# Gradient Coding Using the Stochastic Block Model

Zachary Charles
University of Wisconsin-Madison
Department of Electrical and Computer Engineering
Madison, WI 53706
Email: zcharles@wisc.edu

Dimitris Papailiopoulos
University of Wisconsin-Madison
Department of Electrical and Computer Engineering
Madison, WI 53706
Email: dimitris@ece.wisc.edu

***Abstract*—Gradient descent and its many variants, including mini-batch stochastic gradient descent, form the algorithmic foundation of modern large-scale machine learning. Due to the size and scale of modern data, gradient computations are often distributed across multiple compute nodes. Unfortunately, such distributed implementations can face significant delays caused by straggler nodes, *i.e.,* nodes that are much slower than average. Gradient coding is a new technique for mitigating the effect of stragglers via algorithmic redundancy. While effective, previously proposed gradient codes can be computationally expensive to construct, inaccurate, or susceptible to adversarial stragglers. In this work, we present the stochastic block code (SBC), a gradient code based on the stochastic block model. We show that SBCs are efficient, accurate, and that under certain settings, adversarial straggler selection becomes as hard as detecting a community structure in the multiple community, block stochastic graph model.**

## I. Introduction

In order to scale up machine learning to large data sets, we often use distributed implementations of traditional first-order optimization algorithms. While distributed algorithms can theoretically achieve substantial speedups, in practice the parallelization gains often fall short of the optimal speedup gains predicted by theory [1], [2]. One of the sources of this phenomenon is the presence of *stragglers*. These are compute nodes whose runtime is much higher than the runtime of most other nodes in the system.

There has been significant recent work in reducing the effect of stragglers in distributed algorithms. Current approaches include replicating jobs across nodes [3] and dropping stragglers in settings where the system can tolerate errors [4]. More recently, coding theory has gained traction as a way to speed up distributed. It has been used to reduce the runtime of the shuffling phase of MapReduce [5], make distributed matrix multiplication more efficient [6], and reduce the effect of stragglers in certain machine learning computations [7].

Gradient coding, a straggler-mitigating technique for distributed gradient-based methods, was first proposed in [8] and was later extended in [9]. While gradient coding was initially used for exact reconstruction of the sum of gradients, it was later extended to approximate reconstructions using expander and Ramanujan graphs [9]. Such graphs can be expensive to compute in practice, especially for large numbers of compute nodes, and the approximation eror that they introduce during the gradient recovery problem can be large.

Some efficient and simple approximate gradient codes were studied in [10], including fractional repetition codes (FRCs) and Bernoulli gradient codes (BGCs). FRCs were first presented in [8], but only for exact gradient coding. FRCs achieve very small approximation error when the straggler are chosen at random, but suffer when the stragglers are selected adversarially. However, [10] shows that adversarial straggler selection is NP-hard in general, which implies the existence of gradient codes that might be robust to adversarial stragglers under polynomial-time bounded adversaries assuming some widely accepted computational conjectures.

The main problem we tackle in this paper is whether it is possible to design gradient codes that 1) are efficiently computable, 2) achieve reconstruction error similar to FRCs for random stragglers, 3) can be resistant to adversarial stragglers.

### A. Our Contributions

In this work, we present the *stochastic block code* (SBC), a new gradient code that combines the small reconstruction error of FRCs under random straggler selection with the randomness of BGCs, which is some cases leads to robustness against adversarial stragglers. SBCs are based on the stochastic block model from random graph theory. The code is more effective than BGCs when stragglers are chosen randomly, but more difficult for an adversary to attack than FRCs. We give explicit bounds on the approximation error of SBCs under random straggler selection and show that certain adversarial attacks on SBCs are computationally as hard as recovery problems in community detection for regimes where no polynomial time algorithm is known to exist. Finally, we empirically evaluate SBCs and show that they are capable of achieving small approximation error, even when a constant fraction of the compute nodes are stragglers.

## II. Preliminaries

We consider a distributed master-worker setup of $n$ compute nodes, each of which is assigned $s$ tasks. A pictorial description is given in Figure 1. Each compute node locally computes a set of assigned functions and sends a linear combination of these functions to the master node.

The goal of the master node is to compute the sum of $k$ functions

$$f(\mathbf{x}) = \sum_{i=1}^{k} f_i(\mathbf{x}) = \mathbf{f}^T \mathbf{1_k} \tag{1}$$

978-1-5386-4780-6/18/$31.00©2018 IEEE

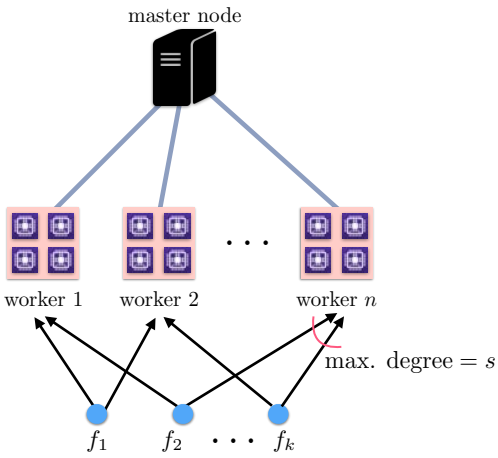



Fig. 1: A master-worker distributed system where each compute node has multiple cores.

in a distributed way, where $f_i : \mathbb{R}^d \to \mathbb{R}^w$ and each $f_i$ can be assigned to and computed locally by any of the $n$ compute nodes. Here, $\mathbf{f} := [f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})]^T$. We denote the output of compute node $j$ by $y_j$.

Due to the straggler effect, we assume that the master only has access to the output of $r < n$ non-straggler compute nodes. By not waiting for the output of straggler nodes, we can drastically improve the runtime of distributed algorithms. If we wish to exactly recover $f(\mathbf{x})$, then we need $r \geq k-s+1$ [8]. However, in practice we may only need to approximately recover $f(\mathbf{x})$, which we can do with significantly fewer non-straggler nodes.

The above setup is relevant to distributed learning algorithms, where we often wish to find some model $\mathbf{x}$ by minimizing

$$\ell(\mathbf{x}) = \sum_{i=1}^{k} \ell(\mathbf{x}; \mathbf{z}_i).$$

Here $\{\mathbf{z}_i\}_{i=1}^k$ are our training samples and $\ell(\mathbf{x}; \mathbf{z})$ is a loss function measuring the accuracy of the model $\mathbf{x}$ with respect to the data point $\mathbf{z}$. In order to find a model that minimizes the sum of losses $\ell(\mathbf{x})$, we often use first-order, or gradient-based, methods. In these algorithms, at every distributed iteration, we would need to compute the gradient of $\ell(\mathbf{x})$ (or a minibatch of the training samples), given by $\nabla\ell(\mathbf{x}) = \sum_{i=1}^k \nabla\ell(\mathbf{x}; \mathbf{z}_i)$. Letting $f_i(\mathbf{x}) = \nabla\ell(\mathbf{x}; \mathbf{z}_i)$, we arrive at the setup in (1). Note that this kind of setup applies to both mini-batch SGD and full-batch gradient descent.

### A. Gradient Codes

A *gradient code* involves a choice of function assignments per compute node, messages sent from the compute node to the master, and a *decoding* method used by the master node to approximately reconstruct the true gradient. For simplicity, we assume that the master node can only compute linear combinations of the output of the non-straggler compute nodes.

Suppose we have $k$ functions to compute and $k$ compute nodes. We wish to construct a $k \times k$ *function assignment matrix* $\mathbf{G}$ that describes which functions each node computes and what message the node passes back to the master node. The column $\mathbf{g}_j$ corresponds to compute node $j$. The entries of column $j$ correspond to the coefficients of the linear combination of functions that the compute node sends back to the master node. In other words, if column $j$ has support $U$, then node $j$ computes $f_i$ for $i \in U$ and has output $y_j = \sum_{i \in U} f_i$ (or more generally, a linear combination of the computed functions). Let $\mathbf{y} = [y_1, \ldots, y_k]^T$.

For the decoding, we assume that we have $r$ non-straggler nodes with indices given by $T$. Decoding the gradient code corresponds to designing some vector $\mathbf{v} \in \mathbb{R}^k$ with support in $T$. The approximation $\tilde{f}$ to $f$ is then given by

$$\tilde{f} = \sum_{i=1}^{k} y_i \mathbf{v}_i = \mathbf{y}^T \mathbf{v}.$$

We define the error of our approximation in terms of the vector $\mathbf{v}$. Note that we have

$$(\tilde{f} - f)^2 = \|\mathbf{f}^T \mathbf{G}\mathbf{v} - \mathbf{f}^T \mathbf{1}_k\|_2^2 \leq \|\mathbf{f}\|_2^2 \|\mathbf{G}\mathbf{v} - \mathbf{1}_k\|_2^2.$$

Since the error depends on $\mathbf{f}$ which we do not know a priori, we instead define the approximation error $\mathrm{err}(\mathbf{v})$ of our gradient code by

$$\mathrm{err}(\mathbf{v}) := \|\mathbf{G}\mathbf{v} - \mathbf{1}_k\|_2^2$$

where $\mathbf{v}$ has support only among the non-straggler nodes.

Let $T$ denote the indices of the non-straggler columns. We define the $k \times k$ non-straggler matrix $\mathbf{G}_T$ as having the same entries as $\mathbf{G}$ in the non-straggler columns and having 0 in the straggler columns. This implies $\mathrm{err}(\mathbf{v}) = \|\mathbf{G}_T\mathbf{v} - \mathbf{1}_k\|_2^2$. The optimal decoding vector $\mathbf{v}_{\text{opt}}$ is defined by

$$\mathbf{v}_{\text{opt}} := \arg\min_{\mathbf{v}} \|\mathbf{G}_T\mathbf{v} - \mathbf{1}_k\|_2^2 = \arg\min_{\mathbf{v}:\mathrm{supp}(\mathbf{v}) \subseteq T} \|\mathbf{G}\mathbf{v} - \mathbf{1}_k\|_2^2.$$

Standard facts about the pseudoinverse of a matrix then imply $\mathbf{v}_{\text{opt}} = \mathbf{G}_T^+ \mathbf{1}_k$. This is one of the possibly infinite many optimal solutions to the above least squares recovery problem. In practice, it may not be feasible to compute $\mathbf{v}_{\text{opt}}$ since it involves computing the pseudo-inverse of a matrix. Moreover, $\mathbf{v}_{\text{opt}}$ is difficult to analyze theoretically since it involves the pseudoinverse of a random matrix. For these two reasons, we will often use and analyze simpler, more efficient decoding methods.

## III. Gradient Coding Methods

In this section, we briefly discuss two previously proposed gradient codes and their properties.

### A. Fractional Repetition Codes

Fractional repetition codes (FRCs) were first proposed in [8] for exact reconstruction of $f$. They were later shown in [10] to be useful in the context of approximate gradient coding. Suppose we are given $k$ tasks and $k$ workers and a desired number $s$ of tasks per worker, where we assume $s|k$ without

loss of generality. We define the $k \times k$ function assignment matrix $\mathbf{G}$ of an FRC by

$$\mathbf{G} = \begin{pmatrix} \mathbf{1}_{s\times s} & \mathbf{0}_{s\times s} & \dots & \mathbf{0}_{s\times s} \\ \mathbf{0}_{s\times s} & \mathbf{1}_{s\times s} & \dots & \mathbf{0}_{s\times s} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{s\times s} & \mathbf{0}_{s\times s} & \dots & \mathbf{1}_{s\times s} \end{pmatrix}.$$

Note that if columns 1 and 2 are both non-stragglers, then using both columns does not help our decoding. To decode, we first look at indices $1, \dots, s$. If any are non-stragglers, we select the first such index $t$ and let $\mathbf{v}_t = 1$. We repeat this with indices $s+1, \dots, 2s$, etc. As long as there is a non-straggler in each of these blocks, a simple calculation shows that $\mathrm{err}(\mathbf{v}) = 0$.

FRCs have small error with high probability, if the stragglers are selected uniformly at random, even with a constant fraction of stragglers [10]. However, the worst case error of FRCs is quite large, as we discuss in Section IV-C. To remedy this, we use gradient codes that employ randomness.

### B. Bernoulli Gradient Codes

Bernoulli gradient codes (BGCs) were first introduced in [10]. We construct $\mathbf{G}$ where $\mathbf{G}_{i,j}$ is Bernoulli with probability $p$. We typically take $p = s/k$, so that each node computes roughly $s$ functions. While we could use optimal decoding and find $\mathbf{v}_{\text{opt}}$, there are more efficient (though slightly less accurate) decoding methods. If $T$ is the set of non-stragglers where $|T| = r$, then somewhat small approximation error can be achieved by setting $\mathbf{v}_i = \frac{k}{rs}$ if $i \in T$, and 0 otherwise. In [10], it was shown that under this decoding, with high probability the error satisfies $\mathrm{err}(\mathbf{v}) = O(k/rp)$. While less accurate than FRCs, BGCs do not seem to be as vulnerable to adversarial straggler selection.

## IV. Stochastic Block Codes

Our goal is to construct a gradient code that combines the small error to random stragglers of FRCs and the potential resilience against polynomial-time adversaries of BGCs. We propose a kind of interpolation between the two codes that we refer to as the *stochastic block code* (SBC). For such codes, we use a function assignment matrix $\mathbf{G}$ with block structure

$$\mathbf{G} = \begin{pmatrix} \mathbf{P} & \mathbf{Q} & \dots & \mathbf{Q} \\ \mathbf{Q} & \mathbf{P} & \dots & \mathbf{Q} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{Q} & \mathbf{Q} & \dots & \mathbf{P} \end{pmatrix}. \tag{2}$$

Here, each $\mathbf{P}, \mathbf{Q}$ is an $s \times s$ matrix with Bernoulli random entries. The entries of each $\mathbf{P}$ are Bernoulli with probability $p$, while the entries of each $\mathbf{Q}$ are Bernoulli with probability $q$. In other words, $\mathbf{G}$ has a stochastic block model structure with $\frac{k}{s}$ partitions of size $s$ with probabilities $p$ within a community and probabilities $q$ between communities. In the following, we will assume $p \geq q$.

Typically we will consider the case where $p$ is close to 1 and $q$ is small, so that $\mathbf{G}$ has blocks similar to $\mathbf{1}_{s\times s}$ on the diagonal and only a small number of ones outside of the diagonal blocks. This model specializes FRCs when $q = 1, p = 0$ and BGCs when $q = p$. By varying $p$ and $q$, we can interpolate between FRCs and BGCs.

### A. Decoding

We would like to devise a computationally efficient method for decoding. Let $S_1 = \{1, \dots, s\}, S_2 = \{s+1, \dots, 2s\}, \dots S_{k/s} = \{k-s+1, \dots, k\}$. Given the set of non-straggler indices $T$, let $T_i = S_i \cap T$. Our decoding method works as follows. For each $T_i \neq \emptyset$, we will select $s_i \in T_i$ at random. We will then add up the columns $\mathbf{g}_{s_i}$ to form a vector $\mathbf{w}$ and then scale $\mathbf{w}$ so that it is close to $\mathbf{1}_k$. The idea is that if we select one column from each block, then their sum should be close to a multiple of the all ones vector. A formal algorithm is given in Figure **??**.

**Algorithm 1:** Stochastic block decoding.

**Input** : The indices $T_1, \dots, T_{k/s}$ of non-straggler nodes.
**Output**: A vector $\mathbf{v}$ that describes the decoding strategy.

$\mathbf{v} = \mathbf{0}_k$;
**if** $p + (\frac{k}{s} - 1)q < 2$ **then**
  $\beta = 1$;
**else**
  $\beta = p + (\frac{k}{s} - 1)q$;
**end**
**for** $i = 1$ **to** $k/s$ **do**
  **if** $T_i \neq \emptyset$ **then**
    Pick $t_i \in T_i$ uniformly at random;
    $\mathbf{v}_{t_i} = 1/\beta$;
  **end**
**end**
return $\mathbf{v}$;

Recall that if we want to use $\mathbf{v}$ to compute an approximation to $f$, we take $\tilde{f} = \mathbf{y}^T\mathbf{v}$, where $\mathbf{y}$ is the vector of outputs of the nodes. Since $\mathbf{v}$ only has non-zero indices corresponding to non-straggler nodes, we can compute $\tilde{f}$ only using the non-straggler nodes.

This decoding corresponds to selecting one non-straggler column (if it exists) from each block of $s$ columns and then adding these columns together. We select at most $\frac{k}{s}$ columns. Let $\mathbf{w}$ denote the sum of the selected columns. We return $\beta^{-1}\mathbf{w}$, where $\beta$ is the expected value of each entry of $\mathbf{v}$. When $p = 1, q = 0$, this method gives us the optimal decoding method for FRCs discussed above.

When $p \approx 1$ and $q < \frac{s}{k}$, scaling by $\beta^{-1}$ is not necessarily beneficial. In this regime, each entry of $\mathbf{w}$ is 1 with high probability, while a few entries may be larger integers if any of the $\mathrm{Ber}(q)$ entries are non-zero. Scaling by $\beta^{-1}$ then introduces errors in to all the entries that are 1 in $\mathbf{w}$. Intuitively, if $X$ is 1 with high probability and 2 with small probability, the variable $Y = X/\mathbb{E}[X]$ has expected value 1, but will never equal 1, while $X$ will equal 1 with high probability. Thus, not scaling by $\beta$ implies $\mathbf{v} \approx \mathbf{1}_k$ with high probability and has the added benefit of simplifying the analysis.

One could improve this algorithm by averaging over the non-straggler columns in each $T_i$, and then adding up the

averages. When $r$ is large, this approach is better than stochastic block decoding, at the expense of being more difficult to analyze theoretically.

### B. Random Stragglers

Suppose we use an SBC with stochastic block decoding and that the $r$ non-straggler indices $T$ are selected uniformly at random from all subsets of $\{1, \ldots, k\}$ of size $r$. We are particularly interested in the setting where the number of stragglers is a constant fraction of $k$, as this is when previous theory such as that in [8] breaks down.

Because of the straggler effect, we only have access to the matrix $\mathbf{G}_T$ where zeros have been filled in to all straggler columns. Note that $\mathbf{G}$ is a random matrix, but even fixing $\mathbf{G}$, $\mathbf{G}_T$ is random. While the worst-case error may still be large, as it is for FRCs, the average case is much better. We have the following theorem about the reconstruction error.

*Theorem 4.1:* Suppose that $s \geq 2\ln(k)k/r$, $1-p \leq k^{-2}$ and that $q = \frac{\gamma}{k}$ for $\gamma < s$. If we apply stochastic block decoding to a SBC with randomly selected stragglers, then with probability at least $1 - \frac{4}{k}$, this produces a vector $\mathbf{v}$ such that

$$\mathrm{err}(\mathbf{v}) \leq 14\sqrt{\frac{\ln(k)\gamma}{s}} \max\left\{k\left((1-p) + \frac{\gamma}{s}\right), 3\ln(k)\right\}.$$

All our proofs can be found in the long version of this paper [11].

Letting $p = 1$, we get the following corollary concerning SBCs that are close to FRCs.

*Corollary 4.2:* Suppose that $s \geq 2\ln(k)k/r$ and $q = \frac{\gamma}{k}$ for $\gamma \leq \frac{3\ln(k)s}{k}$. If the stragglers are selected randomly, then with probability at least $1 - \frac{3}{k}$, applying stochastic block decoding method to such a SBC produces a vector $\mathbf{v}$ such that

$$\mathrm{err}(\mathbf{v}) \leq O\left(\frac{\ln^2(k)}{\sqrt{k}}\right).$$

The above theorem required $q < \frac{s}{k}$. When $(\frac{k}{s} - 1)q \gg p$, we derive the following theorem.

*Theorem 4.3:* Suppose that $s \geq 2\ln(k)k/r$ and that $(\frac{k}{s}-1)(1-q)q \geq \ln(k)$. If the stragglers are selected randomly, then with probability at least $1 - \frac{2}{k}$, applying stochastic block decoding to a SBC produces a vector $\mathbf{v}$ satisfying

$$\mathrm{err}(\mathbf{v}) \leq \frac{16\ln(k)s^2}{kq^2}\left((1-p)p + \left(\frac{k}{s} - 1\right)(1-q)q\right).$$

By setting $p = q$, we can analyze the error of BGCs under stochastic block decoding.

*Corollary 4.4:* Suppose that $s \geq 2\ln(k)k/r$ and $p \geq s\ln(k)/k$. If the stragglers are selected randomly, then with probability at least $1 - \frac{2}{k}$, applying the stochastic block decoding method to a BGC with probability $p$ produces a vector $\mathbf{v}$ satisfying

$$\mathrm{err}(\mathbf{v}) \leq \frac{16\ln(k)s(1-p)}{p}.$$

### C. Adversarial Stragglers

Suppose now that we have $r$ non-stragglers, but they are selected adversarially by some polynomial-time adversary. When $p = 1$ and $q = 0$, an adversary trying to maximize $\mathrm{err}(\mathbf{v})$ would try to select entire blocks from $\mathbf{G}$ with the same function assignments. This strategy is efficient for an adversary to compute, even if the adversary views a permuted $\mathbf{G}$, and leads to a worst-case error of $\mathrm{err}(\mathbf{v}) = k - r$ [10].

The deterministic block structure of $\mathbf{G}$ when $p = 1, q = 0$ is what allows an adversary to so easily find a worst-case set of stragglers. In general, the task of selecting $k - r$ stragglers to maximize $\mathrm{err}(\mathbf{v})$ is NP-Hard [10]. When $p = 1$ and $q \ll p$, an adversary could still achieve relatively high error by selecting adversaries from contiguous blocks in the block structure in (2). This task is equivalent to that of *community detection* in the stochastic block model.

*Definition 4.5:* We say that a community detection algorithm has accuracy $\alpha \in [0, 1]$ if at most $(1 - \alpha)k$ nodes are misclassified with probability tending to 1 as $k \to \infty$.

We say that a community detection algorithm has *exact recovery* if it has accuracy $\alpha = 1$ and that it has *weak recovery* if it has accuracy $\alpha = \frac{1}{k} + \epsilon$ for $\epsilon > 0$. Weak recovery means that the algorithm performs better than randomly assigning community labels.

There has been significant work in understanding when exact and weak recovery are possible. For exact recovery, suppose that we have the stochastic block model in (2) with $p = a\log(k)/k, q = b\log(k)/k$. Then exact recovery is possible exactly when $\sqrt{a} - \sqrt{b} > \sqrt{k/s}$ [12]. This implies the following theorem.

*Theorem 4.6:* Suppose we use a stochastic block code $\mathbf{G}$ with probabilities $p, q$. Then there is no algorithm that can determine the block structure in (2) with accuracy $\alpha = 1$ as long as $\sqrt{p} - \sqrt{q} < \sqrt{\log(k)/s}$.

Thresholds for weak recovery are also known, but not in as much generality. In [13], the authors define the signal-to-noise ratio SNR of a community detection problem with $p = a/k, q = b/k$ and $m$ communities by

$$\mathrm{SNR} = \frac{(a-b)^2}{m(a + (m-1)b)}.$$

Suppose that $p = a/k, q = a/k$ and we have 2 communities. Then weak reconstruction is possible exactly when $(a-b)^2 > 2(a+b)$ [14]. For $m$ communities, weak recovery is possible when $(a-b)^2 > m(a + (m-1)b)$. However, the converse does not necessarily hold [13]. In general, community detection seems to be more difficult for smaller values of $(a-b)^2/m(a+(m-1)b)$. This leads to the following theorem.

*Theorem 4.7:* An adversary can determine the block structure in (2) with accuracy better than random iff weak recovery is possible. In particular, for $s = \frac{k}{2}$, this is possible if and only if $k(p-q)^2 > 2(p+q)$.

We would like to note that although there exist regimes for which identifying the community structure in a SBC can be computationally intractable, we have not identified a set of parameters such that worst case straggler detection is NP-hard

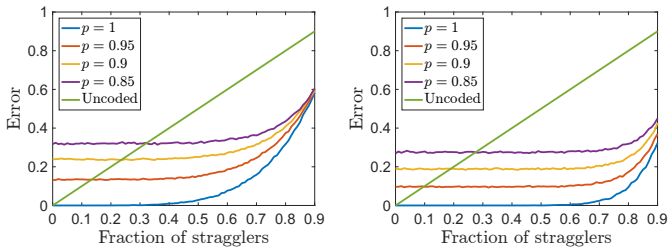
for SBCs, while average case reconstruction error is small. We however think that the connection to community detection is a very interesting direction that can lead to such results.

## V. Empirical Results

In this section, we compare the empirical error of SBCs for different settings of $p$ and $q$ using the stochastic block decoding method and using the optimal decoding method. We are interested in how the error changes as a function of the number of non-stragglers $r$. Recall that the error of a decoding vector $\mathbf{v}$ is given by $\text{err}(\mathbf{v}) = \|\mathbf{G}\mathbf{v} - \mathbf{1}_k\|_2^2$. We compare this to the uncoded error, which equals the fraction of stragglers.

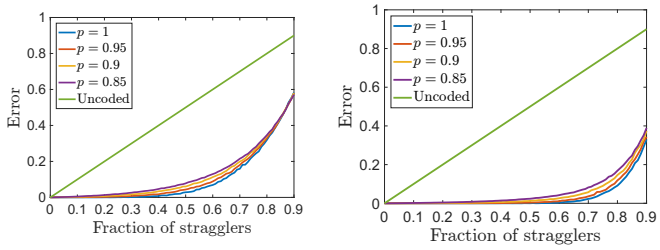
In Figure 2, we plot $\text{err}(\mathbf{v})/k$ where $\mathbf{v}$ comes from stochastic block decoding. We use varying values of $p$ and set $q$ so that the expected number of non-zero entries equals $s$. As our theory suggests, the error increases roughly proportionally to $(1-p)p$ for reasonable $\epsilon$. As $\epsilon$ approaches 1, $p$ has a smaller effect on the error.

In Figure 3, we plot $\text{err}(\mathbf{v}_{\text{opt}})$ for the same values of $p$ and $s$. In this case, the error is less sensitive to $p$ than in stochastic block decoding. Moreover, even when $p = 0.85$, we achieve substantially smaller error than in the uncoded case.




(a) $s = 5$ (b) $s = 10$

Fig. 2: A plot of the average error $\text{err}(\mathbf{v})/k$ over 5000 trials. We take $k = 100$, $r = (1-\epsilon)k$ for varying fractions of stragglers $\epsilon$. The figure on the left has $s = 5$ while the figure on the right has $s = 10$.




(a) $s = 5$ (b) $s = 10$

Fig. 3: A plot of the average error $\text{err}(\mathbf{v}_{\text{opt}})/k$ over 5000 trials. We take $k = 100$, $r = (1-\epsilon)k$ for varying fractions of stragglers $\epsilon$. The figure on the left has $s = 5$ while the figure on the right has $s = 10$.

## VI. Conclusion

In this work, we presented a new gradient code, the stochastic block code. The code is efficiently computable and we provided an efficient decoding method for approximate gradient recovery. The code can be tuned according to whether we care more about random or adversarial stragglers and interpolates between previously proposed codes like FRCs and BGCs. We gave theoretical bounds on the error of SBCs and showed empirically that it can achieve close to zero error in the presence of large numbers of random stragglers. We also show that SBCs can be more robust to adversarial stragglers than FRCs.

Our main results consider a simplified decoding method. In some settings, optimal decoding may be computationally feasible. Understanding optimal decoding error could lead to improved gradient codes. Other open questions involve determining lower bounds on the error of gradient codes and finding optimal codes for a given column sparsity of the function assignment matrix.

## References

[1] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le *et al.*, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1223–1231.

[2] H. Qi, E. Sparks, and A. Talwalkar, "Paleo: A performance model for deep neural networks," in *International Conference on Learning Representations*, 2017. [Online]. Available: \url{https://openreview.net/forum?id=SyVVJ85lg}

[3] N. B. Shah, K. Lee, and K. Ramchandran, "When do redundant requests reduce latency?" *IEEE Transactions on Communications*, vol. 64, no. 2, pp. 715–722, 2016.

[4] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica, "Effective straggler mitigation: Attack of the clones." in *NSDI*, vol. 13, 2013, pp. 185–198.

[5] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "Coded mapreduce," in *Communication, Control, and Computing (Allerton), 2015 53rd Annual Allerton Conference on*. IEEE, 2015, pp. 964–971.

[6] S. Dutta, V. Cadambe, and P. Grover, "Short-dot: Computing large linear transforms distributedly using coded short dot products," in *Advances In Neural Information Processing Systems*, 2016, pp. 2100–2108.

[7] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," in *Information Theory (ISIT), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 1143–1147.

[8] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding," *arXiv preprint arXiv:1612.03301*, 2016.

[9] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic mds codes and expander graphs," *arXiv preprint arXiv:1707.03858*, 2017.

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

[11] Z. Charles and D. Papailiopoulos, "Gradient coding via the stochastic block model," 2018. [Online]. Available: https://tinyurl.com/ydyyb252

[12] E. Mossel, J. Neeman, and A. Sly, "Reconstruction and estimation in the planted partition model," *Probability Theory and Related Fields*, vol. 162, no. 3-4, pp. 431–461, 2015.

[13] E. Abbe, "Community detection and stochastic block models: recent developments," *arXiv preprint arXiv:1703.10146*, 2017.

[14] E. Abbe and C. Sandon, "Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery," in *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*. IEEE, 2015, pp. 670–688.

[15] D. S. Mitrinovic, J. Pecaric, and A. M. Fink, *Classical and new inequalities in analysis*. Springer Science & Business Media, 2013, vol. 61.

[16] E. J. Candes and Y. Plan, "A probabilistic and ripless theory of compressed sensing," *IEEE Transactions on Information Theory*, vol. 57, no. 11, pp. 7235–7254, 2011.