PhaseCode: Fast and Efficient Compressive Phase Retrieval based on Sparse-Graph Codes

Ramtin Pedarsani, Kangwook Lee, and Kannan Ramchandran
Dept. of Electrical Engineering and Computer Sciences
University of California, Berkeley
{ramtin, kw1jjang, kannanr}@eecs.berkeley.edu

***Abstract*— We consider the problem of recovering a complex signal $x \in \mathbb{C}^n$ from m intensity measurements of the form $|a_i x|,\ 1 \leq i \leq m$, where a_i is a measurement row vector. Our main focus is on the case where the measurement vectors are unconstrained, and where x is exactly K-sparse, or the so-called general compressive phase-retrieval problem.**

We introduce *PhaseCode*, a novel family of fast and efficient algorithms (that includes *Unicolor PhaseCode* and *Multicolor PhaseCode*) that are based on a sparse-graph coding framework. As one instance, our *Unicolor PhaseCode* algorithm can provably recover, with high probability, all but a tiny 10^{-7} fraction of the significant signal components, using at most $m = 14K$ measurements, which is a small constant factor from the fundamental limit, with an optimal $\mathcal{O}(K)$ decoding time and an optimal $\mathcal{O}(K)$ memory complexity. We provide extensive simulation results that validate the practical power of our proposed algorithms. A key contribution of our work is the novel use of coding-theoretic tools like density evolution methods for the design and analysis of fast and efficient algorithms for compressive phase-retrieval problems. This contrasts and complements popular approaches to the phase retrieval problem based on alternating-minimization, convex-relaxation, and semi-definite programming.

I. Introduction

The generalized phase retrieval problem is to construct a signal $x \in \mathbb{C}^n$ from the magnitude of linear measurements $y = |Ax|, x \in \mathbb{C}^n$, where $A \in \mathbb{C}^{m \times n}$ is the measurement matrix, and the magnitude is taken on each element of the vector Ax. This problem is motivated by many applications in which one can measure only the intensity of the measurements, such as optics [16], X-ray crystallography [4], astronomy [5], and ptychography [9].

In this paper, we study the compressive phase-retrieval of sparse signals, targeting the case where x is K-sparse. [1]

The phase-retrieval problem has been studied extensively over several decades. Here, we provide a brief review of related works, and refer the readers to [21] for a more extensive literature review. A large body of literature is dedicated to the phase-retrieval problem for the case where the signal to be recovered has no structure and is non-sparse. "Phaselift" proposed by Candes *et al.* [7] and "PhaseCut" proposed by Waldspurger *et al.* [18] are examples of convex relaxation methods to solve the problem via semi-definite programming using $\mathcal{O}(n \log(n))$ measurements. In [15], the authors propose an efficient algorithm based on alternating minimization that reconstructs the signal with $\mathcal{O}(n \log(n)^3)$ measurements.

To the best of our knowledge, the first algorithm for compressive phase retrieval was proposed by Moravec *et al.* in [6]. This approach requires knowledge of the ℓ_1 norm of the signal, making it impractical in most scenarios. The authors in [17] showed that $4K - 1$ measurements are theoretically sufficient to reconstruct the signal, but did not propose any algorithm. The "PhaseLift" method is also proposed for the sparse case in [12] and [14], requiring $\mathcal{O}(K^2 \log(n))$ intensity measurements, and having a computational complexity of $\mathcal{O}(n^3)$, making the method less practical for large-scale applications. Compressive phase-retrieval via generalized approximate message passing (PR-GAMP) is proposed in [8], with good performance in both runtime and noise robustness shown via simulations without theoretical proofs.

Jaganathan *et al.* consider the phase retrieval problem from Fourier measurements [10], [11]. They propose an SDP-based algorithm, and show that the signal can be provably recovered with $\mathcal{O}(K^2 \log(n))$ Fourier measurements [10]. Recently, Cai *et al.* propose proposed SUPER algorithm for compressive phase-retrieval in [1]. The SUPER algorithm uses $\mathcal{O}(K)$ measurements and features $\mathcal{O}(K \log(K))$ complexity with a zero-error-floor asymptotically.

A. Main Contribution

The key contribution of this work is in the introduction of coding theory techniques such as density evolution and sparse-graph codes for compressive phase-retrieval problem. This allows us to come up with a provably efficient and fast PhaseCode family of algorithms that are order-optimal in terms of number of measurements needed, time-complexity, and memory-complexity, which are all $\mathcal{O}(K)$. Furthermore, we provide precise constants for the number of measurements needed to achieve a targeted reliability as defined in Section III. *To the best of our knowledge, this is the first work that provides precise constants for the number of measurements that are small factors from the fundamental lower bound.* As a specific operating point, our proposed Unicolor PhaseCode algorithm can provably recover a frac-

[1] This is easily extended, as is well known, to the case where the signal x is sparse w.r.t. some other basis, such as a wavelet, but in the interests of conceptual clarity, we will not consider such extensions in this work. Our framework also extends to "Fourier-friendly" settings (See [21]), which will not be considered here due to lack of space.

tion of at least $1-10^{-7}$ of the active signal components with $14K$ measurement, with an asymptotically high reliability of $1-\mathcal{O}(1/K)$. This is one instance of an entire family of trade-offs between the number of measurements needed and the fraction of non-zero signal components that can be recovered using PhaseCode.

We also provide simulation results that validate our theoretical findings, and verify the close match between theory and practice. In this regard, we go beyond the UniColor PhaseCode algorithm, which comes with strong theoretical guarantees, to the MultiColor PhaseCode algorithm, which outperforms the UniColor PhaseCode algorithm empirically, but whose theoretical analysis remains open, and will be part of our ongoing and future work. Concretely, for the same instance cited earlier for the UniColor PhaseCode operating point (recovery of $1-10^{-7}$ fraction of the active signal components with $14K$ measurements), the more efficient MultiColor PhaseCode algorithm is shown to need only about $11K$ measurements. Simulations confirm the runtime and memory requirements of our proposed algorithms as being linear in K and independent of n. This allows us to run PhaseCode using parameters as high as $n = 10^{10}$ and $K = 10^4$ on a regular laptop. See Figure 3.

II. Problem Formulation

Consider a complex signal $x \in \mathbb{C}^n$ of length n which is K-sparse; that is, only K out of n components of vector x are non-zero. Let $A \in \mathbb{C}^{m\times n}$ be the measurement matrix that needs to be designed. The phase retrieval problem is to recover the signal x from magnitude measurements $y_i = |a_i x|$, where a_i is the i-th row of matrix A.

The main objectives of the compressive phase retrieval problem are to design matrix A, and the decoding algorithm to recover x, that satisfy the following objectives.

- The number of measurements m is as small as possible. Ideally, one wants m to be close to the fundamental limit of $4K - \mathcal{O}(1)$ [17].
- The decoding algorithm is fast with low computational complexity and memory requirements. Ideally, one wants the time complexity and the memory complexity of the algorithm to be $\mathcal{O}(K)$, which is optimal.
- The reliability of the recovery algorithm should be maximized. Ideally, one wants the probability of failure to be vanishing as the problem parameters K and m get large.[2]

III. The PhaseCode Algorithm

Suppose that $x \in \mathbb{C}^n$ is K-sparse. First we define $A \in \mathbb{C}^{4M\times n}$ to be a "row tensor product"[3] of matrices T and H, where $H \in \{0,1\}^{M\times n}$ is a binary "code" matrix, to be shortly explained, and $T \in 4 \times n$ is the "trigonometric modulation" matrix that provides 4 measurements per each row of H. We define a row tensor product of matrices T and H, $T \otimes H$, as follows. Let $A = T \otimes H = [A_1^T, A_2^T, \ldots, A_M^T]^T$ and $A_i \in \mathbb{C}^{4\times n}$. Then, $A_i(jk) = T_{jk}H_{ik},\ 1 \le j \le 4,\ 1 \le k \le n$.

Example Consider matrices

$$T = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \end{bmatrix} \text{ and } H = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

Then, our measurement matrix A is design from:

$$A = T \otimes H = \begin{bmatrix} 0 & 0.2 & 0.3 \\ 0 & 0.5 & 0.6 \\ 0.1 & 0.2 & 0 \\ 0.4 & 0.5 & 0 \\ 0.1 & 0 & 0.3 \\ 0.4 & 0 & 0.6 \end{bmatrix}.$$

Matrix H is constructed using a carefully chosen "balls-and-bins" model, where as mentioned, the balls refer to the non-zero values of x, and the bins refer to the measurements. Thus, each column of H denotes a ball and each row of H denote a bin. Matrix H is designed to ensure that each ball goes to exactly d bins uniformly at random, where d is a design parameter. For instance, in the above toy example $d = 2$. Thus, $H_{ij} = 1$ if and only if ball j is in bin i, and $H_{ij} = 0$ otherwise. Formally, we construct the ensemble of d-left regular degree bipartite graphs $\mathcal{C}^n(d, M)$, using a balls-and-bins model as follows. We construct a bipartite graph of n left nodes and M right nodes. When a ball goes to a bin, we construct an undirected edge between the corresponding left and right nodes in the bipartite graph.

While we provide the details of how to design matrix T in Section V, for completeness of the description, here we state the precise expression for T without further explanation. Let $\omega' = \frac{2\pi L}{n}$ be a random phase between 0 and 2π, i.e. the discrete random variable L is uniformly distributed between 0 and $n-1$. We design $T \in \mathbb{C}^{4\times n}$ to be

$$T = \begin{pmatrix} e^{\mathbf{i}\omega} & e^{\mathbf{i}2\omega} & \ldots & e^{\mathbf{i}n\omega} \\ e^{-\mathbf{i}\omega} & e^{-\mathbf{i}2\omega} & \ldots & e^{-\mathbf{i}n\omega} \\ \cos(\omega) & \cos(2\omega) & \ldots & \cos(n\omega) \\ e^{\mathbf{i}\omega'} & e^{\mathbf{i}2\omega'} & \ldots & e^{\mathbf{i}n\omega'} \end{pmatrix}. \quad (1)$$

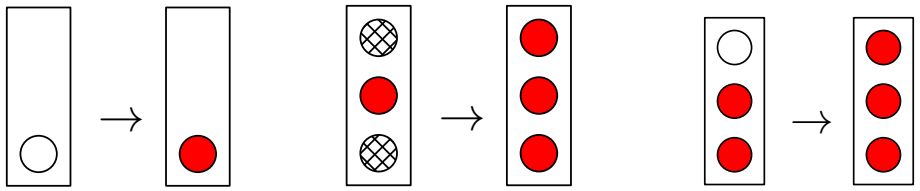
Let $\tilde{x} \in \mathbb{C}^K$ be the shortened vector constructed from the K non-zero components of x in the way that the order of these components' indices are maintained. Let $\tilde{H} \in \mathbb{C}^{M\times K}$ be the corresponding code matrix that is constructed from the active columns of H in the trivial way. Let $\mathcal{C}_1^K(d, M)$ be the ensemble of bipartite graphs induced by $\tilde{x}$. Note that the induced graph also has a d-left regular degree, and when K is large and M/K is a constant, the weight of each row of matrix H or the right-node degree approaches a Poisson random variable with parameter $\lambda = \frac{Kd}{M}$.

As in [3], in the balls-and-bins model, we use the following terminology extensively throughout the paper:

- *Singleton:* A bin is a singleton if it contains one ball.
- *Doubleton:* A bin is a doubleton if it contains exactly

[2] In this work, we are interested in the asymptotic K regime. However, even when K is small, with proper modification of our algorithm, high reliability can be guaranteed when m gets large.

[3] Here, we do not follow popular convention for the notation for tensor product of matrices; instead, we define our own notation that is convenient for our purpose, and which should hopefully not cause any confusion.



(a) Coloring a ball in a singleton bin

(b) Combining colors in a multiton bin

(c) Coloring a ball in a multiton bin

Fig. 1: This figure is showing the basic 3 merge-and-color primitives of our algorithm.

two balls.

- *Multiton:* A bin is a multiton if it contains more than one ball.[4]

We propose two decoding algorithms called Unicolor PhaseCode and Multicolor PhaseCode. We first describe Unicolor PhaseCode, and analyze it in Section IV. Next, we describe Multicolor PhaseCode that is more efficient, but whose analysis remains unresolved, and is part of our ongoing work. Our decoding algorithms are based on the coloring of balls; and merging of different colors. With the aid of the carefully designed matrix T, our decoder is capable of performing the following functions:

- When a ball is in a singleton bin, the ball is colored with a new color. Figure 1a illustrates this operation.
- When *all* the balls in a multiton bin are colored, and the number of colors in that multiton bin is exactly two, then those two colors can be combined into a single composite color. Figure 1b illustrates this operation.
- When a multiton bin consists of exactly one *uncolored ball*, and the other non-empty set of balls in the bin have all the same color (let's say red), then the uncolored ball is colored with that color (i.e. it becomes red). Figure 1c illustrates this operation.

Remark Note that matrix T should enable the decoder to first detect the singleton bins, multiton bins with two colors, and multiton bins with a single uncolored ball and 1 color, and then perform the above functions. In this section, we do not provide the details of how these operations can be done. Instead, we focus on a coloring problem on a balls-and-bins model. The rules of the coloring problem are as mentioned above, and the goal is to design matrix $\tilde{H}$ such that almost all the balls get colored with the same color, when the coloring procedure ends.

Unicolor Algorithm In the first iteration of the algorithm all the singletons are colored. In the second iteration, all the doubletons that each contain two colored balls from the first iteration, are detected, and their colors are combined. Then, the *largest* set of balls having the same color[5] is selected, and *every other ball gets uncolored*. At this point, there is only *one* color, and no new colors are added to the system. Hence, we use the terminology *Unicolor* for this algorithm. In the following iterations, if there is only one uncolored ball in a bin, with one or more colored balls, then that uncolored ball gets colored. (See Figure 1c.) The algorithm continues until no more balls can be colored.

Multicolor Algorithm In the first iteration of the algorithm, all the singletons are colored. In the following iterations, the decoder checks all the non-singleton bins. If they consist of only 2 colors, those colors are combined. (See Figure 1b.) If the colored balls in the bin all have the same color, and if there is only one uncolored ball in that bin, then that ball gets colored. (See Figure 1c.) The algorithm continues until no more balls can be colored, or no more colors can be merged.

We provide pseudocode of both algorithms in [21].

Remark Both algorithms have $\mathcal{O}(K)$ sample and decoding complexity, with Multicolor PhaseCode having a better constant than Unicolor PhaseCode in sample complexity due to its greater color-combining power.

Note that in both algorithms, the recovered balls are the *largest* set of balls having the same color. Intuitively, it is clear that the Unicolor PhaseCode is less expressive, since does not exploit the ability to combine colors after the second iteration. The following example illustrates the two algorithms and illustrates why Unicolor PhaseCode is suboptimal compared to Multicolor PhaseCode.

Example Let $K = 4$, $M = 5$ and $d = 2$. Label the balls by 1 to 4. Suppose that the induced bipartite graph is such that the bins are $\{1\}$, $\{1, 2\}$, $\{3\}$, $\{3, 4\}$, and $\{2, 3, 4\}$. In the first iteration, both algorithms color balls 1 and 3, let us say by red and blue, respectively since these balls are in singletons. In the second iteration, Multicolor PhaseCode can color ball 2 as red using $\{1, 2\}$, and ball 4 as blue using $\{3, 4\}$. However, the Unicolor PhaseCode does not find any doubleton containing balls 1 and 3. Thus, Unicolor PhaseCode has to pick either ball 1 and ball 3 randomly (let's say ball 1) as the largest set with one color. Finally, since bin 5 consists of a red ball and two blue balls, Multicolor PhaseCode has the ability to merge colors red and blue. This

[4] In our terminology, a doubleton bin is also a multiton bin.

[5] Whenever two balls having colors C_1 and C_2 are combined, they get the same composite color C_{12}, and all other balls with colors C_1 and C_2 are also recolored to C_{12}.

m	$12.44K$	$12.72K$	$13.28K$	$\mathbf{13.92K}$	$14.64K$	$15.4K$
$p^*(m)$	1.1×10^{-3}	8×10^{-5}	3.2×10^{-6}	$\mathbf{1 \times 10^{-7}}$	2.9×10^{-9}	7×10^{-11}
d	5	6	7	**8**	9	10

TABLE I: Family of trade-offs between error floor and number of measurements for Unicolor Phasecode. The table shows that to achieve higher reliability, i.e. smaller error floor, the number of measurements m should be increased.

completes the successful decoding of Multicolor PhaseCode, since all the balls are colored and they have the same color. However, Unicolor PhaseCode can only color ball 2 and add it to the largest set. Thus, it recovers only 2 out of 4 balls.

The main theoretical contribution of this paper is the following theorem.

Theorem 1. *Let $A = T \otimes H$ be the measurement matrix, where H is chosen uniformly at random from the ensemble $\mathcal{C}^n(d, M)$ and T is the "modulation" matrix defined in* (1)*. Using the m measurements $y = |Ax|$, Unicolor PhaseCode is able to recover a fraction $1-p^*(m)$ of nonzero components of x with probability $1-\mathcal{O}(1/K)$, where m and $p^*(m)$ form a family of trade-offs as shown in Table I for selected operating points. In particular, Unicolor PhaseCode is able to recover a fraction $1-10^{-7}$ of the non-zero components of x with $14K$ measurements with high probability. Furthermore, the decoding complexity of the algorithm is $\mathcal{O}(K)$ which is order-optimal.*

Proof. See Section IV. ■

The achievable trade-off between reliability and measurement cost as specified by Theorem 1 is shown in the Table I.

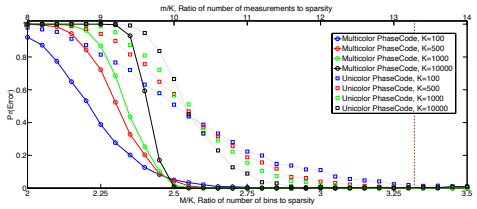


Fig. 2: **Performance of PhaseCode Algorithms**. Unicolor PhaseCode successfully recovers almost all balls when $m = 13.28K$. Multicolor PhaseCode achieves the same level of reliability with $m \simeq 11K$, that is about 17% reduction in number of measurements.

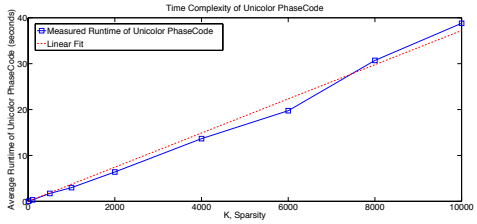


Fig. 3: **Time Complexity of Unicolor PhaseCode**. We measured runtime of Unicolor PhaseCode algorithm. We chose $n = 10^{10}$ and varied K in order to see linear time complexity of the algorithm. Because both computational complexity and memory complexity depend only on K not n, we can easily simulate arbitrarily large n such as 10^{10}.

Before we move to the proof of the main theorem, we first exhibit the simulated performance of Unicolor PhaseCode and Multicolor PhaseCode in Figure 2. Theorem 1 guarantees that Unicolor PhaseCode recovers a fraction $p^*(m)$ of x with m measurements with high probability, where $(m, p^*(m))$ can be chosen from Table I. We chose the 3rd column of the table as an operating point, i.e., $(m, p^*(m)) = (13.28K, 3.2 \times 10^{-6})$. [6] Thus, we expect that the Unicolor PhaseCode algorithm will recover a fraction $1-3.2\times10^{-6}$ of K active symbols with high probability when $m = 13.28K$. Using simulations, we measured the error probability of both Unicolor PhaseCode and Multicolor PhaseCode while m is varied between $8K$ and $14K$; we ran each point 1000 times and determined error probability. Note that the error probability is defined as probability of not recovering a fraction $p^*(m)$ or more of nonzero components of x. We repeated the same set of simulations for several values for K.

As claimed in the theorem, Unicolor PhaseCode algorithm indeed successfully recovers the signal near perfectly with very high probability when $m = 13.28K$. It is also observed that the error probability for larger K is lower than that of smaller K. The simulation results not only support the main theorem but also show the superior performance of Multicolor PhaseCode algorithm over that of Unicolor PhaseCode algorithm: Multicolor PhaseCode is observed to achieve the same level of reliability with $m \simeq 11K$, that is about 17% reduction in number of measurements.

Theorem 1 states also that the decoding complexity of the PhaseCode algorithms is $\mathcal{O}(K)$, which is optimal. Additionally, its memory complexity is $\mathcal{O}(K)$, which is also optimal. In order to corroborate these claims, we measured the running time of Unicolor PhaseCode Algorithm. We chose the same operating point as in the above simulations: $(m, p^*(m), d) = (13.28K, 3.2 \times 10^{-6}, 7)$. We randomly generated signals of length $n = 10^{10}$, and varied the sparsity K up to 10^4 to see the average runtime scaling behavior of the algorithm. The results are plotted in Figure 3. As one can see, the measured decoding time increases linearly with K. The exact runtime can be much improved considering that the simulator is written in Python and not fully optimized, and that we measured the runtime on a normal laptop.[7].

[6] It will be explained in the following section how one can choose an operating point. For these simulations, we set the left degree as 7, i.e., $d = 7$.

[7] For the measurements, we used a laptop with 2GHz Intel Core i7 and 8GB memory.

IV. Analysis of Unicolor PhaseCode

In this section, we analyze the performance of Unicolor PhaseCode using *density evolution* techniques [2], [19] that are used to analyze the performance of message passing algorithms on sparse-graph codes. Our arguments are similar to that in [2]. We find a recursion relating the probability that a randomly chosen ball (or left node) in the graph is *not* colored after j iterations of the algorithm, p_j, to the same probability after $j+1$ iterations of the algorithm, p_{j+1}. *It is important to recognize that our density evolution equation is however, different from that for phase-aware systems (such as LDPC graphs) having similar graph ensemble, and therefore requires a different analysis.* Furthermore, our graph ensemble $\mathcal{C}_1^K(d, M)$ is different from the one in [2] as left nodes have regular degree, while right nodes have irregular degree in our ensemble. (In [2], both left and right nodes have regular degree.)

Let $\rho_i,\ i \geq 1$ be the probability that a randomly selected edge has degree i on the right node. Since ρ_i is the fraction of edges that are connected to a right node of degree i, we have

$$\rho_i = \frac{iM}{Kd}\frac{\lambda^i e^{-\lambda}}{i!} = \frac{\lambda^{i-1}e^{-\lambda}}{(i-1)!}.$$

Define $\rho(t) = \sum_{i=1}^{\infty} \rho_i t^{i-1}$ as the polynomial representing the edge degree distribution of right nodes. Then,

$$\rho(t) = \sum_{i\geq 1} \frac{\lambda^{i-1}e^{-\lambda}}{(i-1)!} t^{i-1} = e^{-\lambda(1-t)}. \tag{2}$$

Let $\mu_i,\ i \geq 1$ be the probability that a randomly selected edge has degree i on the left node. Clearly, $\mu_i = 1_{\{i=d\}}$, where 1_E is the indicator that event E has happened, i.e. $1_E = 1$ if E is true, and $1_E = 0$, otherwise. Define $\mu(t) = \sum_{i=1}^{\infty} \mu_i t^{i-1} = t^{d-1}$ as the polynomial representing the edge degree distribution of left nodes.

At each iteration of Unicolor PhaseCode, we call the giant component as the largest set of balls that have the same color. The algorithm follows 3 major steps to recover almost all the balls by coloring them.

- *Step* 1*:* All the singleton bins and their corresponding balls are found.
- *Step* 2*:* A giant component of colored balls is formed as follows. First, doubleton bins having both balls in singleton bins in step 1 are found, and the corresponding two balls are "linked". Then, the largest set of balls that are linked is considered as the giant component.
- *Step* 3*:* After the giant component is formed, at each iteration of the algorithm, balls are colored at a time, and connected to the giant component.

Let p_j be the probability that a randomly chosen ball does not belong to the giant component at step j. The density evolution equation is an equation relating p_j to p_{j+1}. Under the tree-like assumption, and for $j \geq 2$ one has

$$p_{j+1} = (1 + e^{-\lambda} - e^{-\lambda p_j})^{d-1}. \tag{3}$$

Here is a proof of Equation (3). A ball v passes a message to bin c that it is not part of a giant component at step $j+1$, if none of the other $d-1$ neighbor bins of v can tell v that it is part of the giant component at step j. First note that if a bin is a singleton, it cannot tell the ball that it is part of a giant component. This is a fundamental difference of our decoding process compared to that of conventional peeling-based decoders such as the LDPC decoder for erasure channel [19]. In LDPC decoding, since there is no phase ambiguity, as soon as a singleton bin is detected, the ball in the singleton bin is recovered and it is peeled from all other bins that also contain that ball. However, singleton balls cannot be peeled from other bins in our setting. Indeed, our problem has the peculiar attribute that singleton bins, while critical to initiating the growth of the giant component at the outset, are not useful once a giant component is formed, and too many singletons actually hurt the system performance by featuring useless isolated measurements. This is a significant departure from "phase-aware" measurement systems like LDPC codes.

More precisely, a bin can tell a ball that it is not part of the giant component if the bin is connected to a non-empty set of balls other than v, and they are all in the giant component. This happens with probability

$$\sum_{i=2}^{\infty} \rho_i (1-p_j)^{i-1} = e^{-\lambda p_j} - e^{-\lambda}.$$

Thus, the probability that ball v passes a message to bin c

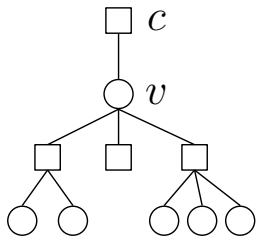


Fig. 4: Length-2 tree-like neighborhood of (v, c) for $d = 4$.

that it is not part of the giant component is the probability that none of the other $d-1$ bins can tell v that it is in the giant component. These messages are all independent if the bipartite graph is a tree. Assuming this, one has

$$p_{j+1} = (1 - (e^{-\lambda p_j} - e^{-\lambda}))^{d-1}.$$

Note that in Multicolor PhaseCode, another possibility of joining the giant component is that v is colored, let's say as red, with the color of the giant component being another color, say blue, and a neighbor bin of v contains only blue and red balls. This will boost system performance by accelerating the coloring and merging process, but is hard to analyze precisely. Therefore, for analytical purposes only, we do not allow this opportunity to be exploited by Unicolor PhaseCode.

An interesting but unfortunate fact is that $p_0 = 1$ is a fixed point of the density evolution equation. Thus, one cannot use (3) at the outset to follow the evolution of p_j, and to

argue that it goes close to 0, since p_j can get stuck at 1. To use Equation (3), we need a more careful characterization of the first two steps of the algorithm that form the giant component. At the first iteration, all the balls in singleton bins are found. Therefore, no giant component is formed yet; thus, $p_1 = 1$. At the second iteration, the giant component is formed by coloring the balls in doubleton bins having both balls in singleton bins found in step 1. After the giant component is formed in the second iteration, the probability that a randomly chosen ball is part of the giant component is p_2. If one can show that p_2 is small enough such that after a fixed number of iterations p_j gets close to 0, then concentration bounds can be used to show that the number of balls not being in the giant component is indeed highly concentrated around its mean after ℓ iterations, Kp_ℓ. In Lemma 2, we show that *if $p_2 = 1 - \delta$ for some arbitrary constant $0 < \delta < 1$ independent of K, p_j gets close to 0 after a constant number of iterations.* Clearly $p_2 = 1 - \delta$ if there exists a giant component of size linear in K after the second step.

Towards this end, in Lemma 1, we form a graph with nodes that are balls which are in singletons. We consider edges between these balls if they are in a doubleton, and we use an Erdos-Renyi random graph model [20] to find parameters d and M for which there is a giant component of size linear in K in the initial phase of the algorithm. See [21] for details.

Lemma 1. *If $d = 5$ and $3.11K \leq M \leq 19.24K$, with probability $1 - \mathcal{O}(1/K)$, there exists a giant component of size linear in K formed by the balls in singletons. For $d = 8$, there is a giant component with high probability if $3.48K \leq M \leq 55.36K$.*

Proof. See [21]. ■

Recall that p_j is the probability that a randomly chosen ball is not part of the giant component at iteration j. By Lemma 1, for proper choices of d and M, there exists a linear-size giant component, let's say of size $K_s = \delta K$ for some constant $0 < \delta < 1$, at step 2. Therefore, $p_2 = \frac{K - K_s}{K} = 1 - \delta$. The following corollary is an immediate result of Lemma 1.

Corollary 1. *There exists a constant $0 < \delta < 1$ independent of K, such that $p_2 = 1 - \delta$.*

Due to the formation of a linear-size giant component in step 2 of the algorithm, we can revisit the density evolution equation (3):

$$p_{j+1} = \left(1 + e^{-\lambda} - e^{-\lambda p_j}\right)^{d-1},$$

with the aid of Corollary 1, which guarantees that p_2 is strictly smaller than 1. Recall that $p_0 = 1$ is a fixed point of (3). But with the giant component formation, we can break away from the shackles of "being stuck" at $p_0 = 1$. With $p_2 < 1$, we hope to find a better fixed point of (3) to which our density evolution will converge.

Towards this end, ideally one wants Equation (1) to have the property

$$p_{j+1} = (1 + e^{-\lambda} - e^{-\lambda p_j})^{d-1} < p_j, \tag{4}$$

for all $p_j \in (0, 1)$. Let's take a closer look at the fixed point equation

$$t = f(t) = (1 + e^{-\lambda} - e^{-\lambda t})^{d-1}. \tag{5}$$

As mentioned, one solution is $t_1^* = 1$. As we can break away from t_1^*, fortunately there exists another solution approximately at $t_2^* \simeq e^{-\lambda(d-1)}$ which is close to 0. From now on, we will refer to this fixed point as the error floor p^*.

Lemma 2. *Let $d = 5$. If $2.33K \leq M \leq 13.98K$, then the fixed point equation (5) has exactly 2 solutions for $t \in [0, 1]$: $t_1^* = 1$ and $t_2^* \simeq e^{-\lambda(d-1)}$ (See Figure 5). For $d = 8$, a similar result holds if $2.63K \leq M \leq 47.05K$.*

Proof. See [21]. ■

With the aid of Lemma 2, we can show that p_j gets very close to fixed point $p^* = t_2^*$ after a constant number of iterations. This is established in the following corollary.

Corollary 2. *For any $\epsilon > 0$, there exists a constant $\ell(\epsilon)$ such that $p_\ell \leq p^* + \epsilon$.*

Table II illustrates how the error floor p^* and the minimum ratio of bins to balls $c = M/K$ change for different values of d. If our reliability target allows the error floor to be set at 1.1×10^{-3}, then $d = 5$ minimizes the number of required bins. Recall that the total number of measurements is $m = 4M$ which matches the result of Table I. (See Section III) If one wants to achieve smaller error floor, then d and c should be both increased.

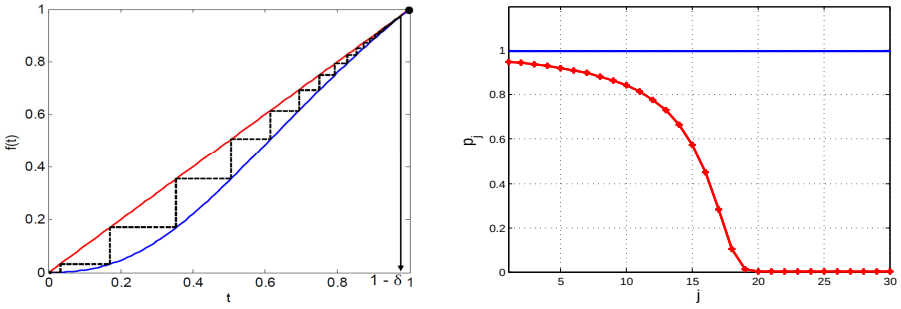
In the density evolution analysis so far, we have shown that the *average* fraction of balls that cannot be recovered will be arbitrarily close to the error floor after a fixed number of iterations, provided that the tree-like assumption is valid. It remains to show that the actual fraction of balls that are not in the giant component after ℓ iterations is highly concentrated around p_ℓ. Towards this end, first in Lemma 3 we show that a neighborhood of depth ℓ of a typical edge is a tree with high probability for a constant ℓ. Second, in Lemma 4, we use the standard Doob's martingale argument [2], to show that the number of uncolored balls after ℓ iterations of the algorithm is highly concentrated around Kp_ℓ.

Consider a directed edge from $\vec{e} = (v, c)$ from a left-node (ball) v to a right-node (bin) c. Define the directed neighborhood of depth ℓ of $(\vec{e})$ as $\mathcal{N}_{\vec{e}}^{\ell}$, that is the subgraph of all the edges and nodes on paths having length less than or equal to ℓ, that start from v and the first edge of the path is not $\vec{e}$.

Lemma 3. *For a fixed ℓ^*, $\mathcal{N}_{\vec{e}}^{2\ell^*}$ is a tree-like neighborhood with probability at least $1 - \mathcal{O}(\log(K)^{\ell^*}/K)$.*

Lemma 4. *Over the probability space of the ensemble of graphs $\mathcal{C}_1^K(d, M)$, let Z be the number of uncolored edges*[8]

[8]An edge is colored if its corresponding ball is colored.



(a) The density evolution curve for parameters $d = 5$ and $\lambda = 2$.

(b) The evolution of p_j after each iteration for $d = 5$ and $\lambda = 2$.

Fig. 5: Figure (a) illustrates the density evolution equation: $p_{j+1} = f(p_j)$. In order to track the evolution of p_j, pictorially, one draws a vertical line from (p_j, p_j) to $(p_j, f(p_j))$, and then a horizontal line between $(p_j, f(p_j))$ and $(f(p_j), f(p_j))$. Since the two curves meet at $(1, 1)$ if $p_0 = 1$, then p_j gets stuck at 1. However, if $p_0 = 1 - \delta$, p_j decreases after each iteration, and it gets very close to 0. Figure (b) illustrates the same phenomenon by showing the evolution of p_j versus the iteration, j. Note that in this example, p_j gets very close to 0 after only 20 iterations.

d	4	**5**	6	7	8	9	10
p^*	2.7×10^{-2}	1.1×10^{-3}	8×10^{-5}	3.2×10^{-6}	1×10^{-7}	2.9×10^{-9}	7×10^{-11}
c	3.31	**3.11**	3.18	3.32	3.48	3.66	3.85

TABLE II: The table shows how error floor, p^*, and $c = M/K$ (which indirectly determines the number of measurements) vary for different values of left degree, d.

after ℓ iterations of the Unicolor PhaseCode algorithm. Then, for any $\epsilon > 0$, there exists a large enough K and constants β and γ such that

$$|\mathbb{E}[Z] - Kdp_\ell| < Kd\epsilon/2 \tag{6}$$

$$\mathbb{P}(|Z - Kdp_\ell| > Kd\epsilon) < 2e^{-\beta\epsilon^2 K^{1/(4\ell+1)}}, \tag{7}$$

where p_ℓ is derived from the density evolution equation (3).

The proofs of the above two lemmas are provided in the appendix of [21].

Now gathering the results of Corollary 2 and Lemmas 1 and 4 completes the proof of Theorem 1. Note that the dominant probability of error is due to the event that the giant component is not formed in the second iteration which happens with probability $\mathcal{O}(1/K)$.

V. Measurement Design: "Trig-Modulation"

In this section, we will explain the choice of the measurement matrix T. Our design of T draws heavily from the proposed trigonometric subsystem in [1] with proper modifications to better match our sparse-graph code subsystem, H, that is distinct from [1]. We also show that one can decrease the number of these trig-based measurements from 5 per bin as proposed in [1] to 4 per bin as we describe.

Define the length 4 vector y_i to be the measurement vector corresponding to the i-th row of matrix H for $1 \le i \le M$. Then $y = [y_1^T, y_2^T, \ldots, y_M^T]^T$, where $y_i = [y_{i,1}, y_{i,2}, y_{i,3}, y_{i,4}]^T$. Let $\omega = \frac{\pi}{2n}$. We design the measurement matrix $T = [t_{j\ell}]$ as follows. For all ℓ, $1 \le \ell \le n$,

$$t_{1\ell} = e^{\mathbf{i}\omega\ell}, \qquad t_{2\ell} = e^{-\mathbf{i}\omega\ell},$$
$$t_{3\ell} = 2\cos(\omega\ell), \quad t_{4\ell} = e^{\mathbf{i}\omega'\ell},$$

where as mentioned in Section III, $\omega' = \frac{2\pi L}{n}$ and L is uniformly distributed between 0 and $n-1$.

As mentioned in Section III, the measurement matrix should enable us to detect whether we have a singleton bin, and if yes, the location index of the corresponding ball in the singleton bin. Furthermore, it should detect if a multiton bin consists of only known balls having exactly two unique colors. We call these as mergeable multitons (See Figure 1b). Finally, if a multiton bin consists of known balls with the same color and only one uncolored ball, the measurement should be able to find the index of uncolored ball. We call these as resolvable multitons as in [1] (See Figure 1c). In the following, we show how each of these detections can be accomplished using "guess and check" approach.

(i) Singletons: Suppose that we want to check the hypothesis that the i-th bin is a singleton. If the bin is a singleton, only one non-zero component of x, let's say x_ℓ, is involved in vector y_i, that is $y_{i,1} = |x_\ell e^{\mathbf{i}\omega\ell}|$, $y_{i,2} = |x_\ell e^{-\mathbf{i}\omega\ell}|$, and so on. Thus, the i-th bin is a singleton only if $y_{i,1} = y_{i,2} = y_{i,4}$. The event that bin i is not a singleton, and all these measurements are equal has measure 0 for our generic choice of signal components. In order to find the index ℓ, one uses $y_{i,3}$ to get

$$\ell = \frac{1}{\omega}\cos^{-1}(\cos(\omega\ell)) = \frac{1}{\omega}\cos^{-1}(\frac{y_{i,3}}{2y_{i,1}}).$$

Note that $\cos(\omega\ell)$ is positive if $0 \le \omega \le \frac{\pi}{2n}$ for all ℓ, $1 \le \ell \le n$.

(ii) Mergeable multitons: Consider a bin i as in Figure 1b, in which we already know that there are some red balls (non-empty set $\mathcal{R}$) and some blue balls (non-

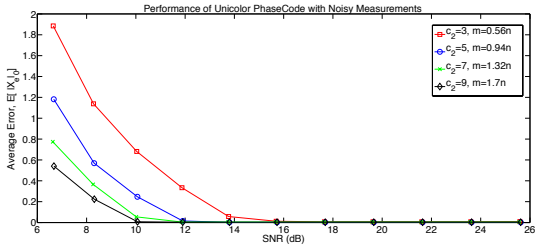


Fig. 6: **Performance of PhaseCode with Noisy Measurements.** We evaluate the robustified Unicolor PhaseCode algorithm with noisy measurements, i.e., $y = |Ax| + w$, where w is white Gaussian noise. Signal-to-noise ratio, SNR, is defined as the power ratio between a measured signal and the added noise signal. We measured the L_0 norm of estimation errors. We used a random signal of length $n = 2048$ and sparsity $K = 5$. Each point is averaged over 500 runs.

empty set $\mathcal{B}$). This means that the red balls are known in magnitude and phase relative to each other, and similarly the blue balls are known relative to each other. However, the relative phase of blue balls and red balls are not known. We can show that if there is no other ball in the bin, the relative phase can be found. See [21] for details.

(iii) Resolvable multitons: Consider a bin, let's say bin i, in which we know that there are some known balls that have the same color. We want to check if bin i has exactly one other uncolored ball; i.e. one unknown non-zero component of x, say x_ℓ, as in Figure 1c. In [21], we show how one can find ℓ and x_ℓ. See [21] for details.

VI. Conclusion and Future Work

We considered the problem of recovering a complex K-sparse signal $x \in \mathbb{C}^n$ from m intensity measurements of the form $|a_i x|$, $1 \leq i \leq m$, where a_i is a measurement row vector. We proposed Unicolor PhaseCode and Multicolor PhaseCode algorithms that are based on a sparse-graph coding framework. We showed that Unicolor PhaseCode can provably recover all but a tiny fraction of the non-zero signal components with high probability. To the best of our knowledge, our work is the first one that characterizes the precise number of measurements needed to guarantee high reliability, rather than only Big Oh statements. Our proposed algorithms have both an order-optimal $O(K)$ decoding time and an order-optimal $O(K)$ memory complexity. In [21], motivated by some important practical classes of optical systems, we also consider a "Fourier-friendly" constrained measurement setting, and show that its performance matches that of the unconstrained setting described here. We also study the general non-sparse signal case, for which we propose a simple deterministic set of $3n - 2$ measurements that can recover the n-length signal under some mild assumptions. We refer the readers to [21] for more details on these results.

In practical systems, the magnitude measurements are noisy due to various sources of noise. Our proposed PhaseCode algorithms can be robustified, while retaining the basic sparse-graph coding architecture of the underlying noiseless PhaseCode algorithm. Without providing much details about the actual measurement designs in this paper, we provide simulation results with noisy measurements in Figure 6, which shows how well our robust PhaseCode algorithm performs in the presence of noise. Rigorous analysis of the noisy case is an important future work regarding compressive phase retrieval problem. We refer the readers to [21] for more discussions on interesting future directions.

References

[1] S. Cai, M. Bakshi, S. Jaggi, and M. Chen, "SUPER: Sparse signals with Unknown Phases Efficiently Recovered," *arXiv preprint arXiv:1401.4451*, 2014.

[2] T. Richardson and R. Urbanke, "The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding" *IEEE Transactions on Information Theory*, vol. 47, pp. 599–618, Feb. 2001.

[3] S. Pawar and K. Ramchandran, "Computing a k-sparse n-length Discrete Fourier Transform using at most $4k$ samples and $O(k \log k)$ complexity," *arXiv preprint arXiv:1305.0870*, 2013.
Proc. 10th Intern. Conf. on Sampling Theory and Applications (SampTA), July 2013.

[4] R. P. Milane, "Phase retrieval in crystallography and optics," *J. Opt. Soc. Am. A*, vol. 7, pp. 394–411, 1990.

[5] J. C. Dainty and J. R. Fienup, "Phase retrieval and image reconstruction for astronomy," *Image Recovery: Theory and Application, Academic Press*, pp. 231–275, 1987.

[6] M. L. Moravec, J. K. Romberg, and R. Baraniuk, "Compressive phase retrieval," in *SPIE Conf. Series*, vol. 6701, 2007.

[7] E. J. Candes, T. Strohmer, and V. Voroninski, "Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming," *Communications on Pure and Applied Mathematics*, vol. 66, no. 8, pp. 1241–1274, 2013.

[8] P. Schniter and S. Rangan, "Compressive phase retrieval via generalized approximate message passing," *Proceedings of Allerton Conference on Communication, Control, and Computing*, 2012.

[9] J. M. Rodenburg, "Ptychography and related diffractive imaging methods," *Advances in Imaging and Electron Physics*, vol. 150, pp. 87–184, 2008.

[10] K. Jaganathan, S. Oymak, and B. Hassibi, "Sparse phase retrieval: Convex algorithms and limitations," *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, pp. 1022–1026, 2013.

[11] K. Jaganathan, S. Oymak, and B. Hassibi, "Phase retrieval for sparse signals using rank minimization," *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3449–3452, 2012.

[12] H. Ohlsson, A. Yang, R. Dong, and S. Sastry, "Compressive phase retrieval from squared output measurements via semidefinite programming," *arXiv preprint, arXiv:1111.6323*, 2011.

[13] D. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 4, 2006.

[14] X. Li and V. Voroninski, "Sparse signal recovery from quadratic measurements via convex programming," *arXiv preprint arXiv:1209.4785*, 2012.

[15] P. Netrapalli, P. Jain, and S. Sanghavi, "Phase retrieval using alternating minimization," *arXiv preprints arXiv:1306.0160*, 2013.

[16] A. Walther, "The question of phase retrieval in optics," *Optica Acta*, vol. 10, no. 1, pp. 41–49, 1963.

[17] M. Akcakaya and V. Tarokh, "New conditions for sparse phase retrieval," *arXiv preprint arXiv:1310.1351*, 2013.

[18] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, maxcut and complex semidenite programming," *Mathematical Programming*, pp. 1–35, 2013.

[19] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman, "Improved low-density parity check codes using irregular graphs," *IEEE Trans. Info. Theory*, vol. 47, pp. 585–598, 2001.

[20] Erdos, P. and Renyi, A. "On the evolution of random graphs," *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, pp. 17–61, 1960.
Processing," *Prentice Hall*, 1989.

[21] R. Pedarsani, K. Lee, and K. Ramchandran, "PhaseCode: Fast and Efficient Compressive Phase Retrieval based on Sparse-Graph-Codes," *arXiv preprint arXiv:1408.0034*, 2014.