Approximate Gradient Coding with Optimal Decoding

Margalit Glasgow, Mary Wootters *Member, IEEE*

***Abstract*—Gradient codes use data replication to mitigate the effect of straggling machines in distributed machine learning. *Approximate gradient codes* consider codes where the data replication factor is too low to recover the full gradient exactly. Our work is motivated by the challenge of designing approximate gradient codes that simultaneously work well in *both* the adversarial and random straggler models. We introduce novel approximate gradient codes based on expander graphs. We analyze the decoding error both for random and adversarial stragglers, when optimal decoding coefficients are used. With random stragglers, our codes achieve an error to the gradient that decays exponentially in the replication factor. With adversarial stragglers, the error is smaller than any existing code with similar performance in the random setting. We prove convergence bounds in both settings for coded gradient descent under standard assumptions. With random stragglers, our convergence rate improves upon rates obtained via black-box approaches. With adversarial stragglers, we show that gradient descent converges down to a noise floor that scales linearly with the adversarial error to the gradient. We demonstrate empirically that our codes achieve near-optimal error with random stragglers and converge faster than algorithms that do not use optimal decoding coefficients.**

***Index Terms*—Coded Computing, Gradient Coding, Optimization, Random Graphs.**

I. Introduction

CONSIDER the task of minimizing some loss function L summed over N data points $\{(x_i, y_i)\}_{i=1}^N$:

$$\min_\theta \sum_{i=1}^N L(x_i, y_i; \theta).$$

When N is large, we can parallelize the computation of the gradient of this function by distributing the data points among m worker machines, as has become common practice for large-scale machine learning problems [1]. Each machine computes the gradient of the functions available to it and returns the sum of these gradients to the parameter server. Recent work has pointed out the prevalence of *stragglers*, i.e. machines that are slow or unresponsive, which can significantly slow down the execution of distributed computing tasks such as synchronous gradient descent [2], [3]. To mitigate this effect, previous work has used a technique called *gradient coding*, which involves replicating each data point and sending it to multiple machines [4]. While this increases the computation load and storage at each machine, it has the potential to speed up convergence by allowing the parameter server to compute an exact or closer approximation to the true gradient, even in the presence of stragglers.

In a typical setting of gradient coding (e.g. [5], [6]), we let $A \in \mathbb{R}^{N\times m}$ be an *assignment matrix* of data points to machines, such that $A_{ij} \neq 0$ if and only if the ith data point is held by machine j. We define the *replication factor* of an assignment as follows.

Definition I.1 (Replication Factor)**.** *The* replication factor *of an assigment matrix $A \in \mathbb{R}^{N\times m}$ is the average number of times a data point is replicated, that is, the number of non-zero entries in A divided by N.*

In coded gradient descent, at each round t of computation, the parameter server broadcasts the current point θ_t to the machines. Each non-straggling machine j returns the single vector

$$g_j := \sum_{i=1}^N A_{ij} \nabla f_i(\theta_t),$$

to the parameter server, where we have defined $f_i(\theta) := L(x_i, y_i; \theta)$. The parameter server then chooses some decoding coefficient vector $w \in \mathbb{R}^m$, where $w_j = 0$ if machine j straggles, and performs the update

$$\theta_{t+1} \leftarrow \theta_t - \gamma \sum_{j=1}^m w_j g_j \tag{1}$$

for some learning rate γ. For any coefficient vector w, we define $\alpha := Aw$, such that the update in Equation (1) can be written

$$\theta_{t+1} \leftarrow \theta_t - \gamma \sum_{i=1}^N \alpha_i \nabla f_i(\theta_t). \tag{2}$$

If a coding scheme—that is, a matrix A and a way of computing the coefficients w—can always achieve $\alpha = \mathbb{1}$, then it recovers the gradient exactly, and Equation (1) can be analyzed as full-batch gradient descent. While this is ideal, it often requires an assignment matrix with a high replication factor. If we cannot recover the full gradient exactly, we are in the case of approximate gradient coding.

Most previous work on approximate gradient coding has fallen into one of two categories. In one line of work, the non-zero coefficients w_j are fixed in advance ([5]) or only depend on the number of stragglers ([6], [7]). In particular, these non-zero coefficients do not depend on the identity of

M. Glasgow was with the Department of Computer Science, Stanford University, Stanford, CA, 94305 USA e-mail: mglasgow@stanford.edu.

M. Wootters was with the Department of Computer Science and Department of Electrical Engineering, Stanford University, Stanford, CA, 94305 USA e-mail:marykw@stanford.edu .

Manuscript received February 15, 2021; revised July 22, 2021.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author. The material includes an appendix with some proofs.

the stragglers. A second line of work ([8], [9], [10]) chooses the decoding coefficients w dynamically depending on which machines straggle. This is called *optimal decoding*[1] because the parameter server chooses w to be any vector

$$w^* \in \underset{w: w_j=0 \text{ if machine j straggles}}{\arg\min} |Aw - \mathbb{1}|_2, \tag{3}$$

where $|\cdot|_2$ denotes the 2-norm of a vector. In optimal decoding, we will denote $\alpha^* := Aw^*$.

In this work we will study gradient coding schemes with optimal decoding. The following formalizes the two definitions of decoding error we study. Let $[m]$ denote the set of integers from 1 to m.

Definition I.2 (Decoding Error under Random Straggler)**.** *Given an assigment matrix $A \in \mathbb{R}^{N\times m}$, we define the random decoding error under a p fraction of random stragglers to be*

$$\mathbb{E}_S\left[|\alpha^* - \mathbb{1}|_2^2\right] = \mathbb{E}_S\left[\min_{\substack{w\in\mathbb{R}^m \\ w_j=0 \forall j\in S}} |Aw - \mathbb{1}|_2^2\right]$$

where S is a random subset of $[m]$ that includes each value with probability p.

In all future instances, we will omit the subscript S and take $\mathbb{E}$ to mean the expectation over the random set of stragglers.

Definition I.3 (Decoding Error under Adversarial Straggers)**.** *Given an assigment matrix $A \in \mathbb{R}^{N\times m}$, we define the adversarial decoding error under a p fraction of adversarial stragglers to be*

$$\max_{S\subset[m]:|S|\le pm}\left[|\alpha^* - \mathbb{1}|_2^2\right] = \max_{S\subset[m]:|S|\le pm}\left[\min_{\substack{w\in\mathbb{R}^m \\ w_j=0 \forall j\in S}} |Aw - \mathbb{1}|_2^2\right].$$

Both random stragglers and adversarial stragglers arise in practice. While random stragglers may arise due to system level variabilities (such as maintainance activities) [2], adversarial stragglers may arise due to hardware differences among machines or in settings where gradients are slower to compute at some data points.

The main objective in approximate gradient coding is to design assignment matrices A with a small replication factor and a small decoding error in the presence of stragglers. The work [8] showed that a particular fractional repetition code (FRC) introduced by [4] achieves the optimal decoding error with random stragglers, over all assigment matrices with the same replication factor. However, the FRC of [4] performs poorly over adversarially chosen stragglers relative to other assignments with the same replication factor. This motivates the main question behind our work:

Question 1. *Are there gradient codes that simultaneously achieve small decoding error under both random and adversarial stragglers?*

As pointed out in the open questions of [8], this question is challenging because of the difficulty of analyzing the decoding error under random stragglers. Indeed, bounding the decoding error with optimal decoding amounts to analyzing the pseudoinverse of the random matrix generated by removing a random set of columns (corresponding to straggling machines) from the assignment matrix.[2] This is particularly challenging with this random matrix is sparse, which arises when the replication factor is small.

A. Contributions

In this paper we develop schemes that achieve small decoding errors in both the random and an adversarial model simultaneously. To ensure that gradient descent will converge to the minimum of $f := \sum_i f_i$ in the random straggler setting, we construct codes that yield an unbiased approximation of the gradient. That is, when each machine is chosen independently to be a straggler with probability p,

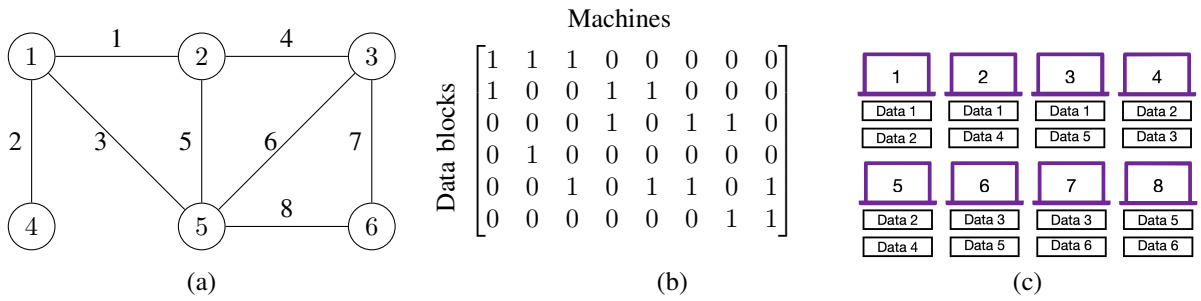
$$\mathbb{E}\sum_i \alpha_i \nabla f_i(\theta_t) = c\nabla f(\theta_t)$$

for some constant c. In such unbiased schemes, where $\mathbb{E}[\alpha] = c\mathbb{1}$, we will define $\overline{\alpha} := \frac{\alpha}{c}$. Here, and in rest of this paper, we use the asymptotic notation big-O and little-o to denote limiting behavior as d goes to ∞: We say that $f(d) = O(g(d))$ if $\limsup_{d\to\infty} \frac{|f(d)|}{g(d)} < \infty$ and $f(d) = O(g(d))$ if $\limsup_{d\to\infty} \frac{f(d)}{g(d)} = 0$. With this notation, our contributions are as follows.

1) **A new approach to analyzing optimal coefficient decoding.** While in general analyzing the optimal decoding coefficients is difficult, we develop a framework in which it is tractable. More precisely, we construct matrices A from a graph G by viewing the data blocks as vertices of G, and the machines as edges of G, holding two data blocks each. (See Definition II.2 and Figure 1). For a desired replication factor d, we partition the data into blocks of size $\frac{dN}{2m}$, and assign each machine exactly two blocks.
   In both the random and the adversarial case, we relate the decoding error to the *spectral expansion* of the graph G, defined as the gap between the largest and second largest eigenvalues of the adjacency matrix of G. In particular, in the random case we are able to analyze the optimal decoding coefficients by considering random sparsifications of this graph.
   Because of the structure of A, in our framework we can compute the optimal decoding coefficients w^* in $c \times m$ operations, where c is a universal constant. This is on the same order of the number of operations for the parameter server to compute the update in Equation (1).
2) **Progress on Question 1.** Using our framework, we construct assignment schemes based on expander graphs (graphs with large spectral expansion) that achieve the following bounds in both the random and adversarial settings.

[1]It's not necessarily the case the "optimal decoding" coefficient lead to optimal convergence. However, we use the term to be consistent with the literature [8], [7], [11], [10].

[2]Formally, $\alpha^* = A(p)(A(p)^T A(p))^\dagger A(p)^T \mathbb{1}$, where $A(p)$ is the matrix obtained by deleting each column of the assignment matrix A with probability p. Here, for a matrix M, $M^\dagger$ denotes the Moore-Penrose pseudoinverse of M.



(a) (b) (c)

Fig. 1. The assignment generated from the pictured graph. (a) A graph G, where the vertices of G correspond to data blocks and the edges correspond to machines. (b) The assignment matrix A generated from G. (c) The distribution of data blocks to machines.

- In the random setting with optimal decoding, we show in Theorem IV.1 that the decoding error decays exponentially in the replication factor d: $\frac{1}{N}\mathbb{E}\left[|\alpha^* - \mathbb{1}|_2^2\right] = p^{d-o(d)}$. This nearly matches the lower bound of $p^d/(1-p^d)$ (see Proposition A.3 in Appendix A) up to the $o(d)$ term in the exponent. In comparison, for all coding schemes with fixed decoding coefficients, we show in Proposition A.1 in Appendix A that the error decays at best like $1/d$: $\frac{1}{N}\mathbb{E}\left[|\overline{\alpha} - \mathbb{1}|_2^2\right] \geq \frac{p}{d(1-p)}$.
- In the adversarial setting, for any choice of $\lfloor pm \rfloor$ stragglers, we show in Corollary V.3 that our coding schemes achieve $\frac{1}{N}|\alpha^* - \mathbb{1}|_2^2 \leq \frac{1-o(1)}{2}\frac{p}{1-p}$. For small p, this is nearly a factor of two improvement over the FRC of [4].

3) **Provable convergence with random stragglers.** With random stragglers, our assignment schemes yield good convergence rates under reasonable assumptions about the f_i. This is because we obtain an unbiased approximation of the gradient and can additionally bound the norm of the covariance matrix of α^*. In particular, we show in Proposition VI.3 that as the desired convergence threshold ϵ approaches 0, the dominant term in the number of iterations of coded gradient descent required is $\frac{\log(1/\epsilon)}{\epsilon}p^{d-o(d)}$, where the little-o hides constant factors that depend on p and the functions f_i. We also provide a black-box[3] method to debias any coding scheme for random stragglers, that is, given any coding scheme for random stragglers, our tool allows us to convert it to an unbaised scheme without needing to know the inner working of the code. This implies that any further progress on Question 1 will yield convergence bounds on gradient descent (see Proposition B.1 in Appendix B).
4) **Provable convergence with worst-case stragglers.** With adversarial stragglers, it is not possible to guarantee convergence to the minimizer θ^* of $f = \sum_{i=1}^n f_i$. However, if the strong convexity of f is larger than the product of the adversarial decoding error $|\alpha^* - \mathbb{1}|_2^2$ and the maximum Lipshitz constant of any ∇f_i, then we can guarantee that coded gradient descent converges down to some noise floor. We show that this noise floor scales linearly with the adversarial quantity $|\alpha^* - \mathbb{1}|_2^2$. More precisely, we show in Corollary VII.2 that we can converge to a floor of $|\theta_t - \theta_*|_2^2 \leq O\left(|\alpha^* - \mathbb{1}|_2^2\right)$, where the big-O hides constant factors that depend on the functions f_i. To our knowledge, this is the first provable convergence guarantee for adversarial stragglers in coded gradient descent. Previous works have obtained adversarial bounds on $|\alpha^* - \mathbb{1}|_2^2$ without establishing convergence results [6], [9], [7]; Corollary VII.2 also implies convergence results for these works as well.
5) **Empirical Success.** Our algorithm produces good non-asymptotic results. In Section VIII, we demonstrate empirically that in the random straggler setting, the expected error $\mathbb{E}\left[|\overline{\alpha^*} - \mathbb{1}|_2^2\right]$ in our schemes nearly meets the lower bound of $p^d/(1-p^d)$. We also show that gradient descent converges in fewer iterations using optimal decoding with our scheme than when using fixed coefficient decoding, and in over d times fewer iterations than an uncoded approach which simply ignores stragglers. In particular, after 50 iterations of our algorithm with a replication factor of 3, we observe at least a $\frac{1}{3p^2}$ improvement in mean squared error over fixed coefficient decoding, and at least a $\frac{1}{10p^2}$ improvement in mean squared error over an uncoded approach after 150 iterations. We observe that our approach converges at the same rate or faster than the state-of-the-art approaches of [10] and [6].

[3]By black-box methods, we mean methods that only leverage the variance of the gradient update and none of its other statistical properties.

B. Related Work

Gradient coding techniques for distributed optimization were first considered in [4], where some assignment schemes based on *fractional repetition codes* (FRC) were used to recover the *exact* gradient under *worst-case* stragglers. In the particular FRC used by [4], the data points and the machines are each partitioned into an equal number of disjoint blocks, and each machine in a block receives all the data points in the corresponding block of data points. This body of work on gradient coding was continued in [12], [13], [11], [14] and [15], which established the exact trade-off between computation load, worst-case straggler tolerance, and communication complexity.

Coding Scheme	Decoding Coefficients	$\frac{1}{N}\mathbb{E}\left[\lvert\alpha-\mathbb{1}\rvert_2^2\right]$	Worst Case $\frac{1}{N}\lvert\alpha-\mathbb{1}\rvert_2^2$	Convergence Proof?
Expander Code (Cor. 23 [6])	Fixed	-	$< \frac{4p}{d(1-p)}$	Yes (random stragglers)
Pairwise Balanced ([5])	Fixed	$\geq \frac{p}{d(1-p)}$ (by **??**)	-	Yes (random stragglers)
BIBD (Const. 1 [7])	Fixed and Optimal	-	$O(\frac{1}{\sqrt{m}})$ $d=\Omega(\sqrt{m})$	No
BRC ([9])	Optimal	$e^{-\Theta(d)}$	-	No
rBGC ([8])	Fixed	$< \frac{1}{(1-p)d}$	-	No
FRC of [4] (and [10])	Optimal	p^d	p	Yes (random stragglers)
Theorem IV.1, Corollary V.2	Optimal	$p^{d-o(d)}$	$\frac{(1+o(1))p}{2(1-p)}$	Yes (both random and adversarial stragglers)

TABLE I

COMPARISON OF RELATED WORK. THE COLUMN CONTAINING THE QUANTITY $\mathbb{E}\left[\lvert\alpha-\mathbb{1}\rvert_2^2\right]$ IS IN EXPECTATION OVER A p FRACTION OF RANDOM STRAGGLERS. THE COLUMN CONTAINING THE QUANTITY $\lvert\alpha-\mathbb{1}\rvert_2^2$ IS THE WORSE CASE VALUE OVER A p FRACTION OF STRAGGLERS. FOR UNBIASED CODING SCHEMES, THE RESULTS PERTAIN TO $\overline{\alpha}$ INSTEAD OF α.

A line of work ([6], [10], [5], [16], [8], [7]) initiated by [6] explores the landscape of *approximate gradient coding*, where the gradient is not recovered exactly. The work [6] considers both exact and approximate gradient codes. The approximate gradient codes in [6] are based on regular expander graphs, and the non-zero decoding coefficients w are fixed up to the number of stragglers. They achieve a decoding error that decays like $1/d$ in the replication factor d, even when the straggling machines are chosen adversarially.[4] They then relax the assumption of adversarially chosen stragglers, and bound the convergence of their coded gradient descent under random stragglers, showing that the run time decreases inversely with d. The work of [5] combines *pair-wise balanced* coding schemes with a tight convergence analysis to yield convergence times that decay like $1/d$; that work also uses fixed decoding coefficients w. The work [7] considers the problem of approximate gradient recovery when the straggling machines are chosen adversarially, and shows that for assignment matrices based on *balanced incomplete block designs* (BIBD), an optimal decoding vector w^* will always have fixed coefficients.

The most related works to ours are [8] and [9], which consider optimal decoding under random stragglers. The work [8] was the first to use optimal decoding in the approximate gradient setting, and established that the the FRC-based assignment of [4] (which is also identical to that in [10]) achieves the decoding error $\frac{1}{N}\mathbb{E}|\alpha^*-\mathbb{1}|_2^2 = p^d$ over random stragglers, which is optimal over all schemes with a replication factor of d. They show that the this FRC performs poorly in the adversarial setting, and so they also provide a random construction called a regularized Bernoulli Gradient Code (rBGC), which they suggest is harder to exploit by a computationally bounded adversary. In [10], the authors provide bounds on the convergence rate of coded gradient descent using the FRC and optimal decoding under random stragglers. The work [9] provides upper and lower bounds on the computational load required to achieve a desired decoding error with high probability over random stragglers. Their upper bound is based on a construction using batch raptor codes (BRC) which achieves $\frac{1}{N}\mathbb{E}|\alpha^*-\mathbb{1}|_2^2 = 1/e^{O(d)}$. We summarize the most relevant results from the work on approximate gradient codes in Table I. To our knowledge, ours is the first analysis of an assignment scheme that achieves a decoding error decaying exponentially in d for random straggler and a decdoing error less than $\frac{pm}{N}$ for adversarial stragglers. We show that the decoding error under random stragglers in our scheme is near-optimal as a function of the computational load, while the decoding error with adversarial stragglers is nearly twice as small as that of the FRC of [4].

Unlike many previous works that only study the decoding error $|\alpha^*-\mathbb{1}|_2$ [6], [9], [7], we also provide convergence results for both the random and adversarial settings. To the best of our knowledge, our work gives the first provable convergence results for approximate gradient coding with adversarial stragglers, although we note that there have been convergence results shown in other adversarial settings of gradient descent [17], [18].

Other work such as [16] also considers the problem of approximate gradient coding, but differs from our framework in that their codes are not based on assignment matrices, or require specific types of loss functions.

C. Organization

In Section II, we describe our construction of approximate gradient codes, and give some intuition for why we can show good bounds on the decoding error of our constructions. In Section III, we characterize the optimal coefficents $w*$ and the resulting vector $\alpha^* = Aw^*$ in terms of the the straggling machines in a graph assignment scheme. In Section IV, we prove our main result Theorem IV.1, on the performance of

[4]This follows by using a Ramanujan expander in Corollary 23 of [6].

m	Number of machines
N	Number of data points
n	Number of data blocks in a graph-based scheme
ℓ	Computational load (maximum points per machine)
d	Replication Factor (averaged over all data points)

TABLE II
PARAMETERS IN THIS WORK. WE ALWAYS HAVE $d = 2m/n = m\ell/N$. A USEFUL PARAMETER REGIME TO KEEP IN MIND IS THE SETTING WHERE $N = m$ AND $\ell = d$.

graph-based assignment schemes in the setting of random stragglers. In Section V, we prove Corollary V.3, on the robustness of graph-based assignment schemes to adversarial stragglers. In Section VI, we state Proposition VI.1 on the convergence of gradient descent for random stragglers. In Section VII, we state Proposition VII on the convergence of gradient descent for adversarial stragglers. In Section VIII, we provide simulations which demonstrate our theoretical claims. We conclude in Section IX. Some proofs are deferred to the appendix.

D. Notation

We will use $|\cdot|_2$ to denote the 2-norm of a vector or the operator norm of a matrix. For a graph $G = (V, E)$ and any sets of vertices $S, T \subset V$, we will denote by $E(S, T)$ the set of edges between vertices in S and vertices in T. We will denote by $\partial(S)$ the edges $E(S, V \setminus S)$. For an edge $e \in E$, we denote by $\delta(e)$ the two endpoints of the edge e.

Let $\mathcal{S}_n$ denote the symmetric group on n elements, and for a graph $G = (V, E)$ on n vertices, let $\mathrm{Aut}(G) \subset \mathcal{S}_n$ denote the set of graph automorphisms of G. We say that a graph is *vertex transitive* if for any vertices $u, v \in V$, there exists some automorphism $\sigma \in \mathrm{Aut}(G)$ such that $\sigma(u) = v$. We denote the action of an automorphism σ on a set $S \subset V$ in the following natural way: $\sigma(S) = \{\sigma(v) : v \in S\}$. For a permutation $\rho \in \mathcal{S}_n$, we denote the action of ρ on a vector β in the following way: $\rho(\beta)_i = \beta_{\rho(i)}$.

II. OUR CONSTRUCTION

In our construction, each machine holds exactly two data blocks, each comprised of $\frac{dN}{2m}$ data points. We introduce the parameter $n := \frac{2m}{d}$ to denote the number of data blocks. We summarize these parameters in Table II.

Remark II.1. *Because our assignment schemes are regular—that is, each data block is replicated an equal number of times—each data point will be replicated exactly d times. Observe that the computational load ℓ, the maximum number of data points per machine, equals $\frac{dN}{m}$. As the regime $m = N$ is the most commonly studied, it is convenient to think of d as equal to ℓ when comparing our results to other work, some of which state results in terms of ℓ. In general, when $m = N$, we must have $\ell \geq d$.*

We can describe these assignment schemes using a graph on n vertices with m edges. We abuse notation and use the assignment matrix A to denote the $n \times m$ assignment matrix of blocks to machines, rather than the $N \times m$ assignment matrix of points to machines. Thus, all of our results are in terms of the replication factor $d = \frac{2m}{n}$, which is independent of the block size.

Definition II.2. *A graph assignment scheme corresponding to a graph G with n vertices and m edges is a matrix $A \in \{0, 1\}^{n \times m}$ in which $A_{ij} = 1$ if the jth edge of G has i as an endpoint.*

An example of Definition II.2 is shown in Figure 1.

Remark II.3. *In contrast to other works (such as [6]), which have also designed codes based on graphs, the graph we consider is* not *a bipartite graph where left vertices correspond to data blocks and right vertices correspond to machines. Rather, it is the non-bipartite graph where the data blocks are the vertices and the machines are the edges.*

Recall that to minimize the decoding error, we want to show that α^* is close to $\mathbb{1}$, such that the gradient updates given in Equation (1) are as close as possible to those in batch gradient descent. By thinking of an assignment scheme as a graph G as above, we are able to characterize α^* in terms of the connected components of a random sparsification of G. In Section III, we show that α_i^* will be close to 1 if vertex i is in component which is either non-bipartite, or bipartite with close to balanced sides.

Expanders are good examples of sparse graphs which have large non-bipartite components under random sparsification. We show this in Corollary IV.4 by proving that randomly sparsified expanders have a giant connected component with high probability (Theorem IV.3). To additionally guarantee that our gradient descent converges to the true minimum, we use vertex transitive expanders, such as Cayley graph expanders, which guarantee that $\mathbb{E}[\alpha^*] = c\mathbb{1}$.

III. CHARACTERIZATION OF α^*

In this section, we characterize α^* in terms of the straggling machines in a graph assignment scheme. This will allow us to prove the desired properties of α^* by studying randomly sparsified graphs.

Suppose we have some graph assignment scheme A corresponding to a graph G. Recall that $\alpha^* = Aw^*$, where

$$w^* \in \underset{w : w_j = 0 \text{ if machine } j \text{ straggles}}{\arg\min} |\mathbb{1} - Aw|_2.$$

We define $G(p)$ to be the random graph where each edge of G is deleted with probability p.

We can think of w^* as a weight vector which has one (possibly) non-zero coordinate w_e^* for each edge e in $G(p)$. We can think of α^* as a vector where each coordinate α_v^* is the sum of weights w_e^* of each edge e incident to v. See Figure 2 for some examples of these.

It follows from Equation (3) that for any edge $e = (u, v)$, α^* satisfies

$$\alpha_u^* + \alpha_v^* = 2. \tag{4}$$

Indeed, at the optimum we have $0 = A^T(\mathbb{1} - Aw^*) = A^T(\mathbb{1} - \alpha^*)$, which implies that for all edges $e = (u, v)$ (which index

the rows of A^T), we have $(1-\alpha_u^*)+(1-\alpha_v^*)=0$, yielding Equation (4).

We can make the following observations which follow from Equation (4):

1) For any set of vertices v in a single connected component, $|1-\alpha_v^*|$ is the same. Indeed, for an edge (u,v), Equation (4) implies that $1-\alpha_u^*=\alpha_v^*-1$, and this relationship extends to a whole connected component.
2) If a component contains an odd cycle of vertices (i.e., is not bipartite), then $\alpha_v^*=1$ for all of the vertices in the component. Indeed, as above, the sign of $1-\alpha_v^*$ alternates along edges of the component, which would produce a contradiction if $1-\alpha_v^*$ was not 0 at every vertex in the odd cycle.
3) If a component $C=L\cup R$ is bipartite with $|L|\geq|R|$, then $\alpha_u^*=1+\frac{|L|-|R|}{|L|+|R|}$ if $u\in R$ and $\alpha_v^*=1-\frac{|L|-|R|}{|L|+|R|}$ if $v\in L$. This is true because the sum of all edge weights going into vertices in L is equal to the sum of all edge weights going into vertices in R, so $\sum_{u\in R}\alpha_u^*=\sum_{v\in L}\alpha_v^*$. Using items (1) and (2) to conclude that α_u^* is constant on $u\in R$ and $\alpha_v^*=2-\alpha_u^*$ is constant on $v\in L$ yields the statement.

These observations suggest the approach that we will use in Section IV: we can bound the contribution to $|\alpha^*-\mathbb{1}|_2^2$ of a particular connected component by simply knowing whether that component is bipartite.

Algorithmically, given the set of non-straggling machines, the observations above allow the parameter server to compute the optimal coefficients w^* in linear time in m. First the parameter server performs a breadth-first search on $G(p)$ to divide the graph into connected components, and determines the two sides L and R of any bipartite components. For each connected component, the parameter server can then compute α_v^* for each v in the component. Finally, the parameter server performs a depth-first search on each component to label each edge with the a value w_e^* such that the sum of all edges incident to vertex v equal α_v^*. Note that the value of w_e^* may depend on the order edges are discovered in the depth-first search, but the vector α^* is unique.

IV. The Decoding Error $|\alpha^*-\mathbb{1}|_2$ under Random Stragglers

In this section we prove our main result about expander graph assignments under random stragglers.

Theorem IV.1. *Let $G=(V,E)$ be any vertex transitive graph with n vertices, m edges, and spectral expansion λ. Let A be the assignment matrix given by G, in accordance with Definition II.2. Suppose for some positive ϵ,*

1) $\lambda>1.5$;
2) $(\frac{\lambda}{2}+1)(1-pe^{1/\lambda})\geq 1+\epsilon$;
3) $\left(1-\frac{e^2p^{\lambda\left(1-\frac{1}{3+\epsilon}\right)}}{(1-pe^{1/\lambda})^2}\right)\left(\lambda-(2d-\lambda)\frac{e^2p^{\lambda\left(1-\frac{1}{3+\epsilon}\right)}}{(1-pe^{1/\lambda})^2}\right)\geq\epsilon$;
4) $\frac{n}{\log(n)^2}\geq\frac{4(1+\epsilon)}{\epsilon^2}\left(2-2\log(\epsilon)+2\log(1+\epsilon)-\log\left(1-pe^{\frac{1}{\lambda}}\right)\right)$.

If each machine straggles independently with probability p, then

1) $\mathbb{E}[\alpha^*]=r\mathbb{1}$ *for some* $r\geq 1-\frac{6}{n}-t$;
2) *For all* i, $\mathbb{E}[(\alpha_i^*-r)^2]\leq\frac{6}{n}+t$;
3) $|\mathbb{E}[(\alpha^*-r\mathbb{1})(\alpha^*-r\mathbb{1})^T]|_2\leq 2k^2\left(t+\frac{6}{n}\right)^2+24$,

where $t=\frac{e^2p^{\lambda\left(1-\frac{1}{3+\epsilon}\right)}}{(1-pe^{1/\lambda})^2}$,

$$k=\frac{2(1+\epsilon)}{\epsilon^2}\left(2\log(n)-2\log(\epsilon)+2\log(1+\epsilon)-\log(1-p)\right),$$

and all expectations are over the random stragglers.

Remark IV.2. *The expected error $\frac{1}{n}\mathbb{E}\left[|\overline{\alpha^*}-\mathbb{1}|_2^2\right]$ is lower-bounded by p^d, because this is the probability that a fixed data block is stored only at straggling machines. Theorem IV.1 implies that, for good expanders, the variance of α^* shrinks exponentially in the replication factor, d. One example of such graphs are the Lubotzky-Phillips-Sarnak (LPS) construction [19] of Ramanujan Cayley graphs, where $\lambda\geq d-2\sqrt{d-1}$. In this sense, our result is tight in d up to factors of $p^{o(d)}$.*

Theorem IV.1 is a consequence of Theorem IV.3 and Corollary IV.4 below. We will state these results and then prove Theorem IV.1 assuming them. We prove Theorem IV.3 and Corollary IV.4 in Appendix C.

Theorem IV.3. *Let G be any d-regular λ-spectral expander with n vertices and suppose for some positive ϵ,*

1) $\lambda>1.5$;
2) $(\frac{\lambda}{2}+1)(1-p)\geq 1+\epsilon$;
3) $\frac{n}{\log(n)^2}\geq\frac{2(3+\epsilon)(1+\epsilon)}{\epsilon^2}(\log(e(1+\epsilon)/\epsilon)-\log(1-p))$.

Let $G(p)$ be a random sparsification of G, where each edge of G is deleted randomly with probability p. With probability at least $1-\frac{5}{n}$,

1) *$G(p)$ has a giant component of size at least $n\left(1-\frac{ep^{\lambda\left(1-\frac{1}{3+\epsilon}\right)}}{(1-p)^2}\right)$ vertices;*
2) *Every vertex is either in a component of size at most k, where k is as in Theorem IV.1, or is in a component of size greater than $n/2$.*

Corollary IV.4. *Let G be any d-regular λ-spectral expander with, and suppose for some positive ϵ,*

1) $\lambda>1.5$;
2) $(\frac{\lambda}{2}+1)(1-pe^{1/\lambda})\geq 1+\epsilon$;
3) $\left(1-\frac{e^2p^{\lambda\left(1-\frac{1}{3+\epsilon}\right)}}{(1-pe^{1/\lambda})^2}\right)\left(\lambda-(2d-\lambda)\frac{e^2p^{\lambda\left(1-\frac{1}{3+\epsilon}\right)}}{(1-pe^{1/\lambda})^2}\right)\geq\epsilon$;
4) $\frac{n}{\log(n)^2}\geq\frac{4(1+\epsilon)}{\epsilon^2}\left(2\log(e(1+\epsilon)/\epsilon)-\log\left(1-pe^{\frac{1}{\lambda}}\right)\right)$.

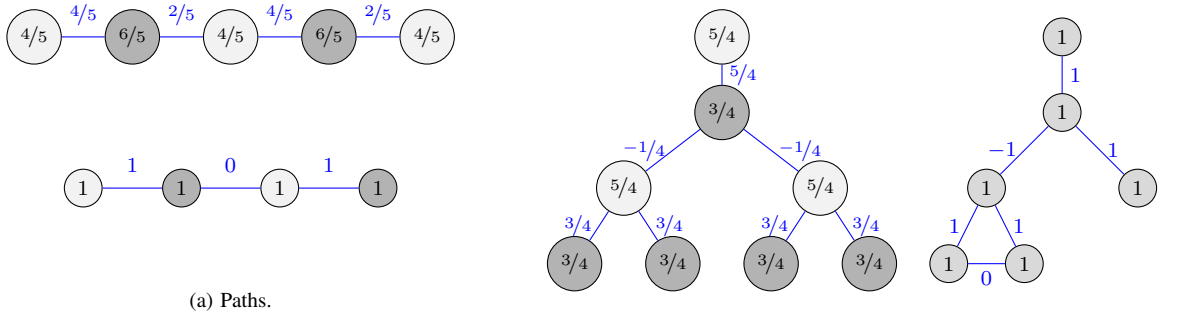
Let $G(p)$ be a random sparsification of G, where each edge of G is deleted randomly with probability p. Then with probability at least $1-\frac{6}{n}$,

1) *$G(p)$ has a non-bipartite giant component of size at least $n\left(1-\frac{e^2p^{\lambda\left(1-\frac{1}{3+\epsilon}\right)}}{(1-pe^{1/\lambda})^2}\right)$.*
2) *Every vertex is either in a component of size at most k, where k is as in Theorem IV.1, or is in a component of size greater than $n/2$.*

We begin by proving Theorem IV.1 assuming Theorem IV.3 and Corollary IV.4.

Proof. (Theorem IV.1) Recall that because the graph G is vertex-transitive, the distribution of α_i^* is equivalent for every

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAIT.2021.3100110, IEEE Journal on Selected Areas in Information Theory



(a) Paths. (b) Tree. (c) Non-bipartite component.

Fig. 2. The optimal choice of w^* (edge labels) and α^* (vertex labels) in various connected components.

vertex i, and so $\mathbb{E}[\alpha^*]$ is some multiple of $\mathbb{1}$. Throughout we will use the fact that by Corollary IV.4, with probability at least $1-\frac{6}{n}$, at least $(1-t)n$ vertices i are in a non-bipartite components, and hence have $\alpha_i^* = 1$. Here, t is as in the statement of Theorem IV.1.

For the first statement, for any i, because $\alpha_i^* \geq 0$, we have

$$\mathbb{E}[\alpha_i^*] \geq \Pr[\alpha_i^* = 1] \geq \Pr[\text{vertex } i \text{ is in a non-bipartite component}] \geq 1 - \frac{6}{n} - t.$$

For the second statement,

$$\mathbb{E}[(\alpha_i^* - \mathbb{E}[\alpha_i^*])^2] \leq \mathbb{E}[(\alpha_i^* - 1)^2] \leq 1 - \Pr[\alpha_i^* = 1] \leq \frac{6}{n} + t,$$

where we used the fact that $|1 - \alpha_i^*| \leq 1$ always.

The third statement follows from the following lemma, which we prove in Appendix C:

Lemma IV.5. *Given the assumptions in Theorem IV.1, we have*

$$|\mathbb{E}[(\alpha^* - r\mathbb{1})(\alpha^* - r\mathbb{1})^T]|_2 \leq 2k^2\left(t + \frac{6}{n}\right)^2 + 24,$$

where r, t and k are as defined in Theorem IV.1.

□

V. The Decoding Error $|\alpha^* - \mathbb{1}|_2$ under Adversarial Stragglers

In this section we show how the spectral properties of an assignment matrix A can be leveraged to bound the adversarial error $|\alpha^* - \mathbb{1}|_2^2$. We show that graph-based assignment schemes which use graphs with large expansion perform nearly twice as well as the FRC of [4] in the adversarial setting.

The following proposition and its proof are almost the same as Proposition 29 in [6].

Proposition V.1. *Let $A \in \mathbb{R}^{N\times m}$ be any assignment matrix on N data points, for which each data point is replicated exactly d times, and each machine holds exactly ℓ data points. Let σ_2 be the second largest singular value of A. For any set of S of s stragglers, there exist some decoding coefficients $w = w(S)$ such that*

$$\frac{1}{N}|\alpha - \mathbb{1}|_2^2 \leq \frac{1}{N}\left(\frac{\sigma_2}{d}\right)^2 \frac{sm}{m-s}$$

.

Proof. For any set S of s stragglers, let $w_i = \frac{m}{d(m-s)}$ for $i \notin S$ and $w_i = 0$ for $i \in S$. Then

$$|\alpha - \mathbb{1}|_2^2 = \left|Aw - \frac{1}{d}A\mathbb{1}\right|_2^2 = |Az|_2^2, \tag{5}$$

where $z = w - \frac{1}{d}\mathbb{1}$. We observe that A has top singular value $\sigma_1 = \sqrt{\ell d}$ and top right singular vector $\mathbb{1}/\sqrt{m}$; this follows from the fact that A^TA evidently has top eigenvector $\mathbb{1}/\sqrt{m}$ and top eigenvalue ℓd.

Observe that $z \perp \mathbb{1}$ and $|z|_2^2 = \frac{1}{d^2}\left(s + (m-s)\frac{s^2}{(m-s)^2}\right) = \frac{ms}{(m-s)d^2}$. Thus

$$|Az|_2^2 \leq \sigma_2^2|z|_2^2 \leq \left(\frac{\sigma_2}{d}\right)^2 \frac{sm}{m-s}$$

as desired. □

Corollary V.2. *Let $A \in \mathbb{R}^{n\times m}$ be a graph assignment scheme corresponding to some d-regular graph G with spectral expansion λ. Then for any set of $\lfloor pm \rfloor$ stragglers, there exists some decoding coefficients w such that*

$$\frac{1}{n}|\alpha - \mathbb{1}|_2^2 \leq \frac{2d-\lambda}{2d}\frac{p}{(1-p)}.$$

Proof. Let $\mathcal{A}(G)$ denote the adjacency matrix of G such that $\lambda_1(\mathcal{A}(G)) = d$ and $\lambda_2(\mathcal{A}(G)) = d - \lambda$. Observe that because A has exactly two ones per column, we have $A^TA = \mathcal{A}(G) + dI$, such that $\sigma_2(A) = \sqrt{\lambda_2(A^TA)} = \sqrt{2d-\lambda}$. Applying Proposition V.1 implies that

$$\frac{1}{n}|\alpha - \mathbb{1}|_2^2 \leq \frac{1}{n}\frac{2d-\lambda}{d^2}\frac{p}{(1-p)}m.$$

Using the fact that $d = 2m/n$ concludes the proof. □

Corollary V.3. *Let $A \in \mathbb{R}^{n\times m}$ be a graph assignment scheme corresponding to some d-regular graph G with spectral expansion λ. Let λ be the spectral gap of G and suppose $\lambda = d - o(d)$. Then for any set of $\lfloor pm \rfloor$ stragglers, there exists some decoding coefficients w such that*

$$\frac{1}{n}|\alpha - \mathbb{1}|_2^2 \leq \frac{1+o(1)}{2}\frac{p}{(1-p)}.$$

Proof. Plugging in $\lambda = d - o(d)$ to Corollary V.2, we have

$$\frac{1}{n}|\alpha - \mathbb{1}|_2^2 \leq \frac{2d-\lambda}{2d}\frac{p}{(1-p)} = \frac{1+o(1)}{2}\frac{p}{1-p}$$

as desired. □

Remark V.4 (Tightness of Corollary V.3)**.** *This bound is nearly tight for graph assignment schemes when p is small. Indeed, for any graph assignment scheme on with mp stragglers and replication factor d, we can adversarially choose the stragglers such that at least $\frac{mp}{d}$ data blocks are not held at any non-straggling machines. Thus, for any decoding coefficients α we have*

$$\frac{1}{n}|\alpha - \mathbb{1}|_2^2 \geq \frac{mp}{dn} = \frac{p}{2}$$

using the fact that $nd = 2m$ for graph-based schemes.

While our scheme improves by nearly a factor of two over the FRC of [4], it is worse by an order of $d/8$ from the expander code of [6], which meets the lower bound in adversarial error up to constant factors for a replication factor of d (See Table I). We leave it as an open question to improve on our scheme for adversarial stragglers while maintaining our performance for random stragglers.

VI. Convergence with Random Stragglers

In this section, we bound the convergence rate of our coded gradient descent algorithm with random stragglers. This algorithm begins by distributing the data blocks according to the assignment matrix A. We additionally shuffle our assignment of data blocks to machines using a random permutation ρ. The iterative computation phase of the algorithm follows Equation (2) in the introduction, where the coefficients w are given by the *optimal* decoding coefficients in Equation (3). For the reader's convenience, we summarize the logical view of this algorithm with random straggers in the algorithm below.

Note that in this algorithm, the optimal decoding vector w^* computed by the parameter server might not be unique, but the vector $\alpha^* := Aw^*$ is unique. Indeed, for a straggler rate of p, let $A(p)$ be the random matrix which is a copy of A with each column replaced with zeros independently with probability $1-p$. Then α^* is the unique projection of the all-ones vector onto the space spanned by $A(p)$, namely

$$\alpha^* = A(p)(A(p)^T A(p))^\dagger A(p)^T \mathbb{1}. \tag{6}$$

Given a matrix A, let P_{α^*} be the distribution of the random vector α^* defined in Equation (6). Similarly let $P_{\overline{\alpha^*}}$ be the distribution of $\overline{\alpha^*}$, which we recall is defined to be the normalization $\alpha^* \frac{|\mathbb{1}|_2}{|\mathbb{E}[\alpha^*]|_2}$. Then for any unbiased decoding scheme, GCOD$(A, p, x_0, \gamma, \{f_i\}, k)$ is stocastically equivalent to the gradient descent algorithm, SGD-ALG$(P_{\overline{\alpha^*}}, x_0, \gamma\mathbb{E}[\alpha_1], \{f_i\}, k)$, given in Algorithm 2.

We provide convergence analysis of SGD-ALG in the following proposition, for distributions P_β with $\mathbb{E}_{\beta\sim P_\beta}[\beta] = \mathbb{1}$.

Proposition VI.1. *Let $f = \sum_i^n f_i$ be a μ-strongly convex function with an L-Lipshitz gradient, and suppose each f_i is convex, and all gradients ∇f_i are L'-Lipshitz. Let x^* be the minimizer of f. Let $\sigma^2 = \sum_i |\nabla f_i(x^*)|_2^2$.*

Algorithm 1 Gradient Coding with Optimal Decoding (GCOD): Logical View with Random Stragglers

procedure GCOD($A, p, \theta_0, \gamma, \{f_i\}, k$)
▷ Distribution Phase
$\rho \sim$ Uniform$(\mathcal{S}_n)$ ▷ ρ is a random permutation
for $i = 1$ **to** n **do**
Send $f_{\rho(i)}$ to all machines j such that $A_{i,j} \neq 0$
end for
▷ Computation Phase
for $t = 1$ **to** k **do**
Parameter server: Send x_{t-1} to each machine
for Machine $j \in [m]$ **do**
$B_j \sim$ Bernoulli(p)
if $B_j = 1$ **then**
Machine j: Send $g_j = \sum_i A_{ij}\nabla f_{\rho(i)}(\theta_{t-1})$ to parameter server.
end if
end for
Parameter server: Computes $w^* \in \arg\min_{w:w_j=0 \text{ if } B_j=0}(|Aw - \mathbb{1}|_2)$
Parameter server: $\theta_t \leftarrow \theta_{t-1} - \gamma\sum_{j:B_j=1} w_j^* g_j$
end forreturn θ_k
end procedure

Algorithm 2 SGD-ALG$(P_\beta, \theta_0, \gamma, \{f_i\}, k)$

procedure SGD-ALG$(P_\beta, \theta_0, \gamma, \{f_i\}, k)$
$\rho \sim$ Uniform$(\mathcal{S}_n)$ ▷ ρ is a random permutation
for $t = 1$ **to** k **do**
$\beta \sim P_\beta$
$\theta_t \leftarrow \theta_{t-1} - \gamma\sum_i \beta_i \nabla f_{\rho(i)}(\theta_{t-1})$
end forreturn θ_k
end procedure

Suppose we run the gradient descent as in Algorithm 2, SGD-ALG$(P_\beta, x_0, \gamma, \{f_i\}, k)$, starting from x_0 for k iterations with some step size $\gamma \leq \frac{1}{sL'+L}$ and some distribution P_β such that $\mathbb{E}[\beta] = \mathbb{1}$. Let $r := \frac{1}{n}\mathbb{E}\left[|\beta - \mathbb{1}|_2^2\right]$, and $s := |\mathbb{E}[(\beta - \mathbb{1})(\beta - \mathbb{1})^T]|$. Then

$$\mathbb{E}\left[|x_k - x^*|_2^2\right] \leq (1 - 2\gamma\mu\,(1 - \gamma(sL' + L)))^k\,|x_0 - x^*|_2^2 + \frac{\gamma r\left(1 + \frac{1}{n-1}\right)\sigma^2}{\mu\,(1 - \gamma(sL' + L))}, \tag{7}$$

where the expectation is over ρ and $\{\beta^{(j)} : j < k\}$.

Corollary VI.2. *For any desired accuracy ϵ, we can choose a step size*

$$\gamma = \frac{\mu\epsilon}{2\mu\epsilon(sL' + L) + 2r\left(1 + \frac{1}{n-1}\right)\sigma^2}$$

such that after

$$k = 2\log(2\epsilon_0/\epsilon)\left(\frac{sL'}{\mu} + \frac{L}{\mu} + \frac{r\left(1 + \frac{1}{n-1}\right)\sigma^2}{\mu^2\epsilon}\right)$$

steps, $\mathbb{E}\left[|x_k - x^*|_2^2\right] \leq \epsilon$*, where* $\epsilon_0 = |x_0 - x^*|^2$.

Given our results on the decoding error $\mathbb{E}|\alpha^* - \mathbb{1}|_2^2$ and covariance of α^* in Theorem IV.1, we can use Proposition VI.1 to bound the convergence time of coded gradient descent with a graph-based assignment scheme in the setting of random stragglers. This yields the following propostion.

Proposition VI.3. *Let* $f = \sum_i^n f_i$ *be a* μ*-strongly convex function with an* L*-Lipshitz gradient, and suppose each* f_i *is convex, and all gradients* ∇f_i *are* L'*-Lipshitz. Let* θ^* *be the minimizer of* f*, and define* $\sigma^2 := \sum_i |\nabla f_i(\theta^*)|_2^2$.

Suppose we perform gradient coding with optimal decoding as in Algorithm 1 with an assignment matrix corresponding to a d*-regular vertex-transitive graph with spectral gap* $d - o(d)$*, such that the number of machines* $m = \frac{nd}{2}$*. Let* p *be the probability of a machine straggling.*

Then for any desired accuracy ϵ*, we can choose some step size* γ *such that after*

$$k = 2\log(\epsilon_0/\epsilon)\left(\frac{L}{\mu} + \log^2(n)p^{2d-o(d)}\frac{L'}{\mu} + \frac{p^{d-o(d)}\sigma^2}{\mu^2\epsilon}\right)$$

steps of gradient descent, we have $\mathbb{E}\left[|\theta_k - \theta^*|_2^2\right] \leq \epsilon$*, where* $\epsilon_0 = |\theta_0 - \theta^*|^2$.

Remark VI.4. *Our results improve over black-box methods for establishing convergence of gradient descent (such as Theorem 34 in [6]) for two reasons. First, we leverage the structure of the covariance matrix of* α^* *to control the dependence on the Lipshitz constants of gradients of each data block. Second, by shuffling the data blocks before assignment, we are able to bound* $\mathbb{E}\left[\left|\sum_{i=1}^n \alpha_i^* \nabla f_i(\theta^*)\right|_2^2\right]$ *much more tightly than the naive bound* $\mathbb{E}\left[|\alpha^* - \mathbb{E}[\alpha^*]|_2^2\right]\sum_i |\nabla f_i(\theta^*)|_2^2$*. This quantity controls the constant that appears in front of* $1/\epsilon$ *in Proposition VI.3. These improvements allow us to converge up to a factor of* n *faster than black-box methods, though the exact improvement depends on the functions* f_i.

Remark VI.5. *The step size used in Proposition VI.3 scales inversely with the quantity* $E|\overline{\alpha^*} - \mathbb{1}|_2^2$*, which controls the variance of the gradient estimate. Choosing a step size inversely proportional to this variance term is common in other work (eg. [6], [20]).*

Remark VI.6. *Proposition VI.3 relies on the assignment scheme being unbiased. However, it can be applied more generally at the expense of doubling the computation load: we show in Appendix B how to debias any assignment scheme.*

We provide proofs of Proposition VI.1 and Corollary VI.2 in Appendix E. Combining Proposition VI.1 with Theorem IV.1 yields Proposition VI.3.

VII. Convergence with Adversarial Stragglers

In this section, we show that with adversarial stragglers, coded gradient descent can converge down to a noise floor which scales with the maximum value $|\alpha^* - \mathbb{1}|_2^2$.

Proposition VII.1. *Let* $f = \sum_i^n f_i$ *be a* μ*-strongly convex function with an* L*-Lipshitz gradient, and suppose each* f_i *is convex, and all gradients* ∇f_i *are* L'*-Lipshitz. Let* x^* *be the minimizer of* f*, and define* $\sigma^2 := \sum_i |\nabla f_i(x^*)|_2^2$*. Suppose we perform gradient descent with the update*

$$x_{k+1} = x_k - \gamma \sum_i \alpha_i^{(k)} \nabla f_i(x_k), \tag{8}$$

such that at each iteration, $\left|\alpha^{(k)} - \mathbb{1}\right|_2^2 \leq r^2$*. Let* $a = 1 - \frac{r\sqrt{L'}}{\sqrt{\mu}}$*, and suppose* $a > 0$*. For any* $0 < \epsilon \leq 1$*, we can choose a constant step size of*

$$\gamma = \frac{\epsilon a \mu}{4(L^2 + 2rLL' + 4r^2(L')^2)}$$

such that for some

$$\begin{aligned} k &\leq \frac{2(1+\epsilon)\log\left(\frac{2a^2\mu^2|y_0|_2^2}{(1+\epsilon)^2r^2\sigma^2}\right)}{3\gamma a\mu\epsilon} \\ &= \frac{4(1+\epsilon)(L^2 + 2rLL' + 4r^2(L')^2)\log\left(\frac{2a^2\mu^2|x_0 - x_*|_2^2}{(1+\epsilon)^2r^2\sigma^2}\right)}{3a^2\mu^2\epsilon^2} \end{aligned} \tag{9}$$

iterations, we have

$$|x_k - x_*|_2 \leq (1+\epsilon)\frac{r\sigma}{a\mu}. \tag{10}$$

Plugging in $\epsilon = 1$, we obtain the following corollary:

Corollary VII.2. *Let* $f = \sum_i^n f_i$ *be a* μ*-strongly convex function with an* L*-Lipshitz gradient, and suppose each* f_i *is convex, and all gradients* ∇f_i *are* L'*-Lipshitz. Let* θ^* *be the minimizer of* f*, and define* $\sigma^2 := \sum_i |\nabla f_i(\theta^*)|_2^2$.

Suppose we perform gradient descent with the update $\theta_{k+1} = \theta_k - \gamma\sum_i \alpha_i^{(k)}\nabla f_i$*, such that at each iteration,* $\left|\alpha^{(k)} - \mathbb{1}\right|_2^2 \leq r$*. Assume* $\mu > rL'$.

Then we can choose some step size γ *such that for some*

$$k \leq \frac{3(L + 2\sqrt{r}L')^2\log\left(\frac{\mu^2\epsilon_0}{2r\sigma^2}\right)}{\left(\mu - \sqrt{\mu r L'}\right)^2},$$

we have $|\theta_k - \theta^*|_2^2 \leq \frac{4r\sigma^2}{\left(\mu - \sqrt{\mu r L'}\right)^2}$*, where* $\epsilon_0 = |\theta_0 - \theta^*|^2$.

Plugging in our decoding error bound on $|\alpha^* - \mathbb{1}|_2^2$ from Corollary V.3 shows that we can converge to a noise floor of

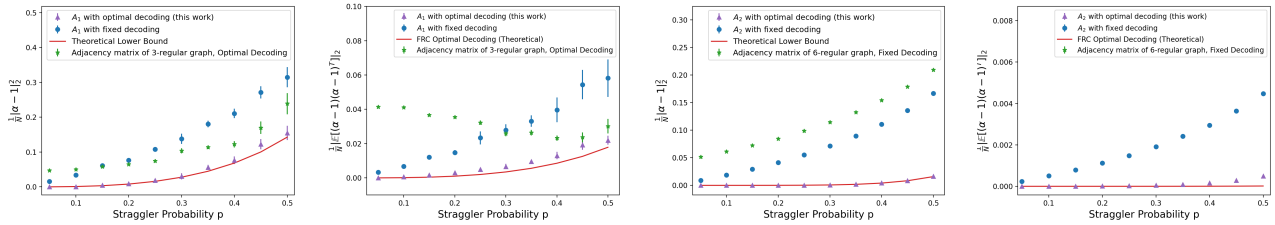
$$\frac{4\sigma^2 n(2d - \lambda)p}{\mu(\sqrt{2d(1-p)}\mu - \sqrt{n(2d-\lambda)pL'})^2}.$$

Remark VII.3. *In the case of a linear regression problem, where* $f_i(\theta) = (a_i^T\theta - b_i)^2$*, if the vectors* $a_i \sim N(0, I_k)$ *and values* $b_i = \langle a_i, \theta\rangle + z_i$ *for* $z_i \sim N(0, \zeta^2)$*, we expect to have* $\mu \approx 2N\left(1 - \sqrt{\frac{k}{N}}\right)^2$*, and* $L' < 10\max\left(\frac{N}{n}, k\right)$ *with high probability [21]. Assuming* $N > 4k$*, Corollary VII.2 shows that for* $p \leq 0.05\min(1, \frac{4N}{nk})$*, we can converge to a noise floor of* $\frac{20pn^2k^2\zeta^2}{N^2}$.

We defer the proof of this proposition to Appendix F.

VIII. Experiments

In this section, we demonstrate empirically that our scheme achieves near-optimal error $\mathbb{E}\left[|\alpha^* - \mathbb{1}|_2^2\right]$ in the random



(a) (b) (c) (d)

Fig. 3. Comparison of variance and covariance of α. All values estimated over 50 runs. Error bars are for the standard deviation of the empirical values, evaluated over 5 experiments. (a) The empirical expectation $\frac{1}{N}\mathbb{E}\left[|\overline{\alpha}-\mathbb{1}|_2^2\right]$ with $m=24$, $d=3$. (b) Norm of the empirical covariance matrix $\frac{1}{N}\left|\mathbb{E}\left[(\overline{\alpha}-\mathbb{1})(\overline{\alpha}-\mathbb{1})^T\right]\right|_2$ with $m=24$, $d=3$. (c) The empirical expectation $\frac{1}{N}\mathbb{E}\left[|\overline{\alpha}-\mathbb{1}|_2^2\right]$ with assignment A_2 with $m=6552$, $d=6$. (d) Norm of the empirical covariance matrix $\frac{1}{N}\left|\mathbb{E}\left[(\overline{\alpha}-\mathbb{1})(\overline{\alpha}-\mathbb{1})^T\right]\right|_2$ with $m=6552$, $d=6$. The points for the Expander code do not appear as they are significantly higher.

stragglers model, and further that our scheme converges in fewer iterations with optimal decoding coefficients than with fixed coefficients. We demonstrate the advantage of our coded approach to uncoded gradient descent, and compare the convergence of our scheme to previous work on coded gradient descent. We run our convergence experiments in both a simulated setting and in a distributed compute cluster, which we describe in more detail in Section VIII-B.

Our experiments are conducted in two different parameter regimes. The first regime is in a setting with $m=24$ machines and $N=60000$ data points, where each data point is replicated $d=3$ times such that the compuational load is $\ell=7500$. The second regime has $m=6552$ machines and $N=6552$ data points, and each data point is replicated $d=6$ times such that $\ell=6$. In the first regime, for our coded approach, we use an assigment matrix A_1 which corresponds to a random 3-regular graph on $n=16$ vertices with $m=24$ edges. While this graph is not vertex-transitive, and hence may result in a biased approximation of the gradient, it represents a practical regime with 24 machines, and is with high probability a good expander. In the second regime, we use an assignment matrix A_2 which corresponds to the degree 6 LPS expander of [19] on $n=2184$ vertices with 6552 edges. We chose this graph because it is the smallest vertex-transitive expander.

In each of these two parameter regimes, we compare the performance the performance of the following four coding schemes:

1) Our coded approach using A_1 or A_2 with optimal decoding.
2) Our coded approach using A_1 or A_2 with fixed decoding.
3) The coded approach of [6]. With $m=24$, we use the for the assignment matrix the adjacency matrix of a random graph on 24 vertices of degree 3. In this regime, we conduct optimal decoding. With $m=6552$, we use a random graph on 6552 vertices of degree 6. Due to the computational complexity of decoding, we use fixed decoding coefficients in this regime.
4) The coded approach of [10] (which uses an FRC) with replication factor $d=3$ or $d=6$.

For fixed decoding, we use the decoding vector w^{fixed} where $w_j^{\text{fixed}}=0$ if j is a straggler and otherwise $w_j^{\text{fixed}}=\frac{1}{d(1-p)}$. In this manner, we have $\mathbb{E}[Aw^{\text{fixed}}]=\mathbb{1}$.

A. The decoding error $\mathbb{E}[|\overline{\alpha^*}-\mathbb{1}|_2^2]$

In our first set of experiments, shown in Figure 3, we compare the decoding error $\mathbb{E}\left[\frac{1}{N}|\overline{\alpha}-\mathbb{1}|_2^2\right]$ and the norm of the covariance $|\mathbb{E}(\overline{\alpha}-\mathbb{1})(\overline{\alpha}-\mathbb{1})^T|_2$ under a p fraction of random stragglers for the first four schemes above. In Figure 3(a)(b), we consider the first regime with $d=3$, and in Figure 3(c)(d) the second regime where $d=6$. We note that the FRC of [4] achieves the theoretical optimum of $\frac{1}{N}\mathbb{E}|\overline{\alpha^*}-\mathbb{1}|_2^2=\frac{p^d}{1-p^d}$, and hence we plot this optimum in place of the results from the FRC. Similarly, for the FRC assigment,we have

$$\left|\mathbb{E}\left[(\overline{\alpha^*}-\mathbb{1})(\overline{\alpha^*}-\mathbb{1})^T\right]\right|_2=\frac{\ell}{N}\mathbb{E}|\overline{\alpha^*}-\mathbb{1}|_2^2.$$

This equation holds because the covariance matrix has zeros everywhere except in entries corresponding to two data points in the same block. In all these three settings, we use $N=n$ data points, such that the computational load is $\ell=6$.

Figure 3 demonstrates that for both assignments A_1 and A_2, our scheme with optimal coefficients achieves near-optimal error $\mathbb{E}\left[\left|\overline{\alpha^*}-\mathbb{1}\right|_2^2\right]$ for small values of p, and significantly outperforms fixed coefficient decoding and the approach of [6].

B. Convergence of Coded Gradient Descent

We compare the performance of coded gradient descent in the four coding schemes listed above in addition to an uncoded scheme which ignores stragglers.

Data. We run gradient descent on a least squares problem $\min_\theta |X\theta-Y|_2^2$, where $X\in\mathbb{R}^{N\times k}$ is chosen randomly with i.i.d. rows from $\mathcal{N}(0,\frac{1}{k}I_k)$, and $\theta\sim\mathcal{N}(0,I_k)$. The observations Y are noisy observations of the form $Y=X\theta+Z$, where $Z\sim\sigma^2\mathcal{N}(0,I_N)$. In our first parameter regime with $m=24$, we use $N=60000$, $k=20000$, and $\sigma=100$. In our second parameter regime with $m=6552$, we use $N=6552$, $k=200$, and $\sigma=1$. We initialize θ at the origin, and let θ_* be the minimizer $(X^TX)^{-1}X^TY$.

Platform and Implementation. In the first regime, we run our experiments on $m = 24$ processors in Stanford's high compute cluster Sherlock, which contains any of the following four processors: Intel E5-2640v4, Intel 5118, AMD 7502, or AMD 7742. We implement the algorithms in Python using MPI4py, an open-source MPI implementation. At each iteration, the PS waits to receive gradient updates from the first $\lceil m(1-p) \rceil$ processors using MPI.Request.Waitany. Then the PS computes optimal or fixed decoding coefficents (as specified by the scheme), takes a gradient step, and sends the next iterate θ to all of the processors. We plot the error $|\theta_t - \theta_*|^2$ after 50 iterations in Figure 4. We start timing once the data has been loaded and the first iteration starts.

In the second regime with $m = 6552$ machines (which is too large for us to test on the Sherlock cluster) we simulate coded gradient descent on a single machine by computing the gradients update used that would be used in the presence of a specified set of stragglers. We artificially select the stragglers independently with probability p. Precisely, our simulations implement Algorithm 2 with a specific input distribution P_β which depends on the coding scheme. Recall that Algorithm 2 is stochastically equivalent to Algorithm 1 if the input distribution P_β equals the distribution of α^*. Hence for optimal decoding with an assignment matrix A, we let the input P_β to Algorithm 2 be the distribution of $\alpha^* = A(p)(A(p)^T A(p))^\dagger A(p)^T \mathbb{1}$ given by Equation (6). Recall here that $A(p)$ is the matrix A where each column is deleted with probablility p. That is, at each iteration, we randomly sample β to be this random vector. For fixed decoding with assignment matrix A, we let P_β be the distribution of Aw^{fixed}. We plot the error $|\theta_t - \theta_*|^2$ after 50 iterations in Figure 5. As per Remark VIII.1 below, in the uncoded approach, we do 300 iterations.

Remark VIII.1. *If the same number of machines are used in both coded and uncoded approaches, but the coded approach has a replication factor of d, then each machine in the coded approach has a gradient computation that is d times bigger. We compensate for this by performing d times as many iterations in the uncoded scheme. Note that if the communication time is the bottleneck, then each iteration of coded gradient descent will take* less *than d times as long as an iteration of uncoded gradient descent: indeed, the communication times should be the same, while the computation time should increase by a factor of d. In this case, we expect the advantage of our approach over an uncoded approach to be greater than our simulations suggest.*

To be fair to all algorithms, for all experiments discussed, we use a grid search to find the best step size. We give more details and show the step size chosen by this grid search in Table IV in Appendix G for all algorithms discussed below.

We observe that our algorithm substantially outperforms the expander code and the uncoded approach, and converges to error comparable with the FRC of [4] (we recall that the FRC of [4] achieves the optimal $\mathbb{E}\left[|\overline{\alpha^*} - \mathbb{1}|_2^2\right]$ for random stragglers, but is substantially sub-optimal for worst-case stragglers). We note that our algorithm in many cases even outperforms the FRC: indeed, Figure 4(a) demonstrates faster convergence, and the table in Figure 4(b) shows that the final error is typically much smaller for our algorithm than for the FRC on the Sherlock cluster. We conjecture that our algorithm is able to outperform the FRC (the theoretical optimum) on a real cluster because the assumption of i.i.d. stragglers is not perfectly correct: indeed, we observe that which machines are straggling tends to stay stagnant throughout a run. We conjecture that the comparatively better performance of our algorithm on worst-case stragglers (relative to the FRC) gives it an advantage in such settings.

IX. Conclusion

In this work, we present an approximate gradient coding scheme based on expander graphs, which performs well both in the adversarial and random straggler settings. We show how to analyze the optimal decoding error of our codes by relating α^* to the connected components in a randomly sparsified graph. We give provable convergence results in both the adversarial and random straggler settings. We conclude with a few open questions.

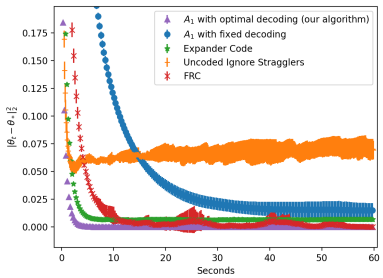
1) We have developed a technique for controlling the variance and covariance of the random variable α^* generated by the optimal decoding coefficients when each machine holds two data blocks, by analyzing the sparsification of random graphs. It is an interesting open question to extend our techniques, or develop new ones, to work for a larger number of data blocks per machine.
2) While our scheme gives the best known error $|\alpha^* - \mathbb{1}|_2^2$ in the adversarial setting given near-optimal error against random stragglers, it could be improved. Is there a coding scheme which achieves near-optimal error $|\alpha^* - \mathbb{1}|_2^2$—that is, decaying like $p^{d-o(d)}$—while simultaneously achieving near-optimal adversarial error—that is, decaying like $\frac{1}{d}$?

Acknowledgment

We thank the anonymous reviewers and Tavor Baharav for helpful comments and suggestions.

References

[1] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," in *Advances in Neural Information Processing Systems*, 2014, pp. 19–27.
[2] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, 2013.
[3] M. Zaharia, A. Konwinski, A. D. Joseph, R. H. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments." in *Osdi*, vol. 8, no. 4, 2008, p. 7.
[4] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *International Conference on Machine Learning*, 2017, pp. 3368–3376.
[5] R. Bitar, M. Wootters, and S. El Rouayheb, "Stochastic gradient coding for straggler mitigation in distributed learning," *IEEE Journal on Selected Areas in Information Theory*, 2020.
[6] N. Raviv, R. Tandon, A. Dimakis, and I. Tamo, "Gradient coding from cyclic mds codes and expander graphs," in *International Conference on Machine Learning*, 2018, pp. 4305–4313.
[7] S. Kadhe, O. O. Koyluoglu, and K. Ramchandran, "Gradient coding based on block designs for mitigating adversarial stragglers," in *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2019, pp. 2813–2817.

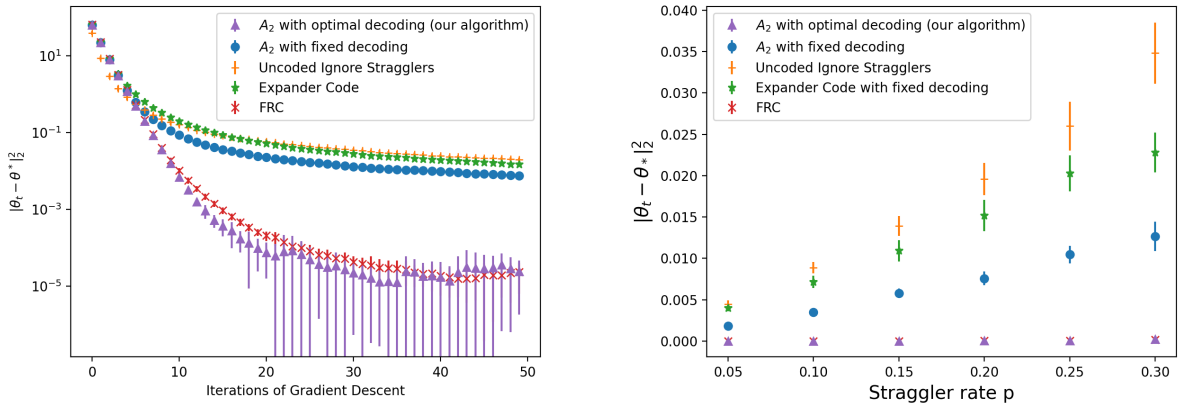


(a)

p	0.05	0.1	0.15	0.2	0.25	0.3
A_1 Optimal	3.4e-30	2.0e-08	8.7e-26	1.2e-26	3.9e-09	6.1e-30
A_1 Fixed	2.5e-3	6.7e-3	6.0e-3	1.9e-2	2.2e-2	2.8e-2
Ignore Stragglers	7.3e-3	2.2e-2	4.2e-2	6.9e-2	1.1e-1	1.4e-1
Expander Optimal	1.2e-4	4.8e-4	2.1e-3	6.6e-3	1.1e-2	2.2e-2
FRC	3.0e-16	2.0e-29	3.5e-30	4.0e-05	5.6e-2	7.4e-4

(b)

Fig. 4. Comparison of coded gradient descent on a distributed cluster with $m = 24$, $N = 60000$. (a) Convergence of gradient descent with $p = 0.2$. (b) $|\theta_t - \theta_*|_2^2$ after 60 seconds. Values are averaged over 8 runs, with error bars for standard deviation.



(a) (b)

Fig. 5. Comparison of simulated gradient descent algorithms with $m = 6552$, $N = 6552$. (a) Convergence of gradient descent with $p = 0.2$; the uncoded approach uses 6x as many iterations as shown. Values are averaged over 20 runs, with error bars for standard deviation. (b) $|\theta_t - \theta_*|_2^2$ after 50 iterations. Values are averaged over 20 runs, with error bars for standard deviation.

[8] Z. Charles, D. Papailiopoulos, and J. Ellenberg, "Approximate gradient coding via sparse random graphs," *arXiv preprint arXiv:1711.06771*, 2017.

[9] S. Wang, J. Liu, and N. Shroff, "Fundamental limits of approximate gradient coding," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 3, pp. 1–22, 2019.

[10] H. Wang, Z. Charles, and D. Papailiopoulos, "Erasurehead: Distributed gradient descent without delays using approximate gradient coding," *arXiv preprint arXiv:1901.09671*, 2019.

[11] Z. Charles and D. Papailiopoulos, "Gradient coding via the stochastic block model," *arXiv preprint arXiv:1805.10378*, 2018.

[12] S. Li, S. M. M. Kalan, A. S. Avestimehr, and M. Soltanolkotabi, "Near-optimal straggler mitigation for distributed gradient methods," in *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2018, pp. 857–866.

[13] W. Halbawi, N. Azizan, F. Salehi, and B. Hassibi, "Improving distributed gradient descent using Reed-Solomon codes," in *2018 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2018, pp. 2027–2031.

[14] L. Chen, H. Wang, Z. Charles, and D. Papailiopoulos, "Draco: Byzantine-resilient distributed training via redundant gradients," in *International Conference on Machine Learning*, 2018, pp. 903–912.

[15] M. Ye and E. A. Abbe, "Communication-computation efficient gradient coding," in *35th International Conference on Machine Learning, ICML 2018*. International Machine Learning Society (IMLS), 2018, p. 9716p.

[16] R. K. Maity, A. S. Rawat, and A. Mazumdar, "Robust gradient descent via moment encoding and LDPC codes," in *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2019, pp. 2734–2738.

[17] J. Haddock, D. Needell, E. Rebrova, and W. Swartworth, "Stochastic gradient descent variants for corrupted systems of linear equations," in *2020 54th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2020, pp. 1–6.

[18] D.-A. Alistarh, Z. Allen-Zhu, and J. Li, "Byzantine stochastic gradient descent," *Advances in Neural Information Processing Systems*, vol. 2018, 2018.

[19] A. Lubotzky, R. Phillips, and P. Sarnak, "Explicit expanders and the Ramanujan conjectures," in *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, 1986, pp. 240–246.

[20] D. Needell, R. Ward, and N. Srebro, "Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm," in *Advances in neural information processing systems*, 2014, pp. 1017–1025.

[21] D. Chafaı, D. Chafä, O. Guédon, G. Lecue, and A. Pajor, "Singular values of random matrices," *Lecture Notes*, 2009.

[22] S. Hoory, N. Linial, and A. Wigderson, "Expander graphs and their applications," *Bulletin of the American Mathematical Society*, vol. 43, no. 4, pp. 439–561, 2006.

[23] B. Spang and M. Wootters, "Unconstraining graph-constrained group testing," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[24] D. Kleinman and M. Athans, "The design of suboptimal linear time-varying systems," *IEEE Transactions on Automatic Control*, vol. 13, no. 2, pp. 150–159, 1968.