Low rank approximation and decomposition of large matrices using error correcting codes

Shashanka Ubaru, Arya Mazumdar, *Member, IEEE,* and Yousef Saad

***Abstract*—Low rank approximation is an important tool used in many applications of signal processing and machine learning. Recently, randomized algorithms were proposed to effectively construct low rank approximations and obtain approximate singular value decompositions of large matrices. Similar ideas were used to solve least squares regression problems. In this paper, we show how matrices from error correcting codes can be used to find such low rank approximations and matrix decompositions, and extend the framework to linear least squares regression problems.**

The benefits of using these code matrices are the following: (i) They are easy to generate and they reduce randomness significantly. (ii) Code matrices have low coherence and have a better chance of preserving the geometry of an entire subspace of vectors. (iii) For parallel and distributed applications, code matrices have significant advantages over structured random matrices and Gaussian random matrices. (iv) Unlike Fourier or Hadamard transform matrices, which require sampling $O(k \log k)$ columns for a rank-k approximation, the log factor is not necessary for certain types of code matrices. (v) Fast multiplication is possible with structured code matrices, so fast approximations can be achieved for general dense input matrices. (vi) Under certain conditions, the approximation errors can be better and the singular values obtained can be more accurate, than those obtained using Gaussian random matrices and other structured random matrices. (vii) For least squares regression problem $\min \|Ax - b\|_2$ where $A \in \mathbb{R}^{n \times d}$, the $(1+\epsilon)$ relative error approximation can be achieved with $O(d \log(1/\delta)/\epsilon)$ samples, with probability $1 - \delta$, when certain code matrices are used.

***Index Terms*—Error correcting codes, low rank approximation, matrix decomposition, randomized algorithms.**

I. Introduction

MANY scientific computations, signal processing, data analysis and machine learning applications, lead to large dimensional matrices that can be well approximated by a low dimensional (low rank) basis [35], [52], [26], [18]. It is more efficient to solve such computational problems by first transforming these high dimensional matrices into a low dimensional space, while preserving the invariant subspace that captures the essential information of the matrix. Low-rank matrix approximation is an integral component of tools such as principal component analysis (PCA) [30], as well as is an important instrument used in many applications like computer vision (e.g., face recognition) [48], signal processing (e.g., adaptive beamforming) [40], recommender systems [19], information retrieval and latent semantic indexing [7], [6], web search modeling [31], DNA microarray data [3], [41] and text mining, to name a few examples. Several algorithms have been proposed in the literature for finding low rank approximations of matrices [35], [52], [26], [18], [11], [27]. Recently, research focussed on developing techniques that use randomization for computing low rank approximations and decompositions of such large matrices [26], [45], [33], [50], [39], [32], [12]. It is found that randomness provides an effective way to construct low dimensional bases with high reliability and computational efficiency. Similar ideas based on random sampling have been proposed in the recent literature for solving least squares (ℓ_2) linear regression problems [20], [45], [43], [12], [39], [21], [13].

Randomization techniques for matrix approximations aim to compute a basis that approximately spans the range of an $m \times n$ input matrix A, by sampling the matrix A using random matrices, e.g. i.i.d Gaussian [26]. This task is accomplished by first forming the matrix-matrix product $Y = A\Omega$, where Ω is an $n \times \ell$ random matrix of smaller dimension $\ell \ll \{m, n\}$, and then computing the orthonormal basis of $Y = QR$ that identifies the range of the reduced matrix Y. It can be shown that $A \approx QQ^\top A$ with high probability. Recently, it has been observed that structured random matrices, like subsampled random Fourier transform (SRFT) and Hadamard transform (SRHT) matrices can also be used in place of fully random matrices [50], [32], [39], [47]. This paper demonstrates how matrices from error correcting codes can be a good choice for computing such low rank approximations.

The input matrices whose low rank approximation is to be computed usually have very large dimensions (e.g., in the order of $10^6 - 10^9$ [26], [51]). In order to form a Gaussian (a fully) random matrix that samples the input matrix, we need to generate a large quantity of random numbers. This could be a serious practical issue (in terms of time complexity and storage). This issue can be addressed by using the structured random matrices, like SRFT and SRHT matrices. An important practical advantage of using these structured random matrices is that their structure allows the computation of matrix-matrix product at a cost of $O(mn \log_2 \ell)$, making the algorithms fast (also known as fast transforms) for general dense input matrices. However, with these matrices mixing of columns might not be as uniform, and there is potential loss in the accuracy.

Another major drawback with fast transforms is that for parallel and distributed applications, particularly when the

S. Ubaru and Y. Saad are with the Department of Computer Science and Engineering, University of Minnesota, Twin Cities, MN USA (e-mail: ubaru001@umn.edu, saad@cs.umn.edu). A. Mazumdar is with Department of Electrical and Computer Engineering, University of Minnesota, Twin Cities, MN USA (e-mail: arya@umn.edu). This work was supported by NSF under grant NSF/CCF-1318597 (S. Ubaru and Y. Saad) and NSF/CCF-1318093 (A. Mazumdar). A preliminary version of this work has appeared as conference proceedings in the 32nd International Conference on Machine Learning, 2015 [49]. Manuscript submitted 29th July, 2015.

input matrix is sparse and/or its columns are distributively stored, it is found that FFT-like algorithms are significantly slower due to communication issues or other machine related issues (machines are optimized for matrix-vector operations) [51]. Also for a rank-k approximation, these matrices require sampling $\ell = O(k \log k)$ columns. Other practical issues arise such as: the Fourier Transform matrices require handling complex numbers and the Hadamard matrices exist only for the sizes which are in powers of 2. All these drawbacks can be overcome if the code matrices presented in this paper are used for sampling the input matrices.

In digital communication, information is encoded (by adding redundancy) to (predominantly binary) vectors or codewords, that are then transmitted over a noisy channel [14]. These codewords are required to be far apart in terms of some distance metric for noise-resilience. Coding schemes usually generate codewords that maintain a fixed minimum Hamming distance between each other, hence they are widespread and act like random vectors. We can define probability measures for matrices formed by stacking up these codewords (see section II-B for details). The idea is to use subsampled versions of these code matrices as sampling matrices in the randomized techniques for matrix approximations.

The idea of using code matrices for such applications is not new in the literature. A class of dual BCH code matrices were used in [2], [32] for Fast Johnson-Lindenstrauss Transform (FJLT) to perform fast dimensionality reduction of vectors. Code matrices have also been used in applications of sparse signal recovery, such as compressed sensing [4] and group testing [22], [37]. For matrix approximations, it is important to show that the sampling matrices used can approximately preserve the geometry of an entire subspace of vectors. In section VI-C, we show that the subsampled code matrices with certain mild properties preserve the geometry of vector subspaces with high probability. Similar to Fourier and Hadamard sampling matrices, fast multiplication is possible with code matrices from certain class of codes due to their structure (see section V for details). Hence, fast approximations can be achieved for general dense input matrices, since the matrix-matrix product can be computed in $O(mn \log_2 \ell)$ cost with these code matrices.

Importantly, the shortcomings of SRFT/SRHT matrices in parallel and distributed environments can be overcome by using code matrices and treating them as dense transforms (details in sections V, VIII). In addition, for certain subsampled code matrices, the logarithmic factor in the number of samples is not required (see section VI-E for an explanation). This is a significant theoretical result that shows order optimality can be achieved in the number of samples required with deterministic matrices. Similar improvements were posed as an open problem in [17] and in [39]. In the context of sparse approximations such improvements appear as main results in many places, see Table 1 of [5].

Randomized approximation algorithms are used to approximately solve overdetermined least squares regression problem faster [20], [45], [21], [13]. Here, we are given a matrix $A \in \mathbb{R}^{n \times d}$ and a vector $b \in \mathbb{R}^n$, with $n \gg d$. The goal is to solve the least squares regression problem $\min_x \|Ax - b\|_2$ faster, (where $\|.\|_2$ is ℓ_2 norm) and output a vector x' such that, with high probability,

$$\|Ax' - b\|_2 \leq (1 + \epsilon)\|A\hat{x} - b\|_2,$$

where $\hat{x}$ is the ℓ_2 minimizer given by the Moore-Penrose pseudo inverse of A, i.e., $\hat{x} = A^\dagger b$ [24]. For details on the applications where we encounter such extremely overdetermined linear system of equations, we refer to [51]. The idea of randomized approximation [20] is to use a sampling matrix to reduce the dimensions of A and b, and then solve the smaller problem to obtain x'. In section VII, we show how the code matrices can be used as the sampling matrix in this least squares regression problem.

The organization of the rest of this paper is as follows: Section II gives the notation used, the problem set up and a brief introduction to error correcting coding techniques. Section III discusses the construction of the subsampled code matrices and the intuition behind the construction. The algorithm of the present paper is described in section IV and the computational cost of the algorithm is discussed in section V. Section VI discusses the error analysis of the algorithm, by first providing the deterministic error bounds for the algorithm. To use these deterministic error bounds, it is required to show that the subsampled code matrices preserve the geometry of an entire subspace of vectors with high probability. This is shown in section VI-C. The bounds for the approximation errors and the singular values obtained by the algorithm are derived in section VI-D. In section VII, we extend the framework to linear least squares (ℓ_2) regression problem and in section VIII, we discuss the choice of error correcting codes for different types of input matrices and computational environments. Section IX illustrates the performance of code matrices via several numerical experiments.

II. Preliminaries

First, we present some of the notations used and give a brief description of error correcting codes used in communication systems and information theory.

A. Notation and Problem Formulation

Throughout the paper, $\|\cdot\|_2$ refers to the ℓ_2 or spectral norm. We use $\|\cdot\|_F$ for the Frobenius norm. The singular value decomposition (SVD) of a matrix A is denoted by $A = U\Sigma V^\top$ and the singular values by $\sigma_j(A)$. We use e_j for the jth standard basis vector. Given a random subset T of indices in $\{1, \ldots, 2^r\}$ with size n and $r \geq \lceil \log_2 n \rceil$, we define a restriction (sampling) operator $S_T : \mathbb{R}^{2^r} \to \mathbb{R}^T$ given by

$$(S_T \boldsymbol{x})(j) = x_j,\ j \in T.$$

A Rademacher random variable takes values ± 1 with equal probability. We write ε for a Rademacher variable.

In low rank approximation methods, we compute an orthonormal basis that approximately spans the range of an $m \times n$ input matrix A. That is, a matrix Q having orthonormal columns such that $A \approx QQ^\top A$. The basis matrix Q must contain as few columns as possible, but it needs to be an

accurate approximation of the input matrix. I.e., we seek a matrix Q with k orthonormal columns such that,

$$\|A - QQ^\top A\|_\xi \leq \epsilon, \tag{1}$$

for a positive error tolerance ϵ and an integer $\xi \geq 2$.

The best rank-k approximation of A with respect to both Frobenius and spectral norm is given by the Eckart-Young theorem [23], and it is $\hat{A}_k = U_k \Sigma_k V_k^\top$, where U_k and V_k are the k-dominant left and right singular vectors of A, respectively and diagonal Σ_k contains the top k singular values of A. So, the optimal Q in (1) will be U_k for $\xi \in \{2, F\}$.

B. Error Correcting Codes

In communication systems, data are transmitted from a source (transmitter) to a destination (receiver) through physical channels. These channels are usually noisy, causing errors in the data received. In order to facilitate detection and correction of these errors in the receiver, error correcting codes are used [34]. A block of information (data) symbols are encoded into a binary vector[1], also called a codeword. Error correcting coding methods check the correctness of the codeword received. The set of codewords corresponding to a set of data vectors (or symbols) that can possibly be transmitted is called the *code*. As per our definition a code $\mathcal{C}$ is a subset of $\mathbb{F}_2^\ell$, ℓ being an integer.

A code is said to be linear when adding two codewords of the code coordinate-wise using modulo-2 arithmetic results in a third codeword of the code. Usually a linear code $\mathcal{C}$ is represented by the tuple $[\ell, r]$, where ℓ represents the codeword length and $r = \log_2 |\mathcal{C}|$ is the number of information bits that can be encoded by the code. There are $\ell - r$ redundant bits in the codeword, which are sometimes called parity check bits, generated from messages using an appropriate rule. It is not necessary for a codeword to have the corresponding information bits as r of its coordinates, but the information must be uniquely recoverable from the codeword.

It is perhaps obvious that a linear code $\mathcal{C}$ is a linear subspace of dimension r in the vector space $\mathbb{F}_2^\ell$. The basis of $\mathcal{C}$ can be written as the rows of a matrix, which is known as the generator matrix of the code. The size of the generator matrix G is $r \times \ell$, and for any information vector $\boldsymbol{m} \in \mathbb{F}_2^r$, the corresponding codeword is found by the following linear map:

$$\boldsymbol{c} = \boldsymbol{m}G.$$

Note that all the arithmetic operations above are over the binary field $\mathbb{F}_2$.

To encode r bits, we must have 2^r unique codewords. Then, we may form a matrix of size $2^r \times \ell$ by stacking up all codewords that are formed by the generator matrix of a given linear coding scheme,

$$\underbrace{C}_{2^r \times \ell} = \underbrace{M}_{2^r \times r} \underbrace{G}_{r \times \ell}. \tag{2}$$

For a given tuple $[\ell, r]$, different error correcting coding schemes have different generator matrices and the resulting codes have different properties. For example, for any two integers t and q, a BCH code [9] has length $\ell = 2^q - 1$ and dimension $r = 2^q - 1 - tq$. Any two codewords in this BCH code maintain a minimum (Hamming) distance of at least $2t + 1$ between them. The pairwise minimum distance is an important parameter of a code and is called just the distance of the code.

As a linear code $\mathcal{C}$ is a subspace of a vector space, the null space $\mathcal{C}^\perp$ of the code is another well defined subspace. This is called the *dual* of the code. For example, the dual of the $[2^q - 1, 2^q - 1 - tq]$-BCH code is a code with length $2^q - 1$, dimension tq and minimum distance at least $2^{q-1} - (t-1)2^{q/2}$. The minimum distance of the dual code is called the *dual distance* of the code.

Depending on the coding schemes used, the codeword matrix C will have a variety of favorable properties, e.g., low coherence which is useful in compressed sensing [36], [4]. Since the codewords need to be far apart, they show some properties of random vectors. We can define probability measures for codes generated from a given coding scheme. If $\mathcal{C} \subset \{0,1\}^\ell$ is an $\mathbb{F}_2$-linear code whose dual $\mathcal{C}^\perp$ has a minimum distance above k (dual distance $> k$), then the code matrix is an *orthogonal array* of strength k [16]. This means, in such a code $\mathcal{C}$, for any k entries of each codeword $\boldsymbol{c}$ say $\mathbf{c}' = \{c_{i_1}, c_{i_2}, \ldots, c_{i_k}\}$ and for any k bit binary string α, we have

$$\mathbf{Pr}[\mathbf{c}' = \alpha] = 2^{-k}.$$

This is called the k-wise independence property of codes. We will use this property of codes in our theoretical analysis (see section VI for details).

The codeword matrix C has 2^r codewords each of length ℓ (a $2^r \times \ell$ matrix), i.e., a set of 2^r vectors in $\{0,1\}^\ell$. Given a codeword $\mathbf{c} \in \mathcal{C}$, let us map it to a vector $\phi \in \mathbb{R}^\ell$ by setting $1 \longrightarrow \frac{-1}{\sqrt{2^r}}$ and $0 \longrightarrow \frac{1}{\sqrt{2^r}}$. In this way, a binary code $\mathcal{C}$ gives rise to a code matrix $\Phi = (\phi_1, \ldots, \phi_{2^r})^\top$. Such a mapping is called binary phase-shift keying (BPSK) and appeared in the context of sparse recovery (e.g., p. 66 [36]). For codes with dual distance ≥ 3, this code matrix Φ will have orthonormal columns. In section VI-C, we will show that these code matrices with certain mild properties can preserve the geometry of vector subspaces with high probability. Hence, in the randomized techniques for matrix approximations, we can use a subsampled and scaled version of this matrix Φ to sample a given input matrix and find the active subspaces of the matrix.

III. Construction of Subsampled Code Matrix

For an input matrix A of size $m \times n$, and a target rank k, we choose $r \geq \lceil \log_2 n \rceil$ as the dimension of the code (length of the message vector) and $\ell > k$ as the length of the code. The value of ℓ will depend on the coding scheme used, particularly on the dual distance of the code (details in section VI-C). We consider an $[\ell, r]$-linear coding scheme and form the sampling matrix as follows: we draw the sampling test matrix say Ω as

$$\Omega = \sqrt{\frac{2^r}{\ell}} DS\Phi, \tag{3}$$

[1]Here, and in the rest of the text, we are considering only binary codes. In practice, codes over other alphabets are also quite common.

where

- D is a random $n \times n$ diagonal matrix whose entries are independent random signs, i.e., random variables uniformly distributed on $\{\pm 1\}$.
- S is the uniformly random downsampler, an $n \times 2^r$ matrix whose n rows are randomly selected from a $2^r \times 2^r$ identity matrix.
- Φ is the $2^r \times \ell$ code matrix, generated using an $[\ell, r]$-linear coding scheme, with BPSK mapping and scaled by $2^{-r/2}$ such that all columns have unit norm.

A. Intuition

The design of a subsampled code matrix is similar to the design of SRFT and SRHT matrices. The intuition for using such a design is well established in [47], [26]. The matrix Φ has entries with magnitude $\pm 2^{-r/2}$ and has orthonormal columns when a coding scheme with dual distance of the codes is ≥ 3 is used.

The scaling $\sqrt{\frac{2^r}{\ell}}$ is used to make the energy of the sampling matrix equal to unity, that is, to make the rows of Ω unit vectors. The purpose of multiplying by D is to flatten out input vectors, we refer to [47] for further details. For a fixed unit vector $\boldsymbol{x}$, the first component of $\boldsymbol{x}^\top DS\Phi$ is given by $(\boldsymbol{x}^\top DS\Phi)_1 = \sum_{j=1}^{n} x_j \varepsilon_j \phi_{j'1}$, where $\phi_{j'1}$ are components of the code matrix Φ, the index j' depends on the downsampler S and ε_j is the Rademacher variable from D. This sum clearly has zero mean and since entries of Φ have magnitude $2^{-r/2}$, the variance of the sum is 2^{-r}. The Hoeffding inequality [28] shows that

$$\mathbb{P}\{|(\boldsymbol{x}^\top DS\Phi)_1| \geq \tilde{t}\} \leq 2e^{-2^r \tilde{t}^2/2}.$$

That is, the magnitude of the first component of $\boldsymbol{x}^\top DS\Phi$ is about $2^{-r/2}$. Similarly, the argument holds for the remaining entries. Therefore, it is unlikely that any one of the ℓ components of $\boldsymbol{x}^\top DS\Phi$ is larger than $\sqrt{2\log(2\ell)/2^r}$, (the failure probability is ℓ^{-1}).

The downsampler S is a formal way of saying, if $n < 2^r$, we choose n out of 2^r possible codewords to form the sampling matrix Ω. Uniform downsampling is used in the theoretical analysis to get an upper bound for the singular values of Ω (see section VI-D). In practice, we choose n numbers between 1 to 2^r, use the binary representation of these numbers as the message vectors (form M) and use the generator matrix G of the coding scheme selected to form the sampling matrix Ω using (2) and BPSK mapping. For dense input matrices, it is advantageous to choose these numbers (message vectors) to be 1 to $2^{\lceil \log_2 n \rceil}$, to exploit the availability of fast multiplication (see details in section V).

IV. Algorithm

We use the same prototype algorithm as discussed in [26] for the low rank approximation and decomposition of an input matrix A. The subsampled code matrices given in (3), generated from a chosen coding scheme is used as the sampling test matrix. The algorithm is as follows:

Algorithm 1 Prototype Algorithm

Input: An $m \times n$ matrix A, a target rank k.
Output: Rank-k factors U, Σ, and V in an approximate SVD $A \approx U\Sigma V^\top$.
1. Form an $n \times \ell$ subsampled code matrix Ω, as described in Section III and (3), using an $[\ell, r]-$linear coding scheme, where $\ell > k$ and $r \geq \lceil \log_2 n \rceil$.
2. Form the $m \times \ell$ sample matrix $Y = A\Omega$.
3. Form an $m \times \ell$ orthonormal matrix Q such that $Y = QR$.
4. Form the $\ell \times n$ matrix $B = Q^\top A$.
5. Compute the SVD of the small matrix $B = \hat{U}\Sigma V^\top$.
6. Form the matrix $U = Q\hat{U}$.

The prototype algorithm requires only two passes over the input matrix (single pass algorithms can also be developed [26, §5.5]), as opposed to $O(k)$ passes required for classical algorithms. This is particularly significant when the input matrix is very large to fit in fast memory (RAM) or when the matrices are streamed [26]. The randomized techniques allow us to reorganize the calculations required to exploit the input matrix properties and the modern computer architecture more efficiently. The algorithm is also well suited for implementation in parallel and distributed environments [51]. For more details on all the advantages of randomized methods over classical techniques, we refer to [45], [50], [26].

Several algorithms have been developed in the literature which build on the above prototype algorithm. An important requirement (rather a drawback) of the prototype algorithm is that, to obtain a good approximation, the algorithm requires the singular values of the input matrix to decay rapidly [26]. Methods such as randomized power method [26], [42] and randomized subspace iteration [26], [25] have been proposed to improve the performance (accuracy) of the prototype algorithm, particularly when the singular values of the input matrix decay slowly. In these methods, step 2 in algorithm 1 is replaced by $Y = (AA^\top)^{q'} A\Omega$, where q' is a small integer. However, these algorithms require $2(q'+1)$ passes over A. Use of structured random matrices like SRFT and SRHT are proposed for faster computation of the matrix product $Y = A\Omega$ [50], [32], for dense input matrices. Algorithm 1 can also be modified to obtain the eigenvalue decompositions of square input matrices [26]. In all these modified algorithms, we can replace the random sampling matrix by our subsampled code matrix. For the analysis in the following sections, we shall consider the prototype algorithm 1.

V. Computational Cost

The key advantage of using structured random matrices (SRFT or SRHT) in the randomized algorithms is that, for a general dense matrix we can compute the matrix-matrix product $Y = A\Omega$ in $O(mn \log_2 \ell)$ time, exploiting the structure of Fourier/Hadamard matrices [45], [50], [39], [32], [26]. The idea of fast multiplications was inspired by articles on Fast Johnson-Lindenstrauss Transform (FJLT) [1], [2] where it is shown that matrix-vector products with such structured matrices can be achieved in $O(n \log_2 \ell)$ time. Interestingly,

Ailon and Liberty [2] give dual BCH code matrices and Hadamard matrices (that are actually a special codes called 1st order Reed-Muller codes) as examples.

Many, if not most of the structured codes can be decoded using the Fast Fourier Transform (FFT) [8]. The corresponding $2^r \times \ell$ code matrix Φ of such structured codes (after BPSK mapping) will have every column of Φ equal to some column of a $2^r \times 2^r$ Hadamard matrix, see definition 2.2 in [2]. Hence, for a general dense matrix in RAM, the matrix-matrix product $Y = A\Omega$ with these structure code matrices can be achieved in $O(mn \log_2 \ell)$ time using the technique described in [2], [32]. If $n < 2^r$, we choose the top $2^{\lceil \log_2 n \rceil}$ codewords of Φ as the rows of Ω such that the columns of Ω are some columns of a $2^{\lceil \log_2 n \rceil} \times 2^{\lceil \log_2 n \rceil}$ Hadamard matrix.

Fast multiplications are possible with matrices from another class of codes known as cyclic codes. In cyclic codes, a circular shift of a codeword results in another codeword of that code. So a $2^r \times \ell$ code matrix Φ generated using an $[\ell, r]$-cyclic code scheme will consist of $2^r/\ell$ blocks of circulant matrices of size $\ell \times \ell$ stacked up when appropriately rearranged. It is known that the matrix-vector products with circulant matrices can be performed in $O(\ell \log_2 \ell)$ operations via FFT [24]. So for a general dense input matrix, the matrix-matrix product $Y = A\Omega$ with such code matrices can also be achieved in $O(mn \log_2 \ell)$ time.

The remaining steps (steps $3-6$) of the algorithm can be computed in $O((m+n)k^2)$ time using the row extraction method described in [26]. Therefore, for a general dense input matrix in RAM, the total computational cost of algorithm 1 using a structured code matrix is $O(mn \log_2 \ell + (m+n)k^2)$.

For sparse input matrices or when the columns of A are distributively stored, we can choose codewords at random from a desired code (as described earlier) making Ω unstructured and $Y = A\Omega$ a dense transform, similar to a random sampling matrix. The computational cost of the algorithm for such cases is $O(nz(A)\ell + (m+n)k^2)$, where $nz(A)$ is the number of nonzero entries in the input matrix A. We will see that, for code matrices with certain properties, $\ell = O(k/\epsilon)$ which is advantageous in these cases (compared to SRFT/SRHT which require $\ell = O(k \log k/\epsilon)$). Additional details of the choice of the code matrix for different types of input matrices and computational environments are given in section VIII.

VI. Analysis

This section discusses the performance analysis of the subsampled code matrices as sampling matrices in algorithm 1. First, we give the deterministic error bounds for the algorithm for a given sampling matrix Ω. Then, we show how code matrices preserve the geometry of an entire subspace of vectors by establishing connection to Johnson Lindenstrauss Transforms (JLT) and random sign matrices via the k-wise independence property of the codes. Finally, we give the bounds for the approximation error and the singular values obtained from the algorithm.

A. Setup

Let A be an $m \times n$ input matrix with SVD given by $A = U\Sigma V^\top$, and partition its SVD as follows

$$A = U \begin{array}{c} \begin{matrix} k & n-k \end{matrix} \\ \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} \end{array} \quad \begin{array}{c} \begin{matrix} n \end{matrix} \\ \begin{bmatrix} V_1^\top \\ V_2^\top \end{bmatrix} \end{array} \begin{array}{l} k \\ n-k \end{array}. \tag{4}$$

Let Ω be the $n \times \ell$ test (sampling) matrix, where ℓ is the number of samples. Consider the matrices

$$\Omega_1 = V_1^\top \Omega \quad \text{and} \quad \Omega_2 = V_2^\top \Omega. \tag{5}$$

The objective of any low rank approximation algorithm is to try and approximate the subspace which spans the top k left singular vectors of A. The test matrix Ω is said to preserve the geometry of an entire subspace of vectors, if for any orthonormal matrix V, a matrix of the form $V^\top \Omega$ is well conditioned [26].

B. Deterministic Error bounds

Algorithm 1 constructs an orthonormal basis Q for the range of Y, and the goal is to quantify how well this basis captures the action of the input matrix A. Let $QQ^\top = P_Y$ where P_Y is the unique orthogonal projector with range(P_Y)=range(Y). If Y is full rank, we can express the projector as : $P_Y = Y(Y^\top Y)^{-1}Y^\top$. We seek to find an upper bound for the approximation error given by, for $\xi \in \{2, F\}$

$$\|A - QQ^\top A\|_\xi = \|(I - P_Y)A\|_\xi.$$

The deterministic upper bound for the approximation error for algorithm 1 is given in [26]. We restate theorem 9.1 in [26] below:

Theorem 1 (Deterministic error bound): Let A be $m \times n$ matrix with singular value decomposition given by $A = U\Sigma V^\top$, and fixed $k \geq 0$. Choose a test matrix Ω and construct the sample matrix $Y = A\Omega$. Partition Σ as in (4), and define Ω_1 and Ω_2 via (5). Assuming that Ω_1 is full row rank, the approximation error satisfies for $\xi \in \{2, F\}$

$$\|(I - P_Y)A\|_\xi^2 \leq \|\Sigma_2\|_\xi^2 + \|\Sigma_2 \Omega_2 \Omega_1^\dagger\|_\xi^2. \tag{6}$$

An elaborate proof for the above theorem can be found in [26]. Using the submultiplicative property of the spectral and Frobenius norms, and the Eckart-Young theorem mentioned earlier, equation (6) can be simplified to

$$\|A - QQ^\top A\|_\xi \leq \|A - \hat{A}_k\|_\xi \sqrt{1 + \|\Omega_2\|_2^2 \|\Omega_1^\dagger\|_2^2}. \tag{7}$$

Recently Ming Gu [25], developed deterministic lower bounds for the singular values obtained from randomized algorithms, particularly for the power method [26]. Given below is the modified version of Theorem 4.3 in [25] for algorithm 1.

Theorem 2 (Deterministic singular value bounds): Let $A = U\Sigma V^\top$ be the SVD of A, for a fixed k, and let $V^\top \Omega$ be partitioned as in (5). Assuming that Ω_1 is full row rank, then algorithm 1 must satisfy for $j = 1, \ldots, k$:

$$\sigma_j \geq \sigma_j(A_k) \geq \frac{\sigma_j}{\sqrt{1 + \|\Omega_2\|_2^2 \|\Omega_1^\dagger\|_2^2 \left(\frac{\sigma_{k+1}}{\sigma_j}\right)^2}} \tag{8}$$

where σ_j are the jth singular value of A and A_k is the rank-k approximation obtained by our algorithm.
The proof for the above theorem can be seen in [25]. For a given sampling matrix Ω, the major challenge is to show that Ω_1 is indeed full rank. That is, we need to show that for any orthonormal matrix V, with high probability $V^\top\Omega$ is well conditioned.

C. Subsampled Code Matrices Preserve Geometry

Recall from section III the construction of the 'tall and thin' $n \times \ell$ subsampled error correcting code matrices Ω. One of the critical facts to show is that these matrices approximately preserve the geometry of an entire subspace of vectors. This will imply that Ω_1 will be full rank and we can use the deterministic bounds mentioned above for our analysis. To prove this, we establish connections between the properties of code matrices and two important results existing in the literature. The first connection is to the well known Johnson-Lindenstrauss Transform (JLT) [29] and the second is with random sign matrices. Both these connections depend on the k-wise independence property of the code matrices.

1) Connection to Johnson-Lindenstrauss Transform: One of the primary results developed in the randomized matrix algorithms literature was establishing the relation between the Johnson-Lindenstrauss Transform (JLT) and preserving the geometry of subspaces [45]. We first give the definition of JLT and then state this important result. We will then show that code matrices under certain mild conditions satisfy JLT.

Definition 1: A matrix $\Omega \in \mathbb{R}^{n\times\ell}$ is Johnson-Lindenstrauss Transform with parameters ϵ, δ or JLT(ϵ, δ) for any $0 < \epsilon, \delta < 1$, if for any vector $v \in \mathbb{R}^n$, it holds

$$(1-\epsilon)\|v\|_2^2 \leq \|v^\top\Omega\|_2^2 \leq (1+\epsilon)\|v\|_2^2$$

with probability $1-\delta$, under certain conditions on ℓ which will depend on ϵ, δ and the reduced dimension desired.
So if the sampling matrix Ω is JLT, it preserves the distance of any vector v whose dimensionality reduction we seek. Sarlos [45] gave the important relation between JLT and random matrix sampling (also known as subspace embedding). The following lemma which is corollary 11 in [45] gives this relation.

Lemma 3: Let $0 < \epsilon, \delta < 1$ and f be some function. If Ω is a JLT from $\mathbb{R}^n$ to $O(k\log(k/\epsilon)/\epsilon^2.f(\delta))$, then for an orthonormal matrix $V \in \mathbb{R}^{n\times k}, n \geq k$ we have

$$\mathbf{Pr}(\forall \in [1..k] : |1 - \sigma_i(V^\top\Omega)| \leq \epsilon) \geq 1-\delta$$

The above lemma shows that, if the sampling matrix Ω is JLT and $\ell = O(k\log(k/\epsilon))$ (choosing $f(\delta)$ close to ϵ^2), then the singular values of $V^\top\Omega$ are bounded, i.e., $V^\top\Omega$ is well conditioned with high probability. So, if our subsampled code matrix is a JLT, then it will preserve the geometry of V with high probability.

Next, we give two results that show that code matrices with certain mild properties satisfy the JLT property. The first result is by Ailon and Liberty [2], where they show a matrix Ω which is 4-wise independent will satisfy JLT. Interestingly, they give the 2 error correcting dual BCH codes as examples and also show how fast multiplication can be achieved with code matrices. A small drawback here is that the maximum entries of A need to be restricted.

The second (stronger) result is by Clarkson and Woodruff [12] (see Theorem 2.2), where they show if Ω is a $4\lceil\log(\sqrt{(2)}/\delta)\rceil$-wise independent matrix, then Ω will satisfy the JLT property. We know that a code matrix with dual distance $> k$ is k-wise independent. Thus, any error correcting code matrix with a dual distance > 4 (more than 2 error correcting ability) will preserve the geometry of and entire subspace of vectors (i.e., Ω_1 is full rank) with high probability.

2) Code matrices as random sign matrices: Any code matrix with a dual distance > 4 will preserve the geometry of V. However, we need the number of samples to be $\ell = O(k\log(k/\epsilon))$, which is similar to a subsampled Fourier or Hadamard matrix. Next, we show that $O(k)$ can be achieved in the number of samples required for code matrices if the codes satisfy certain conditions.

We know that code matrices show some properties of random matrices as the distance of the code increases. So we can treat code matrices as random sign matrices having certain probabilistic distributions. Indeed a code with dual distance above k supports k-wise independent probability measure. This property of code matrices helps us to use the following lemma given in [12, Lemma 3.4] which states,

Lemma 4: Given an integer k and $\epsilon, \delta > 0$. If Ω is $\rho(k + \log(1/\delta))$-wise independent with an absolute constant $\rho > 1$, then for an orthonormal matrix $V \in \mathbb{R}^{n\times k}$ and $\ell = O(k\log(1/\delta)/\epsilon)$, with probability at least $1-\delta$ we have

$$\|V^\top\Omega\Omega^\top V - I\|_2 \leq \epsilon.$$

Thus, a sampling matrix Ω which is $\lceil k + \log(1/\delta)\rceil$-wise independent preserves the geometry of V with number of samples (length) $\ell = O(k/\epsilon)$. Hence, a code matrix with dual distance $> \lceil k + \log(1/\delta)\rceil$ will preserve the geometry of V with $\ell = O(k)$.

Therefore, any code matrix with dual distance > 4 will preserve the geometry of V with $\ell = O(k\log(k/\epsilon))$ and if the dual distance is $> k$, then the code matrix can preserve the geometry of V with $\ell = O(k/\epsilon)$.

D. Error Bounds

The following theorem gives the approximation error bounds when the subsampled code matrix is used as a test matrix Ω in Theorem 1. The upper and lower bounds for the singular values obtained are also given.

Theorem 5 (Error bounds for code matrix): Let A be $m \times n$ matrix with singular values $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots$. Generate a subsampled code matrix Ω from a desired coding scheme as in (3) with $r \geq \lceil\log_2(n)\rceil$ as the dimension of the code. For any code matrix Ω with **dual distance** > 4 **and length** $\ell = O(k\log(k/\epsilon)/\epsilon^2.f(\delta))$, the approximation error for algorithm 1 satisfies, for $\xi \in \{2, F\}$

$$\|A - QQ^\top A\|_\xi \leq \|A - \hat{A}_k\|_\xi\sqrt{1 + \frac{(1+\eta)n}{(1-\epsilon)^2\ell}} \qquad (9)$$

for a small constant $\eta > 0$ with probability at least $1-\delta$ and for any code matrix Ω with **dual distance** $\geq \lceil k + \log(1/\delta) \rceil$ **and length** $\ell = O(k \log(1/\delta)/\epsilon)$, the approximation error satisfies

$$\|A - QQ^\top A\|_F \leq \|A - \hat{A}_k\|_F (1+\epsilon) \tag{10}$$

with failure probability δ. The bounds for the singular values obtained are:

$$\sigma_j \geq \sigma_j(A_k) \geq \frac{\sigma_j}{\sqrt{1 + \left(\frac{(1+\eta)n}{(1-\epsilon)^2 \ell}\right)\left(\frac{\sigma_{k+1}}{\sigma_j}\right)^2}} \tag{11}$$

The proof of the theorem follows from the deterministic bounds given earlier. For the approximate error bounds given in (9), we start from equation (7) in Theorem 1. The terms that depend on the choice of the test matrix Ω are $\|\Omega_2\|_2^2$ and $\|\Omega_1^\dagger\|_2^2$. We saw that the code matrix Ω preserves the geometry of the entire subspace of vectors. This also ensures that the spectral norm of $\Omega_1^\dagger$ is under control. From Lemma 3 and Lemma 3.6 in [33], we have

$$\|\Omega_1^\dagger\|_2^2 = \frac{1}{\sigma_k^2(\Omega_1)} \leq \frac{1}{(1-\epsilon)^2}.$$

We bound the spectral norm of Ω_2 as follows $\|\Omega_2\|_2^2 = \|V_2^\top \Omega\|_2^2 \leq \|V_2\|_2^2 \|\Omega\|_2^2 = \|\Omega\|_2^2 = \sigma_1^2(\Omega)$, since V_2 is an orthonormal matrix. The following two lemmas give the upper bound for the singular values of Ω. The first lemma shows that if a code has dual distance ≥ 3, the resulting code matrix Φ has orthonormal columns.

Lemma 6 (Code matrix with orthonormal columns): A code matrix Φ generated by a coding scheme which results in codes that have dual distance ≥ 3, has orthonormal columns.

Proof. If a code has dual distance 3, then the corresponding code matrix (stacked up codewords as rows) is an orthogonal array of strength 2 [16]. This means all the tuples of bits, i.e., $\{0,0\}, \{0,1\}, \{1,0\}, \{1,1\}$, appear with equal frequencies in any two columns of the codeword matrix C. As a result, the Hamming distance between any two columns of C is exactly 2^{r-1} (half the length of the column). This means after the BPSK mapping, the inner product between any two codewords will be zero. It is easy to see that the columns are unit norm as well.

This fact helps us use Lemma 3.4 from [47] which shows that randomly sampling the rows of such a code matrix results in a well-conditioned matrix and gives bounds for the singular values.

Lemma 7 (Row sampling): Let Φ be a $2^r \times \ell$ code matrix (with orthonormal columns), and let $M = 2^r . \max_{j=1,\ldots,2^r} \|e_j^\top \Phi\|_2^2$. For a positive parameter α, select the sample size

$$n \geq \alpha M \log(\ell).$$

Draw a random subset T from $\{1, \ldots, 2^r\}$ by sampling n coordinates without replacement. Then

$$\sqrt{\frac{(1-\nu)n}{2^r}} \leq \sigma_\ell(S_T \Phi) \text{ and } \sigma_1(S_T \Phi) \leq \sqrt{\frac{(1+\eta)n}{2^r}} \tag{12}$$

with failure probability at most

$$\ell . \left[\frac{e^{-\nu}}{(1-\nu)^{(1-\nu)}}\right]^{\alpha \log(\ell)} + \ell . \left[\frac{e^{\eta}}{(1+\eta)^{(1+\eta)}}\right]^{\alpha \log(\ell)},$$

where $\nu \in [0,1)$ and $\eta > 0$.
Since n is fixed and $M = \ell$ for a code matrix (all the entries of the matrix are $\pm 2^{-r/2}$), we get the condition $n \geq \alpha \ell \log(\ell)$. The parameters α, ν and η are chosen based on the inputs ℓ and n, and the failure probability accepted. The bounds on the singular values of the above lemma are proved in [47] using matrix Chernoff bounds. Since we use the scaling $\sqrt{\frac{2^r}{\ell}}$, the bounds on the singular values of the subsampled code matrix Ω will be

$$\sqrt{\frac{(1-\nu)n}{\ell}} \leq \sigma_\ell(\Omega) \text{ and } \sigma_1(\Omega) \leq \sqrt{\frac{(1+\eta)n}{\ell}}. \tag{13}$$

We substitute the above values for $\|\Omega_2\|^2$ and $\|\Omega_1^\dagger\|^2$ in (7) to get the error bounds in (9) and substitute these values in (8) of theorem 2 to get the bounds on singular values (11).

Clarkson and Woodruff [12] also give the Frobenius norm error bound for low rank approximation using k-wise independent sampling matrices. The error bound in (10) is straight from the following lemma which is a modification of theorem 4.2 in [12].

Lemma 8: If $\Omega \in \mathbb{R}^{n \times \ell}$ is a $\rho(k + \log(1/\delta))$-wise independent sampling matrix, then for $\ell = O(k \log(1/\delta)/\epsilon)$, with probability at least $1 - \delta$, we have

$$\|A - QQ^\top A\|_F \leq \|A - \hat{A}_k\|_F (1+\epsilon). \tag{14}$$

Proof of this lemma is clear from the proof of theorem 4.2 in [12]. This completes the proof of theorem 5. The two norm error bound for the second case (dual distance $> k$) will be same as in equation (9).

The upper bounds for the Frobenius and spectral norm errors (9) obtained for the code matrices are similar to the bounds obtained for the Gaussian random matrices and structured random matrices like SRFT/SRHT given in the review article by Halko et.al [26]. For the structured random matrices, $(1+\epsilon)$ optimal Frobenius norm error has been derived in [39] and [10]. We have a similar $(1+\epsilon)$ optimal Frobenius norm error obtained for code matrices with dual distance $> k$ in (10). Importantly, we show that this optimal error bound can be achieved with number of samples $\ell = O(k/\epsilon)$ as opposed to $O(k \log k/\epsilon)$ required for structured random matrices. Details on how to generate such code matrices with dual distance $> k$ and length $\ell = O(k/\epsilon)$ is given in section VIII. It is an interesting future work, to further improve the spectral norm error bounds for both cases (dual distance > 4 and $> k$) of code matrices and also improve the Frobenius norm error bound for the dual distance > 4 case.

a) Differences in the construction: An important difference between the construction of subsampled code matrices given in (3) and the construction of SRHT or SRFT given in [26], [47] is in the way these matrices are subsampled. In the case of SRHT, a Hadamard matrix of size $n \times n$ is applied to input matrix A and ℓ out of n columns are sampled at random (n must be a power of 2). In contrast, in the case

of subsampled code matrices, a $2^r \times \ell$ code matrix generated from an $[\ell, r]$-linear coding scheme is considered, and n out of 2^r codewords are chosen. The subsampling will not affect the k-wise independent property of the code matrix (or the distinctness of rows) when uniformly subsampled. This need not be true in the case of SRHT. The importance of the distinctness of rows is discussed next.

E. Logarithmic factor

A crucial advantage of the code matrices is that they have very low *coherence*. Coherence is defined as the maximum inner product between any two rows. This is in particular true when the minimum distance of the code is close to half the length. If the minimum distance of the code is d then the code matrix generated from an $[\ell, r]$-code has coherence equal to $\frac{\ell-2d}{2^r}$. For example, if we consider dual BCH code (see sec. II-B) the coherence is $\frac{2(t-1)\sqrt{\ell+1}-1}{2^r}$. Low coherence ensures near orthogonality of rows. This is a desirable property in many applications such as compressed sensing and sparse recovery.

For a rank-k approximation using subsampled Fourier or Hadamard matrices, we need to sample $O(k \log k)$ columns. This logarithmic factor emerges as a necessary condition in the theoretical proof (given in [47]) that shows that these matrices approximately preserve the geometry of an entire subspace of input vectors. The log factor is also necessary to tackle the worst case input matrices. The discussions in sec. 11 of [26] and sec. 3.3 of [47] give more details. In the case of certain subsampled code matrices, the log factor does not seem necessary to tackle the worst case input matrices. To see why this is true, let us consider the worst case example for orthonormal matrix V described in Remark 11.2 of [26].

An infinite family of worst case examples of the matrix V is as follows. For a fixed integer k, let $n = k^2$. Form an $n \times k$ orthonormal matrix V by regular decimation of the $n \times n$ identity matrix. That is, V is a matrix whose jth row has a unit entry in column $(j-1)/k$ when $j \equiv 1 \pmod{k}$ and is zero otherwise. This type of matrix is troublesome when DFT or Hadamard matrices are used for sampling.

Suppose that we apply $\Omega = DFR^\top$ to the matrix $V^\top$, where D is same as in (3), F is an $n \times n$ DFT or Hadamard matrix and R is $\ell \times n$ matrix that samples ℓ coordinates from n uniformly at random. We obtain a matrix $X = V^\top \Omega = WR^\top$, which consists of ℓ random columns sampled from $W = V^\top DF$. Up to scaling and modulation of columns, W consists of k copies of a $k \times k$ DFT or Hadamard matrix concatenated horizontally. To ensure that X is well conditioned (preserve geometry), we need $\sigma_k(X) > 0$. That is, we must pick at least one copy of each of the k distinct columns of W. This is the coupon collector's problem [38] in disguise and to obtain a complete set of k columns with non-negligible probability, we must draw at least $k \log(k)$ columns.

In the case of code matrices, we apply a subsampled code matrix $\Omega = DS\Phi$ to the matrix $V^\top$. We obtain $X = V^\top \Omega = V^\top DS\Phi$, which consists of k randomly selected rows of the code matrix Φ. That is, X consists of k distinct codewords of length ℓ. The code matrix has low coherence and all rows are distinct. If we use a code matrix with dual distance $> k$, then X contains k rows which are k-wise independent (near orthonormal) and $\sigma_k(X) > 0$; as a result the geometry of V is preserved and the log factor is not necessary. Thus, for the worst case scenarios we have an $O(\log k)$ factor improvement over other structured matrices. More importantly, this shows that the order optimal can be achieved with the immediate lower bound of $O(k)$ in the number of samples required with deterministic matrices.

VII. Least squares regression problem

In this section, we extend the framework to solve the least squares (ℓ_2) regression problem. As discussed in the introduction, the idea of randomized approximations is to reduce the dimensions of $A \in \mathbb{R}^{n \times d}$ and $b \in \mathbb{R}^n$ with $n \gg d$, by pre-multiplying them by a sampling matrix $\Omega \in \mathbb{R}^{n \times \ell}$, and then to solve the smaller problem quickly,

$$\min_x \|\Omega^\top Ax - \Omega^\top b\|_2. \tag{15}$$

Let the optimal solution be $x' = (\Omega^\top A)^\dagger \Omega^\top b$. Here we analyze the performance of code matrices as the sampling matrix Ω. We again use the relation between code matrices and the two existing results, i.e., JLT and random sign matrices and use the results developed by Sarlos [45], and Clarkson and Woodruff [12] for our analysis.

We know that, any code matrix with dual distance > 4 satisfies JLT. Sarlos in [45] has derived the performance of JLT matrices as sampling matrices in the ℓ_2 regression problem (15). The following theorem is a modification of theorem 12 in [45].

Theorem 9: Suppose $A \in \mathbb{R}^{n \times d}, b \in \mathbb{R}^n$. Let $\mathcal{Z} = \min_x \|Ax - b\|_2 = \|A\hat{x} - b\|_2$, where $\hat{x} = A^\dagger b$ is the minimizer. Let $0 < \epsilon, \delta < 1$ and $\Omega \in \mathbb{R}^{n \times \ell}$ be a JLT and $\tilde{\mathcal{Z}} = \min_x \|\Omega^\top (Ax - b)\|_2 = \|\Omega^\top (Ax' - b)\|_2$, where $x' = (\Omega^\top A)^\dagger \Omega^\top b$. Then, with probability at least $1 - \delta$, we have

- If $\ell = O(\log(1/\delta)/\epsilon^2)$,
$$\tilde{\mathcal{Z}} \leq (1 + \epsilon)\mathcal{Z}. \tag{16}$$
- If $\ell = O(d \log d. \log(1/\delta)/\epsilon)$,
$$\|Ax' - b\|_2 \leq (1 + \epsilon)\mathcal{Z}. \tag{17}$$
- If $\ell = O(d \log d. \log(1/\delta)/\epsilon^2)$,
$$\|\hat{x} - x'\|_2 \leq \frac{\epsilon}{\sigma_{\min}(A)}\mathcal{Z}. \tag{18}$$

The proof for this theorem can be seen in [45].

If $\sqrt{\|b\|_2^2 - \mathcal{Z}^2} \geq \gamma \|b\|_2$ for some $0 < \gamma \leq 1$, then we can replace the last equation (18) by

$$\|\hat{x} - x'\|_2 \leq \epsilon \left(\kappa(A)\sqrt{\gamma^{-2} - 1}\right) \|\hat{x}\|_2, \tag{19}$$

where $\kappa(A)$ is the 2 norm condition number of A. (This equation is given to be consistent with the results given in the related literature [20], [21], [51].) Thus, any code matrix with dual distance > 4 can be used as the sampling matrix in the least squares regression problem. Again, the performance of such code matrices is very similar to that of structured random

matrices (SRHT) given in [21], [10]. Fast multiplication can be used to sample dense input matrix A.

Similar to the earlier analysis, we can expect improved performance of code matrices when we treat code matrices as random sign matrices and use the k-wise independence property of codes. For this, we use the bounds derived by Clarkson and Woodruff [12] for random sign matrices. The following theorem which is a modification of Theorem 3.1 in [12] gives the upper bound for the regression problem in this case.

Theorem 10: Given $\epsilon, \delta > 0$, suppose $A \in \mathbb{R}^{n\times d}, b \in \mathbb{R}^n$ and A has rank at most k. If Ω is a $\rho(k + \log(1/\delta))$-wise independent with an absolute constant $\rho > 1$, and x' and $\hat{x}$ are solutions as defined before, then for $\ell = O(k\log(1/\delta)/\epsilon)$, with probability at least $1 - \delta$, we have

$$\|Ax' - b\|_2 \leq (1 + \epsilon)\|A\hat{x} - b\|_2.$$

This theorem shows that, if the code matrix is $\lceil k+\log(1/\delta)\rceil$-wise independent (i.e., dual distance $> \lceil k + \log(1/\delta)\rceil$), we can get $\epsilon-$approximate solution for the regression problem with $\ell = O(k\log(1/\delta)/\epsilon)$ samples. Thus, here too, we have the $\log k$ factor gain in the number of samples over other structured random matrices (SRHT) given in [21], [10].

Typically, A is full rank. So in this case, we will need a code matrix with dual distance $> d$ and length of the code $\ell = O(d\log(1/\delta)/\epsilon)$. Article [51] discusses the applications where such overdetermined system of equations are encountered. In typical applications, n will be in the range of 10^6-10^9 and d in the range of 10^2-10^3 (details in [51]). In the next section, we discuss how we can generate code matrices with dual distance $> d$ and minimum length ℓ, and discuss the choice of the error correcting codes for different type of input matrices and computational environments.

VIII. Choice of error correcting codes

A. Codes with dual-distance at least $k + 1$

The requirement of k-wise independence of codewords translates to the dual distance of the code being greater than k. Since a smaller code (less number of codewords, i.e., smaller r) leads to less randomness in sampling, we would like to use the smallest code with dual distance greater than k.

One of the choices of the code can be the family of dual BCH codes. As mentioned earlier, this family has length ℓ, dimension $t\log(\ell + 1)$ and dual distance at least $2t + 1$. Hence, to guarantee dual distance at least k, the size of the code must be $2^{\frac{k\log(\ell+1)}{2}} = (\ell + 1)^{k/2}$. We can choose n vectors of length $\frac{k\log(\ell+1)}{2}$ and form the codewords by simply multiplying these with the generator matrix (over $\mathbb{F}_2$) to form the subsampled code matrix. Therefore, forming these code matrices will be much faster than generating $n \times \ell$ i.i.d Gaussian random matrices or random sign matrices which have k-wise independent rows.

In general, from the Gilbert-Varshamov bound of coding theory [34], it is known that linear codes of size $\sim \sum_{i=0}^{k}\binom{\ell}{i}$ exist that have length ℓ and dual distance greater than k. The construction of these code families are still randomized. However, when $k = O(\ell)$, or the dual distance is linearly growing with the code length, the above construction of dual BCH code does not hold in general. Infinite families of codes that have distance proportional to the length are called *asymptotically good codes*. The Gilbert-Varshamov bound implies that asymptotically good linear codes of size $\sim 2^{\ell h(\frac{k}{\ell})}$ exist[2], that have length ℓ and dual distance greater than k.

B. Choice of the code matrices

Depending on the types of input matrices and the computational environments, we can choose different types of code matrices that best suit the applications. If the input matrix is a general dense matrix which can be stored in the fast memory (RAM), we can choose any structured code matrix with dual distance > 4, $r = \lceil\log_2 n\rceil$ (or choose message vectors to be 1 to $2^{\lceil\log_2 n\rceil}$) and $\ell = O(k\log k)$ (eg., dual BCH codes), so that the fast multiplication technique can be exploited (the log factor will not be an issue). This will be similar to using any other structured random matrices like SRFT or SRHT. In fact, Hadamard matrices are also a class of linear codes, with variants known as Hadamard codes, Simplex codes or 1st-order Reed-Muller codes. The dual distance of Hadamard code is 3. However, with code matrices (say dual BCH codes), subsampling of columns is not required, thus reducing randomness and cost significantly.

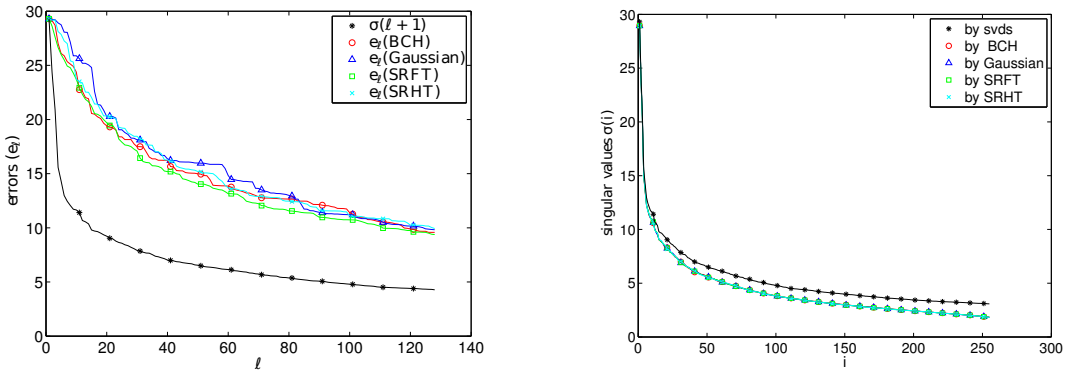
If the input matrix is sparse and/or is distributively stored, and for parallel implementation, we can choose a code matrix with dual distance $> k$ and generate them as mentioned earlier and as in section III. These code matrices are not structured and we can treat them as dense transforms (any random matrices). This will help us overcome the issues with SRFT/SRHT for sparse input matrices and in parallel and distributed applications. These code matrices are easy to generate (than i.i.d Gaussian random matrices), the log factor in the number of samples is not necessary, and so using code matrices in these applications will reduce randomness and cost significantly. If the log factor is not an issue (for smaller k), then we can choose any code matrix with dual distance > 4 and $r = \lceil\log_2 n\rceil$, and form $Y = A\Omega$ as a dense transform. These code matrices are almost deterministic and unlike SRFT/SRHT, subsampling of columns is not required.

In practice, code matrices generated by any linear coding scheme can be used in place of Gaussian random matrices. As there are many available classes of algebraic and combinatorial codes, we have a large pool of candidate matrices. In this paper we chose dual BCH codes for our numerical experiments as they particularly have low coherence, and turn out to perform quite well in practice.

IX. Numerical Experiments

The following experiments will illustrate the performance of subsampled code matrices as sampling matrices in algorithm 1. We compare the performance of dual BCH code matrices against the performance of random Gaussian matrices and subsampled Fourier transform (SRFT) matrices for different input matrices from various applications.

[2] $h(x) \equiv -x\log_2 x - (1 - x)\log_2(1 - x)$ is the binary entropy function



(A) (B)

Fig. 1. (A) The theoretical minimum $\sigma_{\ell+1}$ and approximate error as a function of the number of random samples ℓ using dual BCH code, Gaussian, SRFT and SRHT matrices as sampling matrix in algorithm 1 for input matrix `Kohonen`. (B) Estimates for top 255 singular values computed by algorithm 1 using dual BCH code, Gaussian, SRFT and SRHT matrices and the exact singular values by svds function.

Our first experiment is with a 4770×4770 matrix named Kohonen from the Pajek network (a directed graph's matrix representation), available from the UFL Sparse Matrix Collection [15]. Such graph Laplacian matrices are commonly encountered in machine learning and image processing applications. The performance of the dual BCH code matrix, Gaussian matrix, subsampled Fourier transform (SRFT) and Hadamard (SRHT) matrices are compared as sampling matrices Ω in algorithm 1. For SRHT, we have to subsample the rows as well (similar to code matrices) since the input size is not a power of 2. All experiments were implemented in matlab v8.1.

Figure 1(A) gives the actual error $e_\ell = \|A - Q^{(\ell)}(Q^{(\ell)})^\top A\|$ for each ℓ number of samples when a subsampled dual BCH code matrix, a Gaussian matrix, SRFT and SRHT matrices are used as sampling matrices in algorithm 1, respectively. The best rank-ℓ approximation error $\sigma_{\ell+1}$ is also given. Figure 1(B) plots the singular values obtained from algorithm 1, for $\ell = 255$ and different sampling matrices Ω used. The top 255 exact singular values of the matrix (available in the UFL database) are also plotted. We observe that, in practice, the performance of all four sampling matrices are similar.

Table I compares the errors e_ℓ for ℓ number of samples, obtained for a variety of input matrices from different applications when subsampled dual BCH code, Gaussian and SRFT matrices were used. All matrices were obtained from the UFL database [15]. Matrices lpi_ceria3d (4400×3576) and deter3 (21777×7647) are from linear programming problems. S80PI_n1 (4028×4028) is from an eigenvalue/model reduction problem. Delaunay (4096×4096), EPA (4772×4772) and Kohonen are graph Laplacian matrices. We observe that, for small ℓ, in the first five examples the error performance of code matrices is slightly better than that of Gaussian matrices. For higher ℓ, the error remains similar to the error for Gaussian matrices. Therefore, in practice, we can use code matrices in place of fully random (Gaussian) matrices or structured random matrices due to the advantages of code matrices over the other sampling matrices, as discussed in the previous sections. Next, we illustrate the performance of algorithm 1 with different sampling matrices in a practical application.

Eigenfaces: Eigenfaces is a popular method for face recognition that is based on Principal Component Analysis (PCA) [48], [46]. In this experiment (chosen as a verifiable comparison with results in [25]), we demonstrate the performance of randomized algorithm with different sampling matrices on face recognition. The face dataset is obtained from the AT&T Labs Cambridge database of faces [44]. There are ten different images of each of 40 distinct subjects. The size of each image is 92×112 pixels, with 256 gray levels per pixel. 200 of these faces, 5 from each individual are used as training images and the remaining 200 as test images to classify.

In the first step, we compute the principal components (dimensionality reduction) of mean shifted training image dataset using algorithm 1, with different sampling matrix Ω and different p values. Next, we project the mean-shifted images into the singular vector space using the singular vectors obtained from the first step. The projections are called feature vectors and are used to train the classifier. To classify a new face, we mean-shift the image and project it onto the singular vector space obtained in the first step, obtaining a new feature vector. The new feature vector is classified using a classifier which is trained on the feature vectors from the training images. We used the in-built MATLAB function

TABLE I
COMPARISON OF ERRORS

MATRIX	ℓ	DUAL BCH	GAUSSIAN	SRFT
LPICERIA3D	31	21.8779	23.7234	23.3688
S80PI	63	3.8148	3.8492	3.7975
DELAUNAY	63	6.3864	6.3988	6.3829
LPICERIA3D	63	15.4865	18.3882	16.3619
DETER3	127	9.2602	9.2658	9.2984
EPA	255	5.5518	5.5872	5.4096
KOHONEN	511	4.2977	4.2934	4.2610

TABLE II
COMPARISON OF THE NUMBER OF INCORRECT MATCHES

RANK	DUAL BCH		GAUSSIAN		SRFT		T-SVD
	p		p		p		
k	10	20	10	20	10	20	
10	18	13	19	15	21	18	26
20	14	11	14	12	16	12	13
30	10	08	13	08	12	09	10
40	09	08	08	07	08	10	06

`classify` for feature training and classification. We compare the performance of the dual BCH code matrix, Gaussian matrix and SRFT matrix against exact truncated SVD (T-SVD). The results are summarized in Table II. For $p = 10$ dual BCH code matrices give results that are similar to those of truncated SVD, and for rank $k < 40$, $p = 20$ our results are superior.

X. Conclusion

This paper advocated the use of matrices generated by error correcting codes as an alternative to random Gaussian or subsampled Fourier/Hadamard matrices for computing low rank matrix approximations. Among the attractive properties of the proposed approach are the numerous choices of parameters available, ease of generation, reduced randomness and cost, and the near-orthogonality of rows. We showed that any code matrix with dual distance > 4 preserves the geometry of an entire subspace of vectors. Indeed if the dual distance of the code matrix is $> k$, then the length of the code (sampling complexity) required is in $O(k)$, thus leading to an order optimal in the worst-case guaranteed sampling complexity, an improvement by a factor of $O(\log k)$ over other known deterministic matrices. We saw that fast multiplication is possible with structured code matrices, resulting in fast approximations for general dense input matrices. The implementation issues of FFT-like structured random matrices in the parallel and distributed environments can be overcome by using code matrices as sampling matrices.

It is known that Gaussian matrices perform much better in practice compared to their theoretical analysis [26]. Our code matrices (a) are almost deterministic, and (b) have ± 1 entries. Still, they perform equally well (as illustrated by experiments) compared to random real Gaussian matrices and complex Fourier matrices. Because of the availability of different families of classical codes in the rich literature of coding theory, many possible choices of code matrices are at hand. One of the contributions of this paper is to open up these options for use as structured sampling operators in low-rank approximations and least squares regression problem.

Interesting future works include improving the approximation error bounds obtained in this paper, and extending the framework of code matrices to other similar applications. The connections between code matrices and JLT and random sign matrices might lead to improved analysis in other applications of codes such as sparse recovery [4].

References

[1] N. Ailon and B. Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 557–563. ACM, 2006.

[2] N. Ailon and E. Liberty. Fast dimension reduction using rademacher series on dual bch codes. *Discrete & Computational Geometry*, 42(4):615–630, 2009.

[3] O. Alter, P. O. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences*, 97(18):10101–10106, 2000.

[4] A. Barg, A. Mazumdar, and R. Wang. Restricted isometry property of random subdictionaries. *IEEE Transactions on Information Theory*, 61(8), 2015.

[5] R. Berinde, P. Indyk, and M. Ruzic. Practical near-optimal sparse recovery in the l1 norm. In *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pages 198–205. IEEE, 2008.

[6] M. Berry, Z. Drmac, and E. Jessup. Matrices, vector spaces, and information retrieval. *SIAM Review*, 41(2):335–362, 1999.

[7] M. W. Berry, S. T. Dumais, and G. W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4):573–595, 1995.

[8] R. E. Blahut. Transform techniques for error control codes. *IBM Journal of Research and development*, 23(3):299–315, 1979.

[9] R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and control*, 3(1):68–79, 1960.

[10] C. Boutsidis and A. Gittens. Improved matrix algorithms via the subsampled randomized hadamard transform. *SIAM Journal on Matrix Analysis and Applications*, 34(3):1301–1340, 2013.

[11] C. Boutsidis, M. W. Mahoney, and P. Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 968–977. Society for Industrial and Applied Mathematics, 2009.

[12] K. L. Clarkson and D. P. Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 205–214. ACM, 2009.

[13] K. L. Clarkson and D. P. Woodruff. Low rank approximation and regression in input sparsity time. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 81–90. ACM, 2013.

[14] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[15] T. A. Davis and Y. Hu. The University of Florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):1, 2011.

[16] P. Delsarte and V. I. Levenshtein. Association schemes and coding theory. *Information Theory, IEEE Transactions on*, 44(6):2477–2504, 1998.

[17] A. Deshpande and S. Vempala. Adaptive sampling and fast low-rank matrix approximation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 292–303. Springer, 2006.

[18] P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1):158–183, 2006.

[19] P. Drineas, I. Kerenidis, and P. Raghavan. Competitive recommendation systems. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 82–90. ACM, 2002.

[20] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Sampling algorithms for l 2 regression and applications. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1127–1136. Society for Industrial and Applied Mathematics, 2006.

[21] P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlós. Faster least squares approximation. *Numerische Mathematik*, 117(2):219–249, 2011.

[22] D.-Z. Du and F. K. Hwang. Combinatorial group testing and its applications (applied mathematics). 2000.

[23] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[24] G. H. Golub and C. F. V. Loan. *Matrix Computations, 4th edition*. Johns Hopkins University Press, Baltimore, MD, 4th edition, 2013.

[25] M. Gu. Subspace iteration randomization and singular value problems. *SIAM Journal on Scientific Computing*, 37(3):A1139–A1173, 2015.

[26] N. Halko, P. Martinsson, and J. Tropp. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53(2):217–288, 2011.

[27] S. Har-Peled. Low rank matrix approximation in linear time. *arXiv preprint arXiv:1410.8802*, 2014.

[28] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[29] W. B. Johnson and J. Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

[30] I. Jolliffe. *Principal component analysis*. Wiley Online Library.

[31] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[32] E. Liberty. *Accelerated Dense Random Projections*. PhD thesis, Yale University, 2009.

[33] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51):20167–20172, 2007.

[34] F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes*, volume 16. Elsevier, 1977.
[35] I. Markovsky. *Low Rank Approximation: Algorithms, Implementation, Applications*. Springer, 2012.
[36] A. Mazumdar. *Combinatorial methods in coding theory*. PhD thesis, University of Maryland, 2011.
[37] A. Mazumdar. On almost disjunct matrices for group testing. In *Algorithms and Computation*, pages 649–658. Springer, 2012.
[38] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge International Series on Parallel Computation. Cambridge University Press, 1995.
[39] N. H. Nguyen, T. T. Do, and T. D. Tran. A fast and efficient algorithm for low-rank approximation of a matrix. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 215–224. ACM, 2009.
[40] P. Parker, P. J. Wolfe, and V. Tarokh. A signal processing application of randomized low-rank approximations. In *Statistical Signal Processing, 2005 IEEE/SP 13th Workshop on*, pages 345–350. IEEE, 2005.
[41] S. Raychaudhuri, J. M. Stuart, and R. B. Altman. Principal components analysis to summarize microarray experiments: application to sporulation time series. In *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, page 455. NIH Public Access, 2000.
[42] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.
[43] V. Rokhlin and M. Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proceedings of the National Academy of Sciences*, 105(36):13212–13217, 2008.
[44] F. S. Samaria and A. C. Harter. Parameterisation of a stochastic model for human face identification. In *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*, pages 138–142. IEEE, 1994.
[45] T. Sarlos. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 143–152. IEEE, 2006.
[46] L. Sirovich and M. Meytlis. Symmetry, probability, and recognition in face space. *Proceedings of the National Academy of Sciences*, 106(17):6895–6899, 2009.
[47] J. A. Tropp. Improved analysis of the subsampled randomized hadamard transform. *Advances in Adaptive Data Analysis*, 3(01n02):115–126, 2011.
[48] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.
[49] S. Ubaru, A. Mazumdar, and Y. Saad. Low rank approximation using error correcting coding matrices. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 702–710, 2015.
[50] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert. A fast randomized algorithm for the approximation of matrices. *Applied and Computational Harmonic Analysis*, 25(3):335–366, 2008.
[51] J. Yang, X. Meng, and M. W. Mahoney. Implementing randomized matrix algorithms in parallel and distributed environments. *arXiv preprint arXiv:1502.03032*, 2015.
[52] J. Ye. Generalized Low Rank Approximations of Matrices. *Machine Learning*, 61(1-3):167–191, 2005.