R)$.

Without loss of generality, suppose we want to compute $\theta_{\mathbf{x}_1,\mathbf{x}_2}$. Consider the vectors $\mathbf{v}_1 = (v_{11}, \ldots, v_{1k})$, $\mathbf{v}_2 = (v_{21}, \ldots, v_{2k})$, $\mathbf{v}_e = (v_{e1}, \ldots, v_{ek})$ and the k 3-tuples $\{(v_{1s}, v_{2s}, v_{es})\}_{s=1}^{k}$. Each 3-tuple is a binary string of length 3. Suppose we categorize these tuples in a 2×2 contingency table, with the following four sets

$$A = \{s \mid \texttt{sgn}(v_{2s}) = \texttt{sgn}(v_{es})\} \tag{2}$$
$$B = \{s \mid \texttt{sgn}(v_{2s}) \neq \texttt{sgn}(v_{es})\} \tag{3}$$
$$C = \{s \mid \texttt{sgn}(v_{1s}) \neq \texttt{sgn}(v_{es})\} \tag{4}$$
$$D = \{s \mid \texttt{sgn}(v_{1s}) = \texttt{sgn}(v_{es})\} \tag{5}$$

	A	B
C	n_1 (either 011 or 100)	n_2 (either 001 or 110)
D	n_3 (either 111 or 000)	n_4 (either 101 or 010)

Figure 2. Contingency Table For Our Estimator

In statistics, given a contingency table with samples drawn from a large population, the goal is to find the probability of each cell occurring in the population. The probability of exactly observing n_1, n_2, n_3, n_4 counts with $n = n_1 + n_2 + n_3 + n_4$ is given by

$$p = \mathbb{P}\left[X_1 = n_1, X_2 = n_2, X_3 = n_3, X_4 = n_4\right] \tag{6}$$
$$= \binom{n}{n_1\ n_2\ n_3\ n_4} p_1^{n_1} p_2^{n_2} p_3^{n_3} p_4^{n_4} \tag{7}$$

where each X_i denotes the respective cell, and p_i the probability of seeing an observation from that cell. The log likelihood function is given by

$$l(p_1, p_2, p_3, p_4) = C + \sum_{j=1}^{4} n_j \log(p_j) \tag{8}$$

where C is some constant. The maximum likelihood estimates are given by $\hat{p_i} = \frac{n_i}{n}$. In SRP and SBLSH, we compute $\frac{n_2+n_3}{n}$ to estimate $\theta_{\mathbf{x}_i,\mathbf{x}_j}$.

However, *we already know* the probability $p_1 + p_3$ of observing $n_1 + n_3$ and the probability $p_3 + p_4$ of observing $n_3 + n_4$. These probabilities are given by $1 - \frac{\theta_{\mathbf{x_2 e}}}{\pi}$ and $1 - \frac{\theta_{\mathbf{x_1 e}}}{\pi}$ respectively. We also know that $\sum_i p_i = 1$.

The maximum likelihood estimate (8) can be rewritten in terms of p_3

$$\begin{aligned} l(p_3) = {} & n_1 \log\left(1 - \frac{\theta_{\mathbf{x_2 e}}}{\pi} - p_3\right) + n_4 \log\left(1 - \frac{\theta_{\mathbf{x_1 e}}}{\pi} - p_3\right) \\ & + n_3 \log(p_3) + n_2 \log\left(p_3 - 1 + \frac{\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}}}{\pi}\right) \end{aligned} \tag{9}$$

with the following first and second derivatives given by

$$\begin{aligned} \frac{dl}{dp_3} = {} & \frac{n_1 \pi}{\theta_{\mathbf{x_2 e}} + \pi(p_3 - 1)} + \frac{n_4 \pi}{\theta_{\mathbf{x_1 e}} + \pi(p_3 - 1)} \\ & + \frac{n_2 \pi}{\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}} + \pi(p_3 - 1)} + \frac{n_3}{p_3} \end{aligned} \tag{10}$$

$$\begin{aligned} \frac{d^2 l}{dp_3^2} = {} & -\frac{n_1 \pi^2}{(\theta_{\mathbf{x_2 e}} + \pi(p_3 - 1))^2} - \frac{n_4 \pi^2}{(\theta_{\mathbf{x_1 e}} + \pi(p_3 - 1))^2} \\ & - \frac{n_2 \pi^2}{\left(\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}} + \pi(p_3 - 1)\right)^2} - \frac{n_3}{p_3^2} \end{aligned} \tag{11}$$

To maximize $l(p_3)$ in (9), we need $\hat{p}_3$ which makes the first derivative zero. The second derivative is always negative which makes our value of p_3 a maximum. We mention that the denominators of the fractions in $\frac{dl}{dp_3}$ are only equal to zero when $p_i = 0$, for $i = 1, 2, 3, 4$. By using the relationships $p_1 + p_3 = 1 - \frac{\theta_{\mathbf{x_2 e}}}{\pi}$, $p_3 + p_4 = 1 - \frac{\theta_{\mathbf{x_1 e}}}{\pi}$, $p_1 + p_4 = \frac{\theta}{\pi}$ and $\sum_i p_i = 1$, we can express p_i's in terms of $\theta_{\mathbf{x_1 e}}, \theta_{\mathbf{x_2 e}}$ and θ:

$$\begin{aligned} p_1 &= \tfrac{\theta + \theta_{\mathbf{x_1 e}} - \theta_{\mathbf{x_2 e}}}{2\pi} & p_2 &= \tfrac{\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}} - \theta}{2\pi} \\ p_3 &= \tfrac{2\pi - \theta_{\mathbf{x_1 e}} - \theta_{\mathbf{x_2 e}} - \theta}{2\pi} & p_4 &= \tfrac{\theta + \theta_{\mathbf{x_2 e}} - \theta_{\mathbf{x_1 e}}}{2\pi} \end{aligned} \tag{12}$$

Hence, the denominators of the fractions in $\frac{dl}{dp_3}$ are equal zero when $\theta = -\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}}$, $\theta = \theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}}$, $\theta + \theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}} = 2\pi$ or $\theta = \theta_{\mathbf{x_1 e}} - \theta_{\mathbf{x_2 e}}$. These are the cases when all three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{e}$ lie on the same 2D-plane.

Newton Raphson can be used to find $\hat{p}_3$ with the starting value $\frac{n_3}{n}$, which has quadratic convergence. We should cross multiply the terms in the first derivative and use Newton Raphson on the numerator for numerical stability.

Once $\hat{p}_3$ is found, we compute $\hat{p}_2$ by $\hat{p}_3 - 1 + \frac{\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}}}{\pi}$, and estimate $\theta_{\mathbf{x_1 x_2}}$ by $\hat{\theta} = \pi(1 - \hat{p}_2 - \hat{p}_3)$. Algorithm 1 describes our algorithm for our estimator.

Algorithm 1 Algorithm For Our Estimator

```
Pre-processing stage
Initialize $\mathbf{e}$ ; Initialize $R$
Compute $V = \mathtt{sgn}(XR)$ ; Compute $\mathbf{v}_e = \mathtt{sgn}(\mathbf{e}^T R)$
for each $\mathbf{x}_i \in X$ do
    Compute and store $\theta_{\mathbf{x}_i \mathbf{e}}$
end
Actual stage
for each $\mathbf{v}_i, \mathbf{v}_j \in V$ do
    Count $n_1, n_2, n_3, n_4$ as in Figure 2
    Find $\hat{p}_3$ which maximizes $l(p_3)$ in Equation 9
    Compute $\hat{p}_2 = \hat{p}_3 - 1 + \frac{\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}}}{\pi}$
    Set $\hat{\theta}_{\mathbf{x}_i \mathbf{x}_j} = \pi(1 - \hat{p}_2 - \hat{p}_3)$
end
```

2.2. Theoretical Variance Of Our Estimator

Theorem 2.1. *The variance of our maximum likelihood estimate $\hat{\theta}$ is given by*

$$Var[\hat{\theta}] = \frac{4\pi^2}{k\left(\frac{1}{p_1} + \frac{1}{p_2} + \frac{1}{p_3} + \frac{1}{p_4}\right)} \tag{13}$$

Proof. Our estimate $\hat{\theta}$ is given by

$$\hat{\theta} = \pi(1 - \hat{p}_2 - \hat{p}_3) = -2\pi\hat{p}_3 + 2\pi - (\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}}) \tag{14}$$

and its variance given by $\mathrm{Var}[\hat{\theta}] = 4\pi^2 \mathrm{Var}[\hat{p}_3]$ where $\hat{p}_3$ is our maximum likelihood estimate. Thus we need to find $\mathrm{Var}[\hat{p}_3]$. It can be shown (Shao, 2003) that $\hat{p}_3$ converges in distribution to a normal random variable $N(p_3, \frac{1}{I(p_3)})$. $I(p_3)$ is the expected Fisher information of p_3. We have that

$$\begin{aligned} I(p_3) &= \mathbb{E}\left[-\frac{d^2 l}{dp_3^2}\right] \\ &= \frac{\mathbb{E}[n_1]\pi^2}{(\theta_{\mathbf{x_2 e}} + \pi(p_3 - 1))^2} + \frac{\mathbb{E}[n_4]\pi^2}{(\theta_{\mathbf{x_1 e}} + \pi(p_3 - 1))^2} \\ &\quad + \frac{\mathbb{E}[n_2]\pi^2}{\left(\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}} + \pi(p_3 - 1)\right)^2} + \frac{\mathbb{E}[n_3]}{p_3^2} \qquad (15) \\ &= \frac{k p_1 \pi^2}{(\theta_{\mathbf{x_2 e}} + \pi(p_3 - 1))^2} + \frac{k p_4 \pi^2}{(\theta_{\mathbf{x_1 e}} + \pi(p_3 - 1))^2} \\ &\quad + \frac{k p_2 \pi^2}{\left(\theta_{\mathbf{x_1 e}} + \theta_{\mathbf{x_2 e}} + \pi(p_3 - 1)\right)^2} + \frac{k p_3}{p_3^2} \qquad (16) \\ &= \frac{k}{p_1} + \frac{k}{p_2} + \frac{k}{p_3} + \frac{k}{p_4} \qquad (17) \\ &= k\left(\frac{1}{p_1} + \frac{1}{p_2} + \frac{1}{p_3} + \frac{1}{p_4}\right) \qquad (18) \end{aligned}$$

and hence

$$\mathrm{Var}[\hat{p}_3] = \frac{1}{k\left(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4}\right)} \tag{19}$$

We note that $\mathrm{Var}[\hat{p}_3]$ tends to 0 when any p_i tends to 0. □

This maximum likelihood estimate can be biased with small k, but the bias vanishes and the estimator enjoys good performance when k is large. This is not surprising since maximum likelihood estimators are consistent and asymptotically efficient estimators (Shao, 2003).

Theorem 2.2. *The variance of our maximum likelihood estimate $Var[\hat{\theta}]$ is lower than the original variance $Var[\hat{\theta}_{orig}]$, with equality iff i) $p_1 = p_4$ and $p_2 = p_3$, i.e. $\theta_{\mathbf{x}_1\mathbf{e}} = \theta_{\mathbf{x}_2\mathbf{e}} = \frac{\pi}{2}$, ii) $p_1 = p_4 = 0$, i.e. $\theta = 0$ or iii) $p_2 = p_3 = 0$, i.e. $\theta = \pi$.*

Proof. We have that the variance of $\hat{\theta}$ under SRP is given by $\frac{\theta(\pi-\theta)}{k}$, which can be rewritten as $\frac{\pi^2(p_1+p_4)(p_2+p_3)}{k}$. It suffices to show that $V := \mathrm{Var}[\hat{\theta}_{\text{orig}}] - \mathrm{Var}[\hat{\theta}] \geq 0$ under the constraints that $\sum_i p_i = 1$. We can show that

$$V = \frac{\pi^2\left[(p_1+p_4)(p_2+p_3)(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4}) - 4\right]}{k(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4})} \tag{20}$$

$$= \frac{\pi^2\left[\frac{(p_1-p_4)^2}{p_1p_4}(p_2+p_3) + \frac{(p_2-p_3)^2}{p_2p_3}(p_1+p_4)\right]}{k(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4})}, \tag{21}$$

which clearly is always positive, and equal to zero when $p_1 = p_4$ and $p_2 = p_3$, or $p_1 = p_4 = 0$ or $p_2 = p_3 = 0$. □

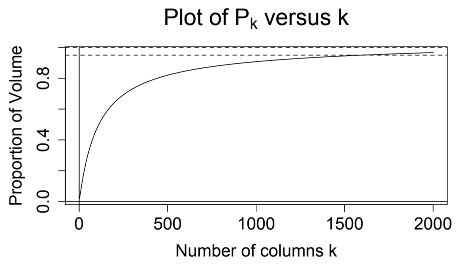

Figure 3. Proportion P_k of $(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}})$ with variance reduction against number of columns, k.

While Theorem 2.2 guarantees we always achieve a lower variance or do no worse than the original variance, we should also account for the storage of $\theta_{\mathbf{x}_i,\mathbf{e}}, 1 \leq i \leq n$. If we store these values in 64 bits, then for a fixed k, we should consider $\mathrm{Var}[\hat{\theta}]$ with k columns versus $\mathrm{Var}[\hat{\theta}_{\text{orig}}]$ with $k + 64$ columns, as storing 64 extra bits is equivalent to running and storing the results of SRP 64 more times.

We notice that

$$\mathrm{Var}[\hat{\theta}_{\text{orig}}]_{k+64} - \mathrm{Var}[\hat{\theta}]_k \tag{22}$$

$$= \frac{\pi^2(p_1+p_4)(p_2+p_3)}{k+64} - \frac{4\pi^2}{k\left(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4}\right)} \tag{23}$$

$$= \frac{\pi^2\left[k(p_1+p_4)(p_2+p_3)(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4}) - 4(k+64)\right]}{k(k+64)(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4})} \tag{24}$$

$$= \frac{k\pi^2\left[\frac{(p_1-p_4)^2}{p_1p_4}(p_2+p_3) + \frac{(p_2-p_3)^2}{p_2p_3}(p_1+p_4) - \frac{256}{k}\right]}{k(k+64)(\frac{1}{p_1}+\frac{1}{p_2}+\frac{1}{p_3}+\frac{1}{p_4})}, \tag{25}$$

which is only positive for certain configurations of p_1, p_2, p_3, p_4, which depend on $(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}})$. However, assuming that $(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}})$ is uniformly distributed in $S :=$ the set of all valid triplets $(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}})$, the proportion of points $(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}})$ with a variance reduction

$$P_k := \frac{\text{volume}(\{(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}}) \in S \mid \mathrm{Var}[\hat{\theta}_{\text{orig}}]_{k+64} - \mathrm{Var}[\hat{\theta}]_k \geq 0\})}{\text{volume}(S)} \tag{26}$$

rapidly increases to 0.95 at about $k = 1500$ as seen in Figure 3. However, the distribution of our angles in practical applications are not uniform, as data must have some structure, hence we can have a better proportion of "good points" with smaller k, as we show in our experiments.

2.3. The choice of the extra vector e

While Theorem 2.2 gives us an inequality guaranteeing a variance reduction or at least being no worse off, we also consider how $(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}})$ affect our variance reduction.

Consider the 3D set of axes with $x \in [0, \pi], y \in [0, \pi], z \in [0, \pi]$. We let x denote values of $\theta_{\mathbf{x}_i,\mathbf{e}}$, y denote values of $\theta_{\mathbf{x}_j,\mathbf{e}}$, and z denote values of θ. Our goal is to visualize the 3D region that consists of **valid** values of $(\theta, \theta_{\mathbf{x}_1\mathbf{e}}, \theta_{\mathbf{x}_2\mathbf{e}})$.

By visualizing any arbitrary three vectors on the unit sphere $\mathbb{S}^2$, we can have any two angles θ_i, θ_j between these three vectors vary between $(0, \pi)$, but the third angle θ_k necessarily needs to fulfill $\theta_i + \theta_j + \theta_k \leq 2\pi$ and $\theta_i + \theta_j \geq \theta_k$.

Hence the 3D region of feasible $\theta, \theta_{\mathbf{x}_i,\mathbf{e}}$, and $\theta_{\mathbf{x}_j,\mathbf{e}}$ is the intersection of the following regions given by $\{0 \leq x + y + z \leq 2\pi\}$, $\{0 \leq z \leq x + y\}$, $\{0 \leq y \leq x + z\}$, and $\{0 \leq x \leq y + z\}$, which is the tetrahedron with $(0, 0, 0)$, $(\pi, \pi, 0)$, $(\pi, 0, \pi)$ and $(0, \pi, \pi)$ as its corners. The four faces of the tetrahedron correspond to the cases when $\mathrm{Var}[\hat{\theta}] = 0$ and the center $\left(\frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}\right)$ corresponds to $\mathrm{Var}[\hat{\theta}]_{\max} = \mathrm{Var}[\hat{\theta}_{\text{orig}}]_{\max} = \frac{\pi^2}{4k}$.

Moreover, the regions which correspond to negligible or no reduction in variance, i.e. $\mathrm{Var}[\hat{\theta}_{\text{orig}}] - \mathrm{Var}[\hat{\theta}] < \epsilon$ occur near the three lines $\{x + y = \pi, z = \pi\}$, $\{x = y, z = 0\}$ and

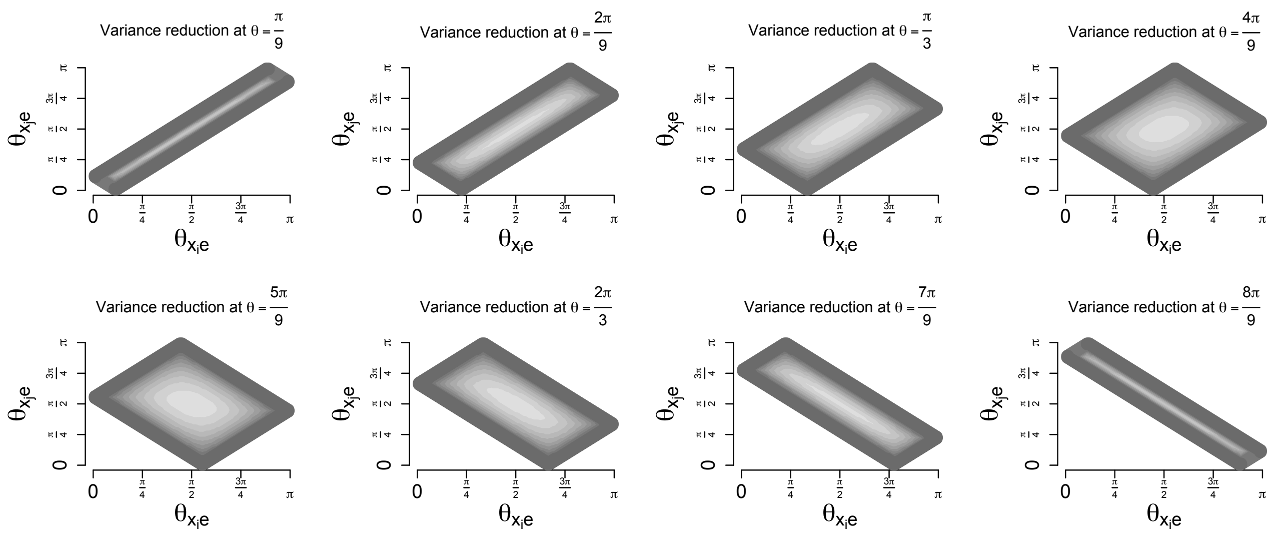

Figure 4. Plots of $\text{Var}[\hat{\theta}_{\text{orig}}] - \text{Var}[\hat{\theta}]$ with varying θ values. Darker lines denote greater variance reduction.

$\{x = y = \frac{\pi}{2}, 0 \leq z \leq \pi\}$. The first two lines $\{x + y = \pi, z = \pi\}$, $\{x = y, z = 0\}$ are the cases where the original variance $\text{Var}[\hat{\theta}_{\text{orig}}]$ is zero, thus no variance reductions can occur. The third line $\{x = y = \frac{\pi}{2}, 0 \leq z \leq \pi\}$ contains the cases where the additional vector $\mathbf{e}$ doesn't provide us additional information as it's orthogonal to $\mathbf{x}_1$ and $\mathbf{x}_2$.

Figure 4 shows some plots of $\text{Var}[\hat{\theta}_{\text{orig}}] - \text{Var}[\hat{\theta}]$ over the valid region of $(\theta_{\mathbf{x_1 e}}, \theta_{\mathbf{x_2 e}})$ for some fixed θ's. We can see that the variance reduction always occurs at the boundary and no significant variance reductions near the center.

The choice of $\mathbf{e}$ has to be fixed for our dataset X. Estimating all pairwise similarities with a fixed extra vector $\mathbf{e}$ implies that some pairs $\mathbf{x}_i, \mathbf{x}_j$ will enjoy good values of $\theta_{\mathbf{x}_i,e}, \theta_{\mathbf{x}_j,\mathbf{e}}$ to get substantial variance reduction if the point $(\theta, \theta_{\mathbf{x}_i,\mathbf{e}}, \theta_{\mathbf{x}_j,\mathbf{e}})$ stays away from the three lines. However, bad pairs $\mathbf{x}_{i'}, \mathbf{x}_{j'}$ would not get substantial variance reduction if $(\theta, \theta_{\mathbf{x}_{i'},\mathbf{e}}, \theta_{\mathbf{x}_{j'},\mathbf{e}})$ is near the three lines.

As a heuristic, we first normalize the vectors $\mathbf{x_i}$, then we set $\mathbf{e}$ to be the first singular vector of X to maximize the proportion of good pairs. The intuition behind this is that among all the unit vectors $\mathbf{v}$, $\mathbf{v} = \mathbf{v_1} :=$ first singular vector of X is a vector that will maximize

$$\|X\mathbf{v}\|^2 = \sum_{\mathbf{i}} (\mathbf{x_i} \cdot \mathbf{v})^2 = \sum_{\mathbf{i}} \cos^2 \theta_{\mathbf{x_i},\mathbf{v}}. \qquad (27)$$

Intuitively, this means $\mathbf{v}_1$ will make a small angle or an angle near π with the majority of the $\mathbf{x}_i$'s, which implies that the majority of $(\theta, \theta_{\mathbf{x}_i,\mathbf{v}_1}, \theta_{\mathbf{x}_j,\mathbf{v}_1})$ will stay far away from the low variance reduction region $\{\theta_{\mathbf{x}_i,\mathbf{v}_1} = \theta_{\mathbf{x}_i,\mathbf{v}_2} = \frac{\pi}{2}, 0 \leq \theta \leq \pi\}$.

However, it may be the case that the best choice of $\mathbf{e}$ depends on the data and the type of analysis to be done.

2.4. Analysis Of Our Estimator

We look at the computational cost of our estimator in the pre-processing period and the evaluation period.

The computational cost could vary widely based on generating the extra vector $\mathbf{e}$. If we want an exact value of the first singular vector, this could be computationally expensive at $O(\min(np^2, n^2p))$. However, as Theorem 2.2 guarantees that any $\mathbf{e}$ would do no worse, we could get an estimation of the first singular vector by probabilistic algorithms (Halko et al., 2011), which can take $O(n^2 s)$, where $s < n$ is a parameter chosen based on the data.

Finding all angles between $\mathbf{e}$ and $\mathbf{x}_i, 1 \leq i \leq n$ takes $O(np)$ time, which is the same as the cost of normalizing or scaling data. Storing these angles takes $O(n)$ space, which is an extra 64 bits per observation.

To estimate the angle between each vector pair, our estimator takes $O(k)$ time which is the same order of time as SRP and SBLSH. Computing the values n_i take $O(k)$ time, and using Newton Raphson costs $O(1)$.

Hence, the overall computational complexity of our algorithm is $O(n^2 s + npk + n^2 k)$ which is $O(\min\{n^2 k, n^2 s\})$.

Our estimator only requires the probability of the random hyperplane separating the two vectors $\mathbf{x}_i, \mathbf{x}_j$. Any derivatives of SRP which have a different probability can still utilize this estimator.

3. Our Experiments

We run our algorithm on the MNIST test dataset (Lecun et al., 1998) and Gisette dataset (Guyon et al., 2005; Lichman, 2013). The MNIST test dataset has $n = 10,000$ observations, and $p = 784$ parameters. The Gisette dataset has $n = 13,500$ observations, and $p = 5,000$ parameters. For both datasets, we normalize the vectors to unit length.

We demonstrate our estimator on both SRP and SBLSH with the super bit depth being the number of columns of the random projection matrix k to ensure the best accuracy. We set the vector $\mathbf{e}$ to be the first singular vector of our datasets for reproducibility.

We run our experiments for the number of columns k (equivalently number of bits) ranging from $\{64, 128, \ldots, 3008\}$ of our random matrix over 100 simulations for both datasets. The motivation for the multiple of 64 stems from the fact that we are keeping an extra 64 bits of information per vector for our estimator by storing the extra angles. This can be thought of as computing and storing an additional 64 more bits with SRP, SBLSH without the use of our estimator. This means that our plots for our estimator are "shifted" to the right by 64 units to account for extra storage. Nevertheless, our estimator still performs well even accounting for the extra bits which we will show in our experiments.

We denote `SRP` and `SBLSH` to be our baselines, and `SRP-est`, `SBLSH-est` to be SRP implemented with our estimator, and SBLSH implemented with our estimator respectively.

Our goal is to show that while our estimator depends on one fixed $\mathbf{e}$ and varying $\theta_{\mathbf{x}_i,e}, \theta_{\mathbf{x}_j,e}$ for all vector pairs, we can still get competitive variance reduction over all vector pairs in estimating angular similarity.

Looking at all pairwise estimates also makes a fairer test, as we do not want to only look at "good pairs of vectors" where θs are within our region and far away from the boundary. Hence we compute the average RMSE of all pairwise estimates of the inner product. This ensures that all good and bad pairs of vectors are accounted for and that there is no cherry picking at all.

We compute the average RMSE of 49,995,000 pairwise angular similarity estimates for the MNIST test dataset, as well as the average RMSE of 91,118,250 pairwise angular similarity estimates for the Gisette dataset. We display them in Figure 5 and Figure 6 respectively. The left side of these plots show our RMSE averaged over 100 simulations, while the right side of these plots show the standard deviations of these RMSE.

In general, we see that `SRP-est` and `SBLSH` have lower and similar average RMSE when computing all pairwise angular similarities compared with `SRP`. In this case, `SBLSH` may potentially outperform `SRP-est` with lower average RMSE. However, as our estimator can be used with any derivation of SRP, we have `SBLSH-est` giving an even lower RMSE.

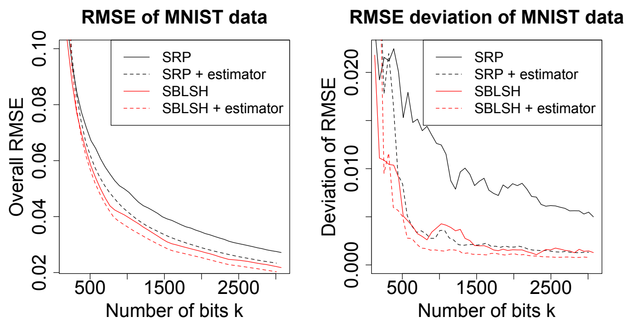

Figure 5. The average RMSE of our pairwise angular similarities for the MNIST dataset are shown in the left plot, and standard deviations of the RMSE shown on the right plot.

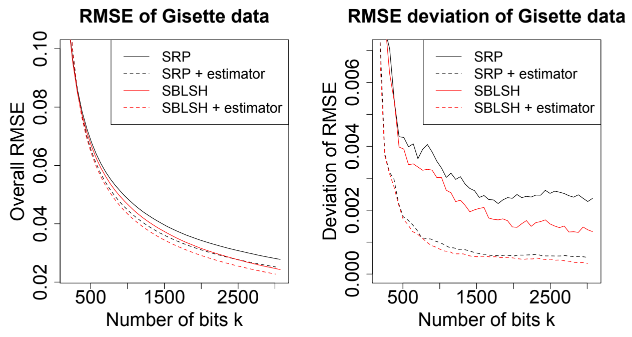

Figure 6. The average RMSE of our pairwise angular similarities for the Gisette dataset are shown in the left plot, and standard deviations of the RMSE shown on the right plot.

When we look at the deviations of the average RMSE of all pairwise estimates, we see that `SRP-est`, `SBLSH-est` tend to have lower standard deviations compared to `SRP`, `SBLSH`.

To give some intuition to magnitude of the right and left graphs of Figure 5 and Figure 6, we show the exact values of the RMSE estimates and 3 standard deviations at $k = 1024$ in Figure 7.

Overall, we see that as k is sufficiently large ($k \approx 256$), using our estimator in conjunction with `SRP` or `SBLSH` can result in a lower RMSE (and lower deviations). Moreover, if our observations are extremely high dimensional, our

	MNIST	Gisette
`SRP`	0.0471 ± 0.0055	0.0466 ± 0.0091
`SRP-est`	**0.0451 ± 0.0049**	**0.0419 ± 0.0025**
`SBLSH`	0.0386 ± 0.0019	0.0446 ± 0.0079
`SBLSH-est`	**0.0378 ± 0.0017**	**0.0405 ± 0.0021**

Figure 7. Average RMSE of pairwise estimates with bounds of 3 standard deviations at $k = 1024$.

estimator may be preferred to $\widehat{\text{SBLSH}}$ due to lower computational complexity in the pre-processing stage, since `SRP-est` and `SBLSH-est` have about the same performance.

Our experiments also do verify the theory that when k is large, our maximum likelihood estimates are unbiased and outperforms SRP and SBLSH.

4. Building Better Estimators When k Is Large

Suppose we repeat the process in building our estimator and generate vectors $\mathbf{e}_1, \mathbf{e}_2$. We can compute and store the angles $\theta_{\mathbf{x}_i \mathbf{e}_j}$ for all $1 \leq i \leq n, 1 \leq j \leq 2$, and compute $V = \mathtt{sgn}(XR), \mathbf{v}_{e_1} = \mathtt{sgn}(\mathbf{e}_1^T R), \mathbf{v}_{e_2} = \mathtt{sgn}(\mathbf{e}_2^T R)$. To compute $\theta_{\mathbf{x}_1 \mathbf{x}_2}$, then we look at the k 4-tuples $\{(v_{1s}, v_{2s}, v_{e_1 s}, v_{e_2 s})\}_{s=1}^{k}$.

We then have a $2 \times 2 \times 2$ contingency table, which we give in Figure 8 in terms of n_1 to n_8.

n_1 (either 1001 or 0110)	n_2 (either 1100 or 0011)
n_5 (either 1110 or 0001)	n_6 (either 1010 or 0101)

n_3 (either 0111 or 1000)	n_4 (either 1101 or 0010)
n_7 (either 1111 or 0000)	n_8 (either 1011 or 0100)

Figure 8. Contingency Table(s) for Estimator 2

Similar to Estimator 1, we now compute the probabilities of the six margins consisting of $M_1 = n_5 + n_6, M_2 = n_2 + n_3, M_3 = n_1 + n_4, M_4 = n_1 + n_5, M_5 = n_2 + n_8, M_6 = n_4 + n_6$. We require the following theorem for this computation.

Theorem 4.1. *Suppose we have three vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$, with angles θ_{ij} being the angle between $\mathbf{x}_i, \mathbf{x}_j$. Given a random hyperplane, the probability that all the vectors are on one side of the hyperplane is given by $1 - \frac{\sum_{ij} \theta_{ij}}{2\pi}$.*

Proof. We can find some subspace spanned by orthogonal vectors $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ such that $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are represented in terms of these $\mathbf{b}_i s$. We can therefore think of this subspace as a "copy" of $\mathbb{R}^3$. WLOG, let $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ be unit vectors corresponding to points A, B, C on the unit sphere $\mathbb{S}^2$, which define a spherical triangle ABC with sides $a = \theta_{2,3}$, $b = \theta_{1,3}$ and $c = \theta_{1,2}$, as the radius of the sphere is 1.

For the 2D plane containing the origin, A and B, consider its unique normal vector pointing at the same hemisphere containing C. We let C' be the point of intersection between that normal vector and the unit sphere. We define A' and B' similarly and thus $A'B'C'$ is the polar triangle corresponding to the spherical triangle ABC. For any 2D plane containing the origin, it corresponds to a unique normal vector pointing at the same hemisphere that contains $A'B'C'$.

It is clear any 2D plane containing the origin that intersects the spherical triangle ABC if and only if it separates one $\mathbf{x}_i$ from the other two vectors (modulo the measure zero case when the plane contains two corners of the spherical triangle ABC). It is also clear that the corresponding normal vector of such a 2D plane intersects the sphere outside the polar triangle $A'B'C'$. Hence, we have that

$$\begin{aligned} &\mathbb{P}[\text{the 3 vectors are not all on the same side of the hyperplane}] \\ &= \frac{\text{Area of hemisphere} - \text{Area of polar triangle } A'B'C'}{\text{Area of hemisphere}} \end{aligned} \tag{28}$$

so

$$\begin{aligned} &\mathbb{P}[\text{all the 3 vectors are on one side of the hyperplane}] \\ &= \frac{\text{Area of polar triangle } A'B'C'}{\text{Area of hemisphere}} \end{aligned} \tag{29}$$

Since the polar triangle $A'B'C'$ is a spherical triangle as well, its area is given by $\angle A' + \angle B' + \angle C' - \pi$. A standard result in spherical trigonometry shows the polar triangle $A'B'C'$ is related to the spherical triangle ABC by

$$\angle A' = \pi - a, \quad \angle B' = \pi - b, \quad \angle C' = \pi - c \tag{30}$$

Putting everything together, we have

$$\mathbb{P}[\text{all the 3 vectors are on one side of the hyperplane}]$$

$$= \frac{\text{Area of polar triangle } A'B'C'}{\text{Area of hemisphere}} \tag{31}$$

$$= \frac{\angle A' + \angle B' + \angle C' - \pi}{\frac{4\pi}{2}} \tag{32}$$

$$= \frac{\pi - a + \pi - b + \pi - c - \pi}{2\pi} \tag{33}$$

$$= \frac{2\pi - \theta_{2,3} - \theta_{1,3} - \theta_{1,2}}{2\pi} \tag{34}$$

$$= 1 - \frac{\theta_{1,2} + \theta_{2,3} + \theta_{1,3}}{2\pi}. \tag{35}$$

□

Theorem 4.1 allows us to express p_is in Figure 8 in terms of $\theta_{\mathbf{x}_1,\mathbf{e}_1}, \theta_{\mathbf{x}_1,\mathbf{e}_2}, \theta_{\mathbf{x}_2,\mathbf{e}_1}, \theta_{\mathbf{x}_2,\mathbf{e}_2}, \theta$, and we can estimate θ by optimizing the new log-likelihood function

$$l = C + \sum_{i=1}^{8} n_i \log(p_i) \tag{36}$$

by using stored values of $\theta_{\mathbf{x}_1,\mathbf{e}_1}, \theta_{\mathbf{x}_1,\mathbf{e}_2}, \theta_{\mathbf{x}_2,\mathbf{e}_1}, \theta_{\mathbf{x}_2,\mathbf{e}_2}$.

In fact, we could keep on generating vectors $\mathbf{e}_1, \ldots, \mathbf{e}_s$ to create more estimators if we know the probabilities of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_s$ being on one side of the hyperplane, which can be found via geometry theory. Algorithm 2 shows the general estimation algorithm.

Algorithm 2 General Algorithm For Our Estimators

```
Pre-processing stage
Initialize $\mathbf{e}_1, \ldots, \mathbf{e}_s$ ; Initialize $R$
Compute $V = \mathtt{sgn}(XR)$
Compute all $\mathbf{v}_{e_s} = \mathtt{sgn}(\mathbf{e}_s^T R)$
for each $\mathbf{x}_i \in X$ do
    Compute and store $\theta_{\mathbf{x}_i \mathbf{e}_s}$
    Compute and store $\theta_{\mathbf{e}_s \mathbf{e}_t}$
end
Actual stage
for each $\mathbf{v}_i, \mathbf{v}_j \in V$ do
    Count $n_1, n_2, n_3, \ldots, n_{2^{s+1}}$
    Factorize log-likelihood into disjoint parts
    Find parameters $\hat{p}_t$ which maximizes these log likelihoods
    Express $\hat{\theta}$ in terms of required $\hat{p}_t$.
end
```

We note that these estimators only prove to be competitive when k is large, as the maximum likelihood estimates will be biased with small k. Intuitively, consider what it means if we set an arbitrary $k = 64$, and we have 8 cells. For some $p_i = \frac{a}{b}, a < b$, the observed $\frac{n_i}{k}$ would not be close to p_i unless k is large. For example, our original estimator quickly outperforms SRP and SBLSH when k is around 256 for our datasets. An estimator with vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ would outperform our original estimator when k is an order of magnitude greater.

We conclude that constructing such estimators are only useful if we have extremely large k, which may not be practical in some cases.

5. Future Work And Applications Of Our Estimator

From our theoretical analysis, any $\mathbf{e}$ generated always gives a reduction in variance or do no worse than the original estimate. In our experiments, we have set $\mathbf{e}$ to be the first singular vector as our heuristic. However, $\hat{\mathbf{e}}$ may be dependent on the problem to be solved. Finding $\hat{\mathbf{e}}$ which theoretically yields the most variance reduction based on a particular problem may be a good avenue of future research.

However, this strategy of adding information to a dataset to take advantage of margins gives rise to many possibilities to further reduce the variance of several other estimates of distances, and we briefly describe some of them here.

Conventional random projections - Conventional random projections are used to estimate the inner products, Euclidean distances or l_p distances (Li et al., 2012) with even values of p between vectors. The estimates are continuous, so a control variate (Kang, 2017) approach together with adding additional information may yield good results.

Stable random projections - Stable random projections are used to estimate l_α distances between vectors. Similar to conventional random projections, we could add extra information and use a control variate technique to further improve these estimates.

Minwise hashing - b-bit minwise hashing (Li & König, 2010) is used to estimate the resemblance of binary vectors, while weighted minwise hashing algorithms (Shrivastava, 2016; Ioffe, 2010) are used to estimate the Jaccard similarity between vectors. We can add an extra information $\mathbf{e}$ that is a binary vector (or weighted vector) to better improve the estimates of the resemblance (or Jaccard similarity).

Conditional random sampling - Conditional random sampling (Church et al., 2006) is a local sampling strategy which is used to estimate Hamming distances and χ^2 distances between vectors, amongst other distances. Adding extra information could also improve these estimates.

6. Conclusion

We have demonstrated that our estimator works well with SRP and SBLSH, as well as showed how to construct similar estimators of the same form.

Our estimator can also be easily implemented on existing applications which use SRP or a modification of SRP without running SRP again. As we are only counting the margins, we can set $\mathbf{e}_1$ to be an existing vector $\mathbf{x}_s$, and compute the respective angles $\theta_{\mathbf{x}_s,\mathbf{x}_i}, s \neq i$.

While we have demonstrated an estimator with good performance, we feel that the idea behind the construction of this estimator (and subsequent estimators) is more important. We hope this idea can be extended to many other practical applications as well.

Acknowledgements

We thank the reviewers who provided us with many helpful comments. We hope we have addressed most of these comments in this version of the paper where possible. This research was supported by the SUTD Faculty Fellow Grant RGFECA17003.

References

Ball, K. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31:1–58, 1997. URL http://library.msri.org/books/Book31/files/ball.pdf.

Charikar, M. S. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pp. 380–388. ACM, 2002.

Church, K. W., Li, P., and Hastie, T. J. Conditional random sampling: A sketch-based sampling technique for sparse data. In *In NIPS*, pp. 873–880, 2006.

Deming, W. E. and Stephan, F. F. On a least squares adjustment of a sampled frequency table when the expected marginal totals are known. *The Annals of Mathematical Statistics*, 11(4):427–444, 1940.

Goemans, M. X. and Williamson, D. P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

Guyon, I., Gunn, S., Ben-Hur, A., and Dror, G. Result analysis of the nips 2003 feature selection challenge. In Saul, L. K., Weiss, Y., and Bottou, L. (eds.), *Advances in Neural Information Processing Systems 17*, pp. 545–552. MIT Press, 2005.

Halko, N., Martinsson, P. G., and Tropp, J. A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, May 2011. ISSN 0036-1445. doi: 10.1137/090771806. URL http://dx.doi.org/10.1137/090771806.

Ioffe, S. Improved consistent sampling, weighted minhash and l1 sketching. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pp. 246–255. IEEE, 2010.

Ji, J., Li, J., Yan, S., Zhang, B., and Tian, Q. Super-bit locality-sensitive hashing. In *Advances in Neural Information Processing Systems*, pp. 108–116, 2012.

Kang, K. *Using the Multivariate Normal to Improve Random Projections*, pp. 397–405. Springer International Publishing, Cham, 2017. ISBN 978-3-319-68935-7. doi: 10.1007/978-3-319-68935-7_43. URL https://doi.org/10.1007/978-3-319-68935-7_43.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Li, P. and Church, K. W. A Sketch Algorithm for Estimating Two-Way and Multi-Way Associations. *Comput. Linguist.*, 33(3):305–354, September 2007. ISSN 0891-2017. doi: 10.1162/coli.2007.33.3.305. URL http://dx.doi.org/10.1162/coli.2007.33.3.305.

Li, P. and König, C. b-bit minwise hashing. In *Proceedings of the 19th international conference on World wide web*, pp. 671–680. ACM, 2010.

Li, P., Hastie, T., and Church, K. W. Improving Random Projections Using Marginal Information. In Lugosi, G. and Simon, H.-U. (eds.), *COLT*, volume 4005 of *Lecture Notes in Computer Science*, pp. 635–649. Springer, 2006. ISBN 3-540-35294-5.

Li, P., Mahoney, M. W., and She, Y. Approximating higher-order distances using random projections. *CoRR*, abs/1203.3492, 2012. URL http://arxiv.org/abs/1203.3492.

Lichman, M. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Shao, J. *Mathematical Statistics*. Springer Texts in Statistics. Springer, 2003. ISBN 9780387953823. URL https://books.google.com.sg/books?id=cyqTPot17QcC.

Shrivastava, A. Exact weighted minwise hashing in constant time. *arXiv preprint arXiv:1602.08393*, 2016.