Queuing Theoretic Analysis of Replication vs Erasure Coding for General Service Time Distribution

Avishek Ghosh, Jerry Park [*]
Department of Electrical Enignnering and Computer Sciences
University of California, Berkeley
avishek_ghosh@berkeley.edu, jjhpark@berkeley.edu

Abstract

We compare the performances of a Replication based system and Erasure coded system for data centers having a very large number of servers. Classically erasure coding is used in data centers in order to provide robustness against server failure. Very recently, it is observed that erasure coding can also be used in data centers to increase the download speed of files[refer Longbo]. Basically, we exploit the redundancy in storage via appropriate load balancing algorithm and show that erasure coded system can outperform replication system. The performance analysis for the aforementioned two systems is done in [refer srikant] with exponential service time distribution. We solve the problem for any general service time distribution. Our formulation is somewhat different and more generic than that of [refer srikant], and if we plug exponential distribution in our formulation, we get the results in [refer srikant] back.

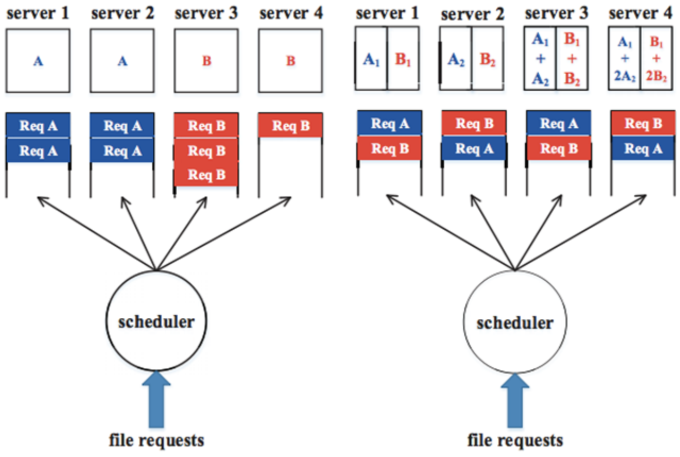
The service time distribution in data centers typically has a heavy tail and this phenomenon is empirically observed in [rashmmi ec cache]. The formulation in this paper can deal with this issue by plugging appropriate heavy tailed distribution as the service distribution (It is observed in [rashmi ec] that zipf distribution, which is a heavy tailed distribution, approximately captures the service discipline of servers in a large server system).

1 Introduction

In large data centers, massive number of servers are deployed to serve user requests. We want the entire system to be robust of individual server failures. The naive way to achieve this is via replication of the available files. There has been a significant amount of research work that focuses on the fact that if Maximum Distance Separable (MDS) codes are used instead of replication, we gain in terms of storage and repair cost. Classically MDS codes in data centers are used for this purpose only.

Also one of the major concerns in data centers is to provide user service with minimum latency, and it turns out that codes can be beneficial in this aspect too. To illustrate the point we will take an example. Consider a data center with 4 servers and having 2 files A and B. In replication scenario, assume each file is replicated twice and stored in 4 servers as shown in Figure 1. In this system, if the requested file is file A, the scheduler forks the request to server 1 and 2. Also, for now, assume that the rate of request is very low, i.e., the queue lengths are almost zero in the servers. If we assume that the service time for each server is denoted by the random variable X_r, the average time to read a particular file will be $\mathbb{E}(X_r)$.

[*] This writeup is submitted as the final Project report for the course EE-290S, Codes for Modern Age, Fall 2016



(a) $(2, 1)$ code

(b) $(4, 2)$ code

Figure 1: An example of Data center with 4 servers with replication (left) and erasure coded system (right) with $(4, 2)$ MDS code. Upon the arrival of file request for a particular file, the scheduler advances the request to that subset of the servers containing the entire (for replication) or portions (for coding) of the file.

Now, let us analyze the erasure coded system. We now consider a $(4, 2)$ MDS code, where each file is chunked into 2 parts A_1 , A_2 and B_1, B_2, and 2 other parity chunks, $A_1 + A_2$ and $A_1 + 2A_2$ (similarly for B) are used (Refer Figure 1). From the property of MDS code, it is sufficient to read any 2 out of this 4 chunks to recover the entire file. We will assume that there is no queuing effect. Later on, we relaxed this constraint. We observe that, the time to read any entire file (say file A for instance) will be $\mathbb{E}\{\max(X_c^1, X_c^2)\}$, where the service time is denoted by random variable X_c^i, and X_c^is are independent and identically distributed. Since each file is chunked into 2 halves, we can assume, $\mathbb{E}(X_c^i) \approx \frac{1}{2}\mathbb{E}(X_r)$. We use approximation instead of equality because for some distributions (such as "shift+exponential") the relation holds in approximate sense.

The gain in using MDS codes is characterized as the difference in expected download time of any arbitrary file. In this scenario, the gain, $G = \mathbb{E}(X_r) - \mathbb{E}\{\max(X_c^1, X_c^2)\}$. As a candidate distribution, we first choose exponential; i.e., $X_r \sim \exp(1)$ and $X_c^i \sim \exp(2)$. A simple calculation gives, $G = 0.25$, which is positive, and hence it is advantageous to use coding over replication. We now choose the "shift + exponential" distribution, which is found to characterize server distribution in many data centers empirically [ref sppeding up paper]. Under this discipline, $X_r = c + Y_r$, where $c > 0$ is a small constant and $Y_r \sim \exp(1)$. Similarly, $X_c^i = c + Y_c^i$, where $Y_c^i \sim \exp(2)$. The shift c can be seen as a characteristic of the server, and thus it is invariant to the file size being read. Plugging in, we get $G = 0.25 > 0$.

These intuitive observation motivates us to investigate the scenario where the arrival request is non zero, i.e., the queuing effect cannot be nullified. In this work, we will rigorously show that even with queuing effects, MDS codes outperforms replication.

1.1 Scheduling Policy:

The scheduling policy here is a randomized load balancing policy. Assume that there are a total of L servers, each of which stores a large number of different type of files. Each file is stored in n servers. Upon the read request of a particular file, the scheduler forks the request to those k servers having the shortest queue length (for an (n, k) MDS code). Also assume that there are a total of $I = \Theta(L \log L)$ files. The files are stored such that the load on each file is approximately the same The arrival process is modeled as a Poisson with rate $L\lambda$, where $\lambda \in (0, 1)$. Also each arrival requests a file uniformly at random out of I files. From the property of Poisson process, this ensures that, the arrival to a subset of n server out of L server is also Poisson with same arrival rate.

The load-balancing algorithm considered in this work is as follows: instead of considering $\binom{L}{n}$ different locations upon each job arrival, we select one subset uniformly at random and sample all the queues in that subset. Without loss of generality, we will still call our policy a randomized Batch Sampling (BS) policy.

1.2 Related Work and Our Contribution

Erasure coded system has been studied extensively in coding theory literature over the last few years. Apart for providing robustness, the delay improvement aspect of erasure coding is first shown in [refer longbo]. The delay-storage tradeoff for erasure coded system is explained in [refer Joshi] with fork-join approach. [other work] targets the problem from the point of view of joint optimization of storage cost are delay incurred. Very recently, a mean field based analysis of erasure coded system over replication codes is reported in [Srikant, Infocom], where the analysis is done assuming exponential service time distribution.

We seek to extend the result of [srikant] to a more generic setting, with general service time distribution. We used the power of d choices scheduling scheme to achieve our bound, and a simple plug in of exponential distribution yields the bound presented in [srikant]. The purpose of our analysis is to capture more realistic service distribution for servers in data centers like shift plus exponential or the heavy tailed zipf distribution. In this work, we explicitly characterized the shift plus exponential distribution.

2 Low Arrival Scenario

We will analyze the performance of replication and erasure coded system, but we will keep the same storage requirement. We compare the mean file access delay of (nk, k) and $(n, 1)$ (replication factor n) system. This is because, in erasure coded system, the file is chunked to $1/k$ th of the original file, and hence to match the storage requirements, one needs to compare n replicated system to (nk, k) erasure coded system. We will derive the mean queue length distribution for these two systems and compare the mean file access time.

In this section, we will first analyze the mean file access time when the arrival rate is very low. Since the queue length is almost zero, the mean file access delay will be $\bar{W}(n, 1) = \mathbb{E}(X_r)$, where X_r is the service time of a single queue. For the erasure coded system, in low arrival scenario, we can expect that all the queues start serving the request. The mean file access delay is given by,

$$\bar{W}(nk, k) = \mathbb{E}\left[\max_{i=1,2,\dots,k} X_c^i\right]$$

The gain,

$$G = \bar{W}(n, 1) - \bar{W}(nk, k) = \mathbb{E}(X_r) - \mathbb{E}\left[\max_{i=1,2,\dots,k} X_c^i\right] \tag{1}$$

To compute the exact gain, we will consider two special service distributions: a) Exponential and b) Shift plus Exponential. In case of exponential distribution, $X_c^i \sim \exp(k)$ and $X_r \sim \exp(1)$. We can compute the expected first order statistic,

$$\mathbb{E}\left[\max_{i=1,2,\dots,k} X_c^i\right] = \frac{H(k)}{k}$$

where, $H(k) = \sum_{l=1}^{k} \frac{1}{l}$ denotes the k-th Harmonic number.

The gain G becomes,

$$G = 1 - \frac{H(k)}{k} \tag{2}$$

which is greater than 0.

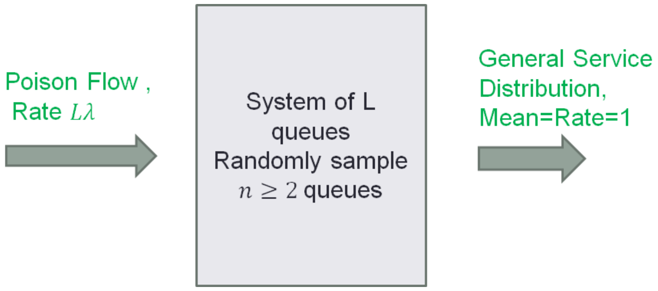

Poison Flow ,
Rate $L\lambda$
System of L queues Randomly sample $n \geq 2$ queues
General Service Distribution, Mean=Rate=1

Figure 2: Random Load Balancing Scheme, with L servers, n out of them are chosen at random and the request is advanced to the queue having the smallest queue length among the n sampled queues.

In case of shift plus exponential distribution, $\mathbb{E}(X_r) = 1 + c$ and,

$$\mathbb{E}\left[\max_{i=1,2,\ldots,k} X_c^i\right] = c + \frac{H(k)}{k}$$

Thus the gain,

$$G = 1 - \frac{H(k)}{k} \tag{3}$$

Therefore we can see for these two candidate distributions that, the gain in greater than zero, meaning that erasure coded system is advantageous in terms of expected latency.

3 Analysis with Non-zero Arrival rate

In this section we will analyze the scenario where the file request is non-zero, i.e., the queuing effects can not be ignored. To analyze the performances in these scenario, we will exploit the famous power of $d \geq 2$ choices [refer midmazer, refer bramsom]. The load balancing policy is explained in Figure 2. If requests (from a Poisson process) comes to a bank of L servers, with a general service distribution with mean unity and scheduling policy is such that, you are allowed to sample $n \geq 2$ queue and join the one with the shortest queue length. [refer bramsom] showed that under this scheduling policy, the queue length distribution vanished double exponentially when L is large (refer to Figure 2 for details). Formally in the limit $L \to \infty$, if P_M is defined as the expected queue length distribution (in steady state) having a minimum queue length of M, then,

$$P_M = \exp\left(- n^{(1+o(1))M} \right) \tag{4}$$

We will separately analyze the replication coded and erasure coded system.

3.1 Replication Based System

In the replication based system the input request is forked to n servers and it is sufficient to read only 1 server (refer to Figure 3). We are interested in the quantity, $\bar{W}(n, 1)$. Hence the system indicated here exactly matches with the scenario where one out of n queues are read, and we can apply the result of [refer bramson] in this case.

3.2 Erasure Coded System

In the erasure coded scenario, we need to compute $\bar{W}(nk, k)$, i.e., we fork the input request to nk servers and read k out of them. We employ the notion of parallel sampling for this system. The operation is as follows: we divide the nk servers into k batch n servers and read 1 out of each batch. We claim that this scheduling policy will be an upper bound on the original reading out shortest k out of total nk server policy. This is because we are selecting one queue from each batch, and it may be possible the k queues being read are not the shortest k queues. So the average reading time of this system is an upper-bound on $\bar{W}(nk, k)$. The scheduling policy is sketched in Figure 4.

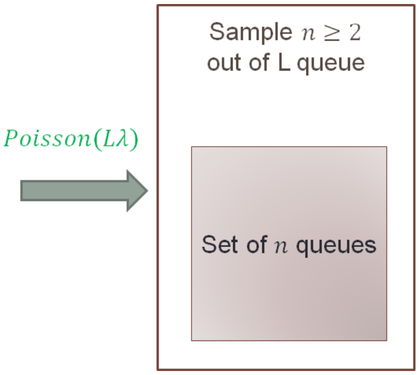


Figure 3: Scenario with Replication coding.

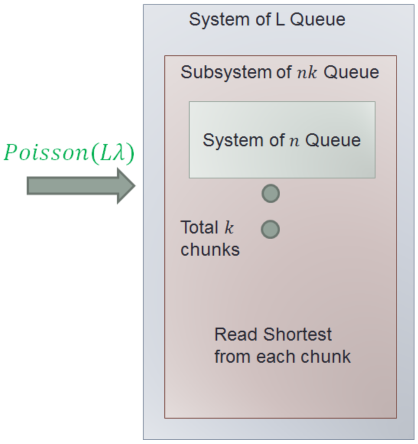


Figure 4: Scenario with Erasure coding.

4 Mean Delay Analysis

In this section, we will compare the mean delay between replication and erasure coded system. Recall that for an (n, k) MDS code, the file reading will be complete when reading from all k queues will be complete. According to the scheduling policy, the job is sent to k shortest length queues. Suppose the queue length of these k queues are $\hat{Q}_i(n, k)$, where$i = 1, 2, \ldots, k$. The service distribution of each queue is X_c^i and these random variables are iid. So, the job experiences a delay of,

$$\max_{i=1,2,\ldots,k} \sum_{j=1}^{\bar{Q}_i(n,k)+1} X_c^j \tag{5}$$

In the above equation, the plus one in the upper limit of the sum is because the time by which the reading from a particular queue is complete will be the sum of the time taken to clear the queue and the time taken to process the particular job of interest. Therefore, the mean service delay is given by,

$$\bar{W}(n, k) = \mathbb{E}\left[\max_{i=1,2,\ldots,k} \sum_{j=1}^{\bar{Q}_i(n,k)+1} X_c^j\right] \tag{6}$$

Our goal is to compare $\bar{W}(nk, k)$ with $\bar{W}(n, 1)$. We will first characterize $\bar{W}(nk, k)$.

Recall from Section 3.2 that, we will sample k batch of n queues each and fork the input request to the shortest among the mentioned batches. Let us denote the queue length of the selected queues as $\hat{Q}_i(nk, k), \forall i = 1, 2, \ldots, k$. Now, we sort the queues according to queue length distribution and obtain $\hat{Q}'_j(nk,k)$ such that, $\hat{Q}'_1(nk,k) \leq \hat{Q}'_2(nk,k) \leq \ldots \leq \hat{Q}'_k(nk,k)$. We now will follow the steps of [refer Srikant paper], to obtain the following,

$$\bar{W}(nk,k) \leq \mathbb{E}\Bigg[\sum_{j=1}^{\hat{Q}'_1(nk,k)+1} \max_{i=1,2,\ldots,k} X_c^i\Bigg] + \sum_{l=2}^{k}\mathbb{E}\Bigg[\sum_{j=\hat{Q}'_{l-2}(nk,k)+2}^{\hat{Q}'_l(nk,k)+1} \max_{i=l,l+1,\cdot,k} X_c^i\Bigg] \tag{7}$$

For iid random variables X_c^i, we define the expected first order statistic,

$$\mathbb{E}\Bigg[\max_{i=1,2,\ldots,m} X_c^i\Bigg] := S(m)$$

Since the queue length process, $\hat{Q}'_i(nk,k)$ is independent of the service random variables, X_c^i, Equation 7 can be rewritten as,

$$\bar{W}(nk,k) \leq \Bigg((1+\mathbb{E}[\hat{Q}'_1(nk,k)])S(k) + \sum_{l=2}^{k}(\mathbb{E}[\hat{Q}'_l(nk,k)] - \mathbb{E}[\hat{Q}'_{l-1}(nk,k)])S(k-l+1)\Bigg)$$

After rearranging the term in the above equation, we get

$$\bar{W}(nk,k) \leq S(k) + \sum_{l=1}^{k}(S(k-l+1) - S(k-l))\mathbb{E}[\bar{Q}_l(nk,k)] \tag{8}$$

where, $\bar{Q}_l(nk,k)$ is the steady state queue length of the corresponding queue. This is because the arrival is Poisson and we can apply the PASTA (Poisson arrival see time averages) property.

On the other hand, for the replication based system, the expected delay will be,

$$\begin{aligned} \bar{W}(n,1) &= \mathbb{E}\Bigg[\sum_{j=1}^{\hat{Q}(n,1)+1} X_r\Bigg] & (9)\\ &= (1+\mathbb{E}(\hat{Q}(n,1))\mathbb{E}(X_r) & (10)\end{aligned}$$

With Equation 7 and 10, we have the following,

$$\begin{aligned} \bar{W}(nk,k) - \bar{W}(n,1) \quad &\leq \quad \Bigg(S(k) + \sum_{l=1}^{k}(S(k-l+1) - S(k-l))\mathbb{E}[\bar{Q}_l(nk,k)]\Bigg) \\ &\quad - \Bigg(1+\mathbb{E}(\hat{Q}(n,1))\Bigg)\mathbb{E}(X_r) & (11)\end{aligned}$$

Now we use the double exponential decay property. For $n \geq 2$, with probability distribution given by Equation 4, $\mathbb{E}(\hat{Q}(n,1) \approx 0$ and $\mathbb{E}[\bar{Q}_l(nk,k)] \approx 0$. Substituting, we get,

$$\bar{W}(nk,k) - \bar{W}(n,1) \leq S(k) - \mathbb{E}(X_r) \tag{12}$$

We now compare Equation 1 and Equation 12, we observer that, in low arrival regime, $S(k) - \mathbb{E}(X_r)$ was the exact difference between replication and erasure coded system. In case of positive traffic, this becomes a lower bound on gain G. Formally,

$$G = \bar{W}(n,1) - \bar{W}(nk,k) \geq S(k) - \mathbb{E}(X_r) \tag{13}$$

4.1 Exact lower bound for Typical Distributions

We plug 2 typical distributions, a) Exponential and b) Shift plus Exponential. In the first case, $S(k) = \frac{H(k)}{k}$, so that,

$$G \geq 1 - \frac{H(k)}{k} \tag{14}$$

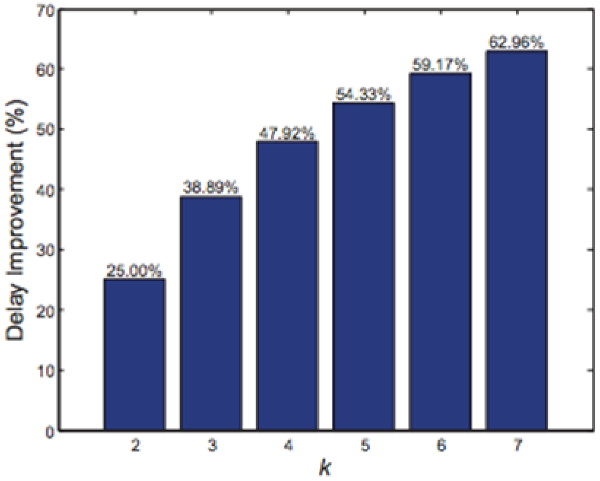


Figure 5: Variation of the lower bound of gain G with respect to k

which is the result presented in [refer Srikant]. Also, in case (b), $S(k) = c + \frac{H(k)}{k}$ and $\mathbb{E}(X_r) = 1 + c$. Therefore, in this case also, the gain

$$G \geq 1 - \frac{H(k)}{k} \tag{15}$$

Remark 1. We provide a generic lower bound on the gain G, for any general service time distribution, where the first term $S(k)$ depends on the distribution of X_c^i, and the second term is $\mathbb{E}(X_r)$, dependent on the service distribution X_r. We plugged in two typical distributions and got back the state of the art lower bound on gain G.

Remark 2. If we are working with a system with large k, the harmonic number $H(k) \approx \log k$ and the factor, $\frac{H(k)}{k} = \frac{\log k}{k} \approx 0$, so the lower bound on the gain becomes,

$$\lim_{k\to\infty} G \geq 1 - \lim_{k\to\infty} \frac{\log k}{k} = 1 \tag{16}$$

5 Simulations

We will provide simulation results in this section. Specially, we are interested in the improvement rate with respect to k. To answer this question, we plot the upper bound $1 - \frac{H(k)}{k}$ in Figure 5. We see that, the improvement is monotonic in k and with much higher k, goes to 1, as given by Equation 16. We also notice that, the rate of improvement is slow with a higher value of k, i.e., with a high k, improvement becomes marginal. We need to mention a few points here,

1. It is empirically observed that, the performance of an erasure coded system is not monotonic with k, i.e., the performance improves upto certain k and then falls back.
2. Our approach to the problem can potentially answer this question. The lower bound we achieve, is a complicated function of k and there is no reason to be monotonic. For candidate distributions like exponential and shift plus exponential, this turns out to be monotonic, but for complex distributions that characterize the behavior of servers in data centers, the bound may have an optima point. We target to characterize this in future.

6 Conclusion and Future Work

We presented the analysis of coding vs replication in general service time distribution. We obtain an upper bound on gain, and explicitly calculated the upper-bound for exponential and shift plus exponential distribution. We observe that the gain monotonically increases with k under these two distribution, which typically is not the case in practical data centers. We wish to analyze our bounds for heavy tailed distribution like zipf distribution to capture the scenario of an actual server. This is considered as our future course of action.

References

[1] Christoph Studer, Patrick Kuppinger, Graeme Pope, and Helmut Bölcskei. Recovery of Sparsely Corrupted Signals. *IEEE Transaction on Information Theory*, 58(5):3115–3130, 2012.

[2] Peter J. Rousseeuw and Annick M. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, 1987.

[3] John Wright, Alan Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust Face Recognition via Sparse Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[4] John Wright and Yi Ma. Dense Error Correction via ℓ^1 Minimization. *IEEE Transaction on Information Theory*, 56(7):3540–3560, 2010.

[5] Nam H. Nguyen and Trac D. Tran. Exact recoverability from dense corrupted observations via L1 minimization. *IEEE Transaction on Information Theory*, 59(4):2036–2058, 2013.

[6] Yudong Chen, Constantine Caramanis, and Shie Mannor. Robust Sparse Regression under Adversarial Corruption. In *30th International Conference on Machine Learning (ICML)*, 2013.

[7] Brian McWilliams, Gabriel Krummenacher, Mario Lucic, and Joachim M. Buhmann. Fast and Robust Least Squares Estimation in Corrupted Linear Models. In *28th Annual Conference on Neural Information Processing Systems (NIPS)*, 2014.

[8] Adrien-Marie Legendre (1805). On the Method of Least Squares. In (Translated from the French) D.E. Smith, editor, *A Source Book in Mathematics*, pages 576–579. New York: Dover Publications, 1959.

[9] Peter J. Rousseeuw. Least Median of Squares Regression. *Journal of the American Statistical Association*, 79(388):871–880, 1984.

[10] Peter J. Rousseeuw and Katrien Driessen. Computing LTS Regression for Large Data Sets. *Journal of Data Mining and Knowledge Discovery*, 12(1):29–45, 2006.

[11] Thomas Blumensath and Mike E. Davies. Iterative Hard Thresholding for Compressed Sensing. *Applied and Computational Harmonic Analysis*, 27(3):265–274, 2009.

[12] Prateek Jain, Ambuj Tewari, and Purushottam Kar. On Iterative Hard Thresholding Methods for High-dimensional M-Estimation. In *28th Annual Conference on Neural Information Processing Systems (NIPS)*, 2014.

[13] Yiyuan She and Art B. Owen. Outlier Detection Using Nonconvex Penalized Regression. arXiv:1006.2592 (stat.ME).

[14] Rahul Garg and Rohit Khandekar. Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property. In *26th International Conference on Machine Learning (ICML)*, 2009.

[15] Allen Y. Yang, Arvind Ganesh, Zihan Zhou, Shankar Sastry, and Yi Ma. A Review of Fast ℓ_1-Minimization Algorithms for Robust Face Recognition. CoRR abs/1007.3753, 2012.

[16] Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *The Annals of Statistics*, 28(5):1302–1338, 2000.

[17] Thomas Blumensath. Sampling and reconstructing signals from a union of linear subspaces. *IEEE Transactions on Information Theory*, 57(7):4660–4671, 2011.

[18] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. In Y. Eldar and G. Kutyniok, editors, *Compressed Sensing, Theory and Applications*, chapter 5, pages 210–268. Cambridge University Press, 2012.