A Compressed Sensing Approach to Pooled RT-PCR Testing for COVID-19 Detection

SABYASACHI GHOSH [1], **RISHI AGARWAL**[1], **MOHAMMAD ALI REHAN**[1], **SHREYA PATHAK**[1],
PRATYUSH AGARWAL[1], **YASH GUPTA**[1], **SARTHAK CONSUL**[2], **NIMAY GUPTA**[1], **RITIKA**[1], **RITESH GOENKA**[1],
AJIT RAJWADE [1], **AND MANOJ GOPALKRISHNAN**[2]

[1]Department of Computer Science and Engineering, IIT Bombay, Mumbai 400076, India
[2]Department of Electrical Engineering, IIT Bombay, Mumbai 400076, India

CORRESPONDING AUTHOR: AJIT RAJWADE (e-mail: ajitvr@cse.iitb.ac.in).

ABSTRACT We propose 'Tapestry', a single-round pooled testing method with application to COVID-19 testing using quantitative Reverse Transcription Polymerase Chain Reaction (RT-PCR) that can result in shorter testing time and conservation of reagents and testing kits, at clinically acceptable false positive or false negative rates. Tapestry combines ideas from compressed sensing and combinatorial group testing to create a new kind of algorithm that is very effective in deconvoluting pooled tests. Unlike Boolean group testing algorithms, the input is a quantitative readout from each test and the output is a list of viral loads for each sample relative to the pool with the highest viral load. For guaranteed recovery of k infected samples out of $n \gg k$ being tested, Tapestry needs only $O(k \log n)$ tests with high probability, using random binary pooling matrices. However, we propose deterministic binary pooling matrices based on combinatorial design ideas of Kirkman Triple Systems, which balance between good reconstruction properties and matrix sparsity for ease of pooling while requiring fewer tests in practice. This enables large savings using Tapestry at low prevalence rates while maintaining viability at prevalence rates as high as 9.5%. Empirically we find that single-round Tapestry pooling improves over two-round Dorfman pooling by almost a factor of 2 in the number of tests required. We evaluate Tapestry in simulations with synthetic data obtained using a novel noise model for RT-PCR, and validate it in wet lab experiments with oligomers in quantitative RT-PCR assays. Lastly, we describe use-case scenarios for deployment.

INDEX TERMS Compressed sensing, coronavirus, COVID-19, group testing, Kirkman/Steiner triples, mutual coherence, pooled testing, sensing matrix design.

I. INTRODUCTION

The coronavirus disease of 2019 (COVID-19) crisis has led to widespread lockdowns in several countries, and has had a major negative impact on the economy. Early identification of infected individuals can enable quarantining of the individuals and thus control the spread of the disease. Such individuals may often be asymptomatic for many days. Widespread testing with the RT-PCR (reverse transcription polymerase chain reaction) method can help identify the infected individuals. However, widespread testing is not an available option in many countries due to constraints on resources such as testing time (∼ 3–4 hours per round), basic equipment, skilled manpower and reagents.

The current low rate of COVID-19 infection in the world population [1] means that most samples tested are not infected, so that most tests are wasted on uninfected samples. Group testing is a process of pooling together samples of n different people into multiple pools, and testing the pools instead of each individual sample. A negative result on a pool implies that all samples participating in it were negative. This saves a huge amount of testing resources, especially with low infection rates. Group testing for medical applications has a

long history dating back to the 1940 s when it was proposed for testing of blood samples for syphilis [2]. Simple two-round group testing schemes have already been applied in the field by several research labs [3], [4] for COVID-19 testing. Such two-round group testing schemes require pooling of samples and a second round of sample handling for all samples in positive pools. This second round of sample handling can increase the time to result and be laborious to perform since it requires the technician to wear PPE one more time, do another round of RNA extraction, and PCR. In situations where the result needs to be delivered fast, a second round of sample handling and testing must be avoided. In such situations, these schemes are less attractive.

We present Tapestry, a novel combination of ideas from combinatorial group testing and compressed sensing (CS) [5] which uses the quantitative output of PCR tests to reconstruct the viral load of each sample in a single round. Tapestry has been validated with wet lab experiments with oligomers [6]. In this work, we elaborate on the results from the algorithmic perspective for the computer science and signal processing communities. Tapestry has a number of salient features which we enumerate below.

1) Tapestry delivers results in a single round of testing, without the need for a second confirmatory round, at clinically acceptable false negative and false positive rates. The number m of required tests is only $O(k \log n)$ for random binary pooling matrix constructions, as per compressed sensing theory for random binary matrices [7]. In the targeted use cases where the number of infected samples $k \ll n$, we see that $m \ll n$. However, our deterministic pooling matrix constructions based on Kirkman Triple Systems [8], [9] require fewer tests in practice (see Section III-F8 for a discussion on why this may be the case). Consequently we obtain significant savings in testing time and resources such as number of tests, quantity of reagents, and manpower.

2) Tapestry reconstructs relative viral loads i.e., ratio of viral amounts in each sample to the highest viral amount across pools. It is believed that super-spreaders and people with severe symptoms have higher viral load [10], [11], so this quantitative information might have epidemiological relevance.

3) Tapestry takes advantage of quantitative information in PCR tests. Hence it returns far fewer false positives than traditional binary group testing algorithms such as COMP (Combinatorial Orthogonal Matching Pursuit) [12], while maintaining clincally acceptable false negative rates. Furthermore, it takes advantage of the fact that a negative pool has viral load exactly zero. Traditional CS algorithms do not take advantage of this information. Hence, Tapestry demonstrates better sensitivity and specificity than CS algorithms.

4) Because each sample is tested in three pools, Tapestry can detect some degree of noise in terms of cross-contamination of samples and pipetting errors.

5) Tapestry allows PCR test measurements to be noisy. We develop a novel noise model to describe noise in PCR experiments. Our algorithms are tested on this noise model in simulation.

6) All tuning parameters for execution of the algorithms are inferred on the fly in a data driven fashion.

7) Each sample contributes to exactly three pools, and each pool has the same number of samples. This simplifies the experimental design, conserves samples, keeps pipetting overhead to a minimum, and makes sure that dilution due to pool size is in a manageable regime.

The organization of the paper is as follows. We first present a brief overview of the RT-PCR method in Section II. The precise mathematical definition of the computational problem being solved in this paper is then put forth in Section III-A. We describe traditional and CS-based group-testing algorithms for this problem in Section III-B, III-C and III-D. The Tapestry method is described in Section III-D. The sensing matrix design problem, as well as theoretical guarantees using Kirkman Triple Systems or random binary matrices, are described in Section III-F. Results on synthetic data are presented in Section IV. This is followed by results on data from lab experiments performed with oligomers to mimic the clinical situation as closely as possible. In Section V, we compare our work to recent related approaches. We conclude in Section VI with a glance through different scenarios where our work could be deployed. The supplemental material contains several additional experimental details as well as proofs of some theoretical results.

II. RT-PCR METHOD

We present here a brief summary of the RT-PCR process, referring to [13] for more details. In the RT-PCR method for COVID-19 testing, a sample in the form of naso- or oro-pharyngeal swabs is collected from a patient. The sample is then dispersed into a liquid medium. The RNA molecules of the virus present in this liquid medium are converted into complementary DNA (cDNA) via a process called reverse transcription. DNA fragments called primers complementary to cDNA from the viral genome are then added. They attach themselves to specific sections of the cDNA from the viral genome if the virus is present in the sample. The cDNA of these specific viral genes then undergoes a process of exponential amplification in an RT-PCR machine. Here, cDNA is put through several cycles of alternate heating and cooling in the presence of Taq polymerase and appropriate reagents. This triggers the creation of many new identical copies of specific portions of the target DNA, roughly doubling in number with every cycle of heating and cooling. The reaction volume contains sequence-specific fluorescent markers which report on the total amount of amplified DNA of the appropriate sequence. The resulting fluorescence is measured, and the increase can be observed on a computer screen in real time. The time when the amount of fluorescence exceeds the threshold level is known as the threshold cycle C_t, and is a quantitative readout from the experiment. A smaller C_t indicates

greater number of copies of the virus. Usually C_t takes values anywhere between 16 to 32 cycles in real experiments. PCR can detect even single molecules. A single molecule typically would have C_t value of around 40 cycles. A typical RT-PCR setup can test 96 samples in parallel. The test takes about 3-4 hours to execute.

III. TESTING METHODS
A. STATEMENT OF THE COMPUTATIONAL PROBLEM

Let x denote a vector of n elements where x_i is the viral load (i.e. viral amount) of the ith person. Throughout this paper we assume that only one sample per person is extracted. Hence x contains the viral loads corresponding to n different people. Note that $x_i = 0$ implies that the ith person is not infected. Due to the low infection rate for COVID-19 as yet even in severely affected countries [1], x is considered to be a sparse vector with at the most $k \ll n$ positive-valued elements. In group testing, small and equal volumes of the samples of a subset of these n people are pooled together according to a sensing or pooling matrix $A = (A_{ji})_{m \times n}$ whose entries are either 0 or 1. The viral loads of the pools will be given by:

$$z_j = \sum_{i=1}^{n} A_{ji}x_i = A^j x, \ 1 \leq j \leq m, \ 1 \leq i \leq n, \quad (1)$$

where $A_{ji} = 1$ if a portion of the sample of the ith person is included in the jth pool, and A^j is the jth row of A. In all, some $m < n$ pools are created and individually tested using RT-PCR. We now have the relationship $z = Ax$, where z is the m-element vector of viral loads in the mixtures, and A denotes a $m \times n$ binary 'pooling matrix' (also referred to as a 'sensing matrix' in CS literature). Note that each positive RT-PCR test will yield a noisy version of z_j, which we refer to as y_j. The relation between the 'clean' and noisy versions is given as follows (also see Eqn. (7)):

$$y_j = z_j(1+q)^{e_j} = (1+q)^{e_j} A^j x, \quad (2)$$

where $e_j \sim \mathcal{N}(0, \sigma^2)$ and $q \in (0, 1)$ is the fraction of viral cDNA that replicates in each cycle. The factor $(1+q)^{e_j}$ reflects the stochasticity in the growth of the numbers of DNA molecules during PCR. Here σ is known and constant. Equivalently for positive tests, we have:

$$\log y_j = \log(A^j x) + \log(1+q)e_j. \quad (3)$$

In case of negative tests, y_j as well as z_j are 0-valued, and no logarithms need be computed. In non-adaptive group testing, the core computational problem is to estimate x given y and A without requiring any further pooled measurements. It should be noted that though we have treated each element of x to be a fixed quantity, it is in reality a random variable of the form $x_i \sim \text{Poisson}(\lambda_i)$ where $\lambda_i \geq 0$. If matrix A contains only ones and zeros, this implies that $z_j \sim \text{Poisson}(A^j x)$ because the sum of Poisson random variables is also a Poisson random variable.

1) DERIVATION OF NOISE MODEL

For a positive pool j, the quantitative readout from RT-PCR is not its viral load but the observed cycle time t_j when its fluorescence reaches a given threshold F (see Section II). In order to be able to apply CS techniques (see Section III-C), we derive a relationship between the cycle time of a sample and its viral load. Because of exponential growth (see [14]), the number of molecules of viral cDNA in pool j at cycle time t, denoted by $v_j(t)$ is given by:

$$v_j(t) = z_j(1+q)^t. \quad (4)$$

Also, t is a real number, with $\lfloor t \rfloor$ indicating the number of PCR cycles that have passed, and $t - \lfloor t \rfloor$ indicating the fraction of wall-clock time within the current cycle. The fluorescence of the pool, $f_j(t)$, is directly proportional to the number of virus molecules $v_j(t)$. That is,

$$f_j(t) = Kv_j(t) = Kz_j(1+q)^t, \quad (5)$$

where K is a constant of proportionality. Suppose the fluorescence of pool j should reach the threshold value F at cycle time τ_j, according to Eqn. (5). Due to the stochastic nature of the reaction, as well as measurement error in the PCR machine, the threshold cycle output by the machine will not reflect this true cycle time. We model this discrepancy as Gaussian noise. Hence, the true cycle time τ_j and the observed cycle time t_j are related as $\tau_j = t_j + e_j$, where $e_j \sim \mathcal{N}(0, \sigma^2)$ as before. Now, since $f_j(\tau_j) = F$, using Eqn. (5), we have

$$F = Kz_j(1+q)^{\tau_j} = Ky_j(1+q)^{t_j}. \quad (6)$$

The latter equality is since we use the noisy cycle threshold t_j to compute viral load, where y_j is defined to be the noisy viral load of pool j. Hence we find

$$y_j = z_j(1+q)^{\tau_j - t_j} = z_j(1+q)^{e_j} = (1+q)^{e_j} A^j x, \quad (7)$$

obtaining the relationship from Eqn. (2).

Constants F and K are unknown. Hence it is not possible to directly obtain y_j from t_j without additional machine-specific calibration. However, we can find the ratio between the noisy viral loads of two pools using Eqn. (6). Let y_{min} be the noisy viral load of the pool with the minimum observed threshold cycle (t_{min}) among all pools. Then we define relative viral loads as:

$$\widetilde{y}_j = \frac{y_j}{y_{min}} = (1+q)^{t_{min} - t_j}, \widetilde{z}_j = \frac{z_j}{y_{min}}, \widetilde{x} = \frac{x}{y_{min}} \quad (8)$$

where $\widetilde{z}_j$ is the relative viral load of a pool, $\widetilde{y}_j$ is its noisy version, and $\widetilde{x}$ is the vector of relative viral loads of each sample. We note that due to Eqn. (7), the following relation holds:

$$\widetilde{y}_j = \widetilde{z}_j(1+q)^{e_j} = (1+q)^{e_j} A^j \widetilde{x}, \quad (9)$$

Hence we can apply CS techniques from Section III-C to determine the relative magnitudes of viral loads without knowing F and K. We provide more comments about the settings of various noise model parameters for our experiments, in Section IV, particularly in Section IV-A6.

B. COMBINATORIAL GROUP-TESTING

Combinatorial Orthogonal Matching Pursuit (COMP) is a Boolean nonadaptive group testing method [15, Sec. 2.3]. Here one uses the simple idea that if a mixture $\widetilde{y}_j$ tests negative then any sample $\widetilde{x}_i$ for which $A_{ji} = 1$ must be negative. Note that pools which test negative are regarded as noiseless observations, as argued in Section III-A1. The other samples are all considered to be positive. This algorithm guarantees that there are no 'false negatives'. However it can produce a very large number of 'false positives'. For example, a sample $\widetilde{x}_k$ will be falsely reported to be positive if every mixture $\widetilde{y}_j$ it is part of, also contains at least one other genuinely positive sample. The COMP algorithm is largely insensitive to noise. Moreover a small variant of it can also produce a list of 'high confidence positives,' after identifying the (sure) negatives. This happens when a positive mixture $\widetilde{y}_j$ contains only one sample $\widetilde{x}_i$, not counting the other samples which were declared sure negatives in the earlier step. Such a step of identifying 'high confidence positives' is included in the so-called **Definite Defectives** (DD) Algorithm [15, Sec. 2.4]. However DD labels all remaining items to be negative, potentially leading to a large number of false-negatives. The performance guarantees for COMP have been analyzed in [12] and show that COMP requires $ek(1 + \delta) \log n$ tests for an error probability less than $n^{-\delta}$ (see Section III-F8). This analysis has been extended to include the case of noisy test results as well [12]. However COMP can result in a large number of false positives if not enough tests are used, and it also does not predict viral loads.

C. COMPRESSED SENSING FOR POOLED TESTING

Group testing is intimately related to the field of compressed sensing (CS) [16], which has emerged as a significant sub-area of signal and image processing [5], with many applications in biomedical engineering [17]–[19]. In CS, an image or a signal x with n elements, is directly acquired in compressed format via m linear measurements of the form $y = Ax + \eta$. Here, the measurement vector y has m elements, and A is a matrix of size $m \times n$, and η is a vector of noise values. If x is a sparse vector with $k \ll n$ non-zero entries, and A obeys the so-called restricted isometry property (RIP), then *exact* recovery of x from y, A is possible [20] if $\eta = 0$. In the case of measurement noise, the recovery of x produces a solution that is provably close to the original x. A typical recovery problem P0 consists of optimizing the following cost function:

$$\min \|x\|_0 \text{ s.t. } \|y - Ax\|_2 \leq \varepsilon, \quad (10)$$

where ε is an upper bound (possibly a high probability upper bound) on $\|\eta\|_2$, and $\|x\|_0$ is the number of non-zero elements in x. In the absence of noise, a unique and exact solution to this problem is possible with as few as $2\tilde{}k$ measurements in y if x has k non-zero elements [20]. Unfortunately, this optimization problem P0 is NP-Hard and the algorithm requires brute-force subset enumeration. Instead, the following problem P1 (often

termed 'Basis Pursuit Denoising' or BPDN) is solved in practice:

$$\min \|x\|_1 \text{ s.t. } \|y - Ax\|_2 \leq \varepsilon. \quad (11)$$

P1 is a convex optimization problem which yields the same solution as the earlier problem (with similar conditions on x, A) at significantly lower computational cost, albeit with $O(k \log n)$ measurements (i.e. typically greater than $2\tilde{}k$) [5], [20].

The order k restricted isometry constant (RIC) of a matrix A is defined as the smallest constant δ_k, for which the following relationship holds for all k-sparse vectors x (i.e. all vectors with at the most k non-zero entries): $(1 - \delta_k)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_k)\|x\|_2^2$. The matrix A is said to obey the order k restricted isometry property (RIP) if δ_k is close to 0. This property essentially implies that no k-sparse vector (other than the zero vector) can lie in the null-space of A. Unique recovery of k-sparse signals requires that no $2\tilde{}k$-sparse vector lies in the nullspace of A [20]. A matrix A which obeys RIP of order $2\tilde{}k$ satisfies this property. It has been proved that matrices with entries randomly and independently drawn from distributions such as Rademacher or Gaussian, obey the RIP of order k with high probability [21], provided they have at least $O(k \log n)$ rows. There also exist **deterministic binary sensing matrix designs** (e.g. [22]) which require $O(\max(k^2, \sqrt{n}))$ measurements. However it has been shown recently [23] that the constant factors in the deterministic case are *significantly smaller* than those in the former random case when $n < 10^5$, making the deterministic designs more practical for typically encountered problem sizes. The solution to the optimization problems P0 and P1 in Eqns. (10) and (11) respectively, are provably robust to noise [5], and the recovery error decreases with decrease in noise magnitude. The error bounds for P0 in Eqn. (10) are of the form, for solution $\hat{x}$ [24]:

$$\frac{\varepsilon}{\sqrt{1 + \delta_{2\tilde{}k}}} \leq \|x - \hat{x}\|_2 \leq \frac{\varepsilon}{\sqrt{1 - \delta_{2\tilde{}k}}}, \quad (12)$$

whereas those for P1 in Eqn. (11) have the form [24]:

$$\|x - \hat{x}\|_2 \leq \varepsilon \zeta(\delta_{2\tilde{}k}). \quad (13)$$

Here $\zeta(\delta_{2\tilde{}k})$ is a monotonically increasing function of $\delta_{2\tilde{}k} \in (0, 1)$ and has a small value in practice.

The Restricted Isometry Property as defined above is also known as RIP-2, because it uses the ℓ_2-norm. Many other sufficient conditions for recovery of k-sparse vectors exist. We define the following which we use later in Section III-F and supplemental Section S.V to prove theoretical guarantees of our method.

Definition 1: **RIP-1:** [25, Defn. 8] A $m \times n$ matrix A is said to obey RIP-1 of order k if $\exists \delta_k \in (0, 1)$ such that for all k-sparse vectors $x \in \mathbb{R}^n$,

$$\|x\|_1 \leq \|Ax\|_1 \leq (1 + \delta_k)\|x\|_1$$

.

Definition 2: **RNSP:** [23, Eqn. 12] A $m \times n$ matrix A is said to obey the Robust Nullspace Property (RNSP) of order

---
Algorithm 1: Tapestry Method.

1:   Input: n samples, $m \times n$ pooling matrix A
2:   Perform pooling according to pooling matrix A and create m pooled samples
3:   Run RT-PCR test on these m pooled samples and receive $m \times 1$ vector of cycle threshold values t
4:   Compute $m \times 1$ vector of relative viral loads $\widetilde{y}$ from t
5:   Use COMP to filter out negative tests and sure negative samples. Compute submatrix $A_{\bar{\mathcal{X}},\bar{\mathcal{Y}}}, \widetilde{y}_{\bar{\mathcal{Y}}}$ and list $\mathcal{HCP}$ of 'high-confidence positives' along with their viral loads (see Section III-B).
6:   Use a CS decoder to recover relative viral loads $\widetilde{x}_{\bar{\mathcal{X}}}$ from $\widetilde{y}_{\bar{\mathcal{Y}}}, A_{\bar{\mathcal{X}},\bar{\mathcal{Y}}}$
7:   Compute $n \times 1$ relative viral load vector $\widetilde{x}$ by setting its entries from $\widetilde{x}_{\bar{\mathcal{X}}}$, and setting remaining entries to 0.
8:   **return** $\widetilde{x}, \mathcal{HCP}$.
---

k if $\exists \; \rho < 1$ and $\tau > 0$ such that for all $x \in \mathbb{R}^n$ it holds that

$$||x_S||_2 \leq \rho||x_{\bar{S}}||_1 + \tau||Ax||_2$$

for all $S \subset \{1 \ldots n\}$ with $|S| \leq k$.

Definition 3: ℓ_2-RNSP: [7, Defn. 1] A $m \times n$ matrix A is said to obey the ℓ_2-robust Nullspace Property (ℓ_2-RNSP) of order k if $\exists \; \rho \in (0, 1)$ and $\tau > 0$ such that for all $x \in \mathbb{R}^n$ it holds that

$$||x_S||_2 \leq \frac{\rho}{\sqrt{k}}||x_{\bar{S}}||_1 + \tau||Ax||_2$$

for all $S \subset \{1 \ldots n\}$ with $|S| \leq k$.

Over the years, a variety of different techniques for compressive recovery have been proposed. We use some of these for our experiments in Section III-D. These algorithms use different forms of sparsity and incorporate different types of constraints on the solution.

D. CS AND TRADITIONAL GT COMBINED

The complete pipeline of the Tapestry method is presented in Algorithm 1. First, a wet lab technician performs pooling of n samples into m pools according to a $m \times n$ pooling matrix A. Then they run the RT-PCR test on these m pools (in parallel). The output of the RT-PCR tests – the threshold cycle (C_t) values of each pool – is processed to find the relative viral load vector $\widetilde{y}$ of the m pools (as shown in Eqn. (8)). This is given as input to the Tapestry decoding algorithm, which outputs a sparse relative viral load vector $\widetilde{x}$.

The Tapestry decoding algorithm, our approach toward group-testing for COVID-19, involves a two-stage procedure.[1] In the first stage, we apply the COMP algorithm described in Section III-B, to identify the sure negatives (if any) in $\widetilde{x}$ to form a set $\mathcal{X}$. Let $\mathcal{Y}$ be the set of zero-valued

---
[1]The two-stage procedure is purely algorithmic. It does not require two consecutive rounds of testing in a lab.

measurements in $\widetilde{y}$ (i.e. negative tests). Please refer to Section III-A1 for the definition of $\widetilde{x}, \widetilde{y}$. Moreover, we define $\bar{\mathcal{X}}, \bar{\mathcal{Y}}$ as the complement-sets of $\mathcal{X}, \mathcal{Y}$ respectively. Also, let $y_{\bar{\mathcal{Y}}}$ be the vector of $m - |\mathcal{Y}|$ measurements which yielded a positive result. Let $x_{\bar{\mathcal{X}}}$ be the vector of $n - |\mathcal{X}|$ samples, which does not include the $|\mathcal{X}|$ surely negative samples. Let $A_{\bar{\mathcal{X}},\bar{\mathcal{Y}}}$ be the submatrix of A, having size $(m - |\mathcal{Y}|) \times (n - |\mathcal{X}|)$, which excludes rows corresponding to zero-valued measurements in y and columns corresponding to negative elements in x. In the second stage, we apply a CS algorithm to recover $\widetilde{x}_{\bar{\mathcal{X}}}$ from $\widetilde{y}_{\bar{\mathcal{Y}}}, A_{\bar{\mathcal{X}},\bar{\mathcal{Y}}}$. *To avoid symbol clutter, we henceforth just stick to the notation* y, x, A, m, n, *even though they respectively refer to* $\widetilde{y}_{\bar{\mathcal{Y}}}, \widetilde{x}_{\bar{\mathcal{X}}}, A_{\bar{\mathcal{X}},\bar{\mathcal{Y}}}, m - |\mathcal{Y}|, n - |\mathcal{X}|$.

Note that the CS stage following COMP is very important for the following reasons:

1) COMP typically produces a large number of false positives. The CS algorithms help reduce the number of false positives as we shall see in later sections.
2) COMP does not estimate viral loads, unlike CS algorithms.
3) In fact, unlike CS algorithms, COMP treats the measurements in y as also being binary, thus discarding a lot of useful information.
4) COMP preserves the RIP-1, RIP-2, RNSP, and ℓ_2-RNSP of the pooling matrix, i.e. if A obeys any of RIP-1, RIP-2, RNSP or ℓ_2-RNSP of order k, then $A_{\bar{\mathcal{X}},\bar{\mathcal{Y}}}$ also obeys the same property of the same order k with the same parameters. We formalize and prove these claims in the supplemental Section S.V.

However, the COMP algorithm prior to applying the CS algorithm is also very important for the following reasons:

1) Viral load in negative pools is exactly 0. COMP identifies the sure negatives in x from the negative measurements in y. Traditional CS algorithms do not take advantage of this information, since they assume all tests to be noisy (Eqns. (10) and (11)). It is instead easier to discard the obvious negatives before applying the CS step.
2) Since COMP identifies the sure negatives, therefore, it effectively reduces the size of the problem to be solved by the CS step from (m, n) to $(m - |\mathcal{Y}|, n - |\mathcal{X}|)$.
3) In a few cases, a (positive) pool in $\bar{\mathcal{Y}}$ may contain only one contributing sample in $\bar{\mathcal{X}}$, after negatives have been eliminated by COMP. Such a sample is called a 'high-confidence positive,' and we denote the list of high-confidence positives as $\mathcal{HCP}$. In rare cases, the CS decoding algorithms we employed (see further in this section) did not recognize such a positive. However, such samples will still be returned by our algorithm as positives, in the set $\mathcal{HCP}$ (see last step of Alg. 1, and 'definite defectives' in Section III-B).

For CS recovery, we employ one of the following algorithms after COMP: the non-negative LASSO (NNLASSO), non-negative orthogonal matching pursuit (NNOMP), Sparse Bayesian Learning (SBL), and non-negative absolute deviation regression (NNLAD). For problems of small size, we also apply

a brute force (BF) search algorithm to solve a problem similar to P0 from Eqn. (10) combinatorially.

1) THE NON-NEGATIVE LASSO (NNLASSO)

The LASSO (least absolute shrinkage and selection operator) is a penalized version of the constrained problem P1 in Eqn. (11), and seeks to minimize the following cost function:

$$J_{lasso}(x; y, A) := \|y - Ax\|_2^2 + \lambda\|x\|_1. \tag{14}$$

Here λ is a regularization parameter which imposes sparsity in x. The LASSO has rigorous theoretical guarantees [26] (chapter 11) for recovery of x as well as recovery of the support of x (i.e. recovery of the set of non-zero indices of x). Given the non-negative nature of x, we implement a variant of LASSO with a non-negativity constraint, leading to the following optimization problem:

$$J_{nnlasso}(x; y, A) := \|y - Ax\|_2^2 + \lambda\|x\|_1 \text{ s.t. } x \geq 0. \tag{15}$$

Selection of λ: There are criteria defined in [26] for selection of λ under iid Gaussian noise, so as to guarantee statistical consistency. However, in practice, **cross-validation** (CV) can be used for optimal choice of λ in a purely data-driven fashion from the available measurements. The details of this are provided in the supplemental Section S.III.

2) NON-NEGATIVE ORTHOGONAL MATCHING PURSUIT (NNOMP)

Orthogonal Matching Pursuit (OMP) [27] is a greedy approximation algorithm to solve the optimization problem in Eqn. (10). Rigorous theoretical guarantees for OMP have been established in [28]. OMP proceeds by maintaining a set $\mathcal{H}$ of 'selected coefficients' in x corresponding to columns of A. In each round a column of A is picked greedily, based on the criterion of maximum absolute correlation with a residual vector $r := y - \sum_{k \in \mathcal{H}} A_k \hat{x}_k$. Each time a column is picked, *all* the coefficients extracted so far (i.e. in set $\mathcal{H}$) are updated. This is done by computing the orthogonal projection of y onto the subspace spanned by the columns in $\mathcal{H}$. The OMP algorithm can be quite expensive computationally. Moreover, in order to maintain non-negativity of x, the orthogonal projection step would require the solution of a non-negative least squares problem, further adding to computational costs. However, a fast implementation of a non-negative version of OMP (NNOMP) has been developed in [29], which is the implementation we adopt here. For the choice of ε in Eqn. (10), we can use CV as described in Section III-D1.

3) SPARSE BAYESIAN LEARNING (SBL)

Sparse Bayesian Learning (SBL) [30], [31] is a non-convex optimization algorithm based on Expectation-Maximization (EM) that has empirically shown superior reconstruction performance to most other CS algorithms with manageable computation cost [32]. In SBL, we consider the case of Gaussian

noise in y and a Gaussian prior on elements of x, leading to:

$$p(y|x) = \frac{\exp(-\|y - Ax\|_2^2/(2\sigma^2))}{(2\pi\sigma^2)^{n/2}} \tag{16}$$

$$p(x_i; \varphi_i) = \frac{\exp(-x_i^2/(2\varphi_i))}{\sqrt{2\pi\varphi_i}}; \varphi_i \geq 0. \tag{17}$$

Since both x and φ (the vector of the $\{\varphi_i\}_{i=1}^n$ values) are unknown, the optimization for these quantities can be performed using an EM algorithm. In the following, we shall denote $\Phi := \text{diag}(\varphi)$. Moreover, we shall use the notation $\Phi^{(l)}$ for the estimate of Φ in the lth iteration. The E-step of the EM algorithm here involves computing $Q(\Phi|\Phi^{(l)}) := E_{x|y; \Phi^{(l)}} \log p(y, x; \Phi)$. It is to be noted that the posterior distribution $p(x|y; \Phi^{(l)})$ has the form $\mathcal{N}(\mu, \Sigma)$ where $\mu := \Sigma A^T y/\sigma^2$ and $\Sigma := (A^T A/\sigma^2 + (\Phi^{(l)})^{-1})^{-1}$. The M-step involves maximization of $Q(\Phi|\Phi^{(l)})$, leading to the update $\Phi^{(l+1)} = \text{diag}(\mu_i^2 + \Sigma_{ii})$. The E-step and M-step are executed alternately until convergence. Convergence to a fixed-point is guaranteed, though the fixed point may or may not be a local minimum. However, all local minima are guaranteed to produce sparse solutions for x (even in the presence of noise) because most of the φ_i values shrink towards 0. The SBL procedure can also be modified to dynamically update the noise variance σ^2 (as followed in this paper), if it is unknown. All these results can be found in [31]. Unlike NNLASSO or NNOMP, the SBL algorithm from [31] expressly requires Gaussian noise. However we use it as is in this paper for the simplicity it affords. Unlike NNOMP or NNLASSO, there is no explicit non-negativity constraint imposed in the basic SBL algorithm. In our implementation, the non-negativity is simply imposed at the end of the optimization by setting to 0 any negative-valued elements in μ, though more principled, albeit more computationally heavy, approaches such as [33] can be adopted.

4) NON-NEGATIVE ABSOLUTE DEVIATION REGRESSION (NNLAD)

The Non-Negative Absolute Deviation Regression (NNLAD) [34] and Non-negative Least squares (NNLS) [7] seek to respectively minimize

$$J_{nnlad}(x; y, A) := \|y - Ax\|_1 \text{ s.t. } x \geq 0, \tag{18}$$

$$J_{nnls}(x; y, A) := \|y - Ax\|_2 \text{ s.t. } x \geq 0. \tag{19}$$

It has been shown in [34] that NNLAD is sparsity promoting for certain conditions on the sensing matrix A, and that its minimizer x^* obeys bounds of the form $||x - x^*||_1 \leq C||\eta||_1$, where C is a constant independent of x, x^*, η, y. A salient feature of NNLAD/NNLS is that they do not require any parameter tuning. This property makes them useful for matrices of smaller size where cross-validation may be unreliable.

E. GENERALIZED BINARY SEARCH TECHNIQUES

There exist *adaptive group testing* techniques which can determine k infected samples in $O(k \log n)$ tests via repeated

binary search. These techniques are impractical in our setting due to their sequential nature and large pool sizes. We provide details of these techniques in the supplemental Section S.II. We also compare with a two-stage approach called Dorfman's method [2] in Section IV-A7.

F. SENSING MATRIX DESIGN

1) PHYSICAL REQUIREMENTS OF THE SENSING MATRIX

The sensing matrix A must obey some properties specific to this application such as being non-negative. For ease and speed of pipetting, it is desirable that the entries of A be (1) binary (where $A_{ji} = 0$ indicates that sample i did not contribute to pool j, and $A_{ji} = 1$ indicates that a fixed volume of sample i was pipetted into pool j), and (2) sparse. Sparsity ensures that not too many samples contribute to a pool, and that a single sample does not contribute to too many pools. The former is important because typically the volume of sample that is added in a PCR reaction is fixed. Increasing pool size means each sample contributes a smaller fraction of that volume. This leads to dilution which manifests as a shift of the C_t value towards larger numbers. If care is not taken in this regard, this can affect the power of PCR to discriminate between positive and negative samples. The latter is important because contribution of one sample to a large number of pools could lead to depletion of sample.

2) RIP-1 OF EXPANDER GRAPH ADJACENCY MATRICES

The Restricted Isometry Property (RIP-2) of sensing matrices is a sufficient condition for good CS recovery as described in Section III-C. However the matrices which obey the aforementioned physical constraints are not guaranteed to obey RIP-2. Instead, we consider sensing matrices which are adjacency matrices of **expander graphs**. A left-regular bipartite graph $G((\mathcal{V}_\mathcal{I}, \mathcal{V}_\mathcal{O}), \mathcal{E} \subseteq \mathcal{V}_\mathcal{I} \times \mathcal{V}_\mathcal{O})$ with degree of each vertex in $\mathcal{V}_\mathcal{I}$ being d, is said to be a (k, ϵ)-**unbalanced expander graph** for some integer $k > 0$ and some real-valued $\epsilon \in (0, 1)$, if for every subset $\mathcal{S} \subseteq \mathcal{V}_\mathcal{I}$ with $|\mathcal{S}| \leq k$, we have $|N(\mathcal{S})| \geq (1 - \epsilon)d|\mathcal{S}|$. Here $N(\mathcal{S})$ denotes the union set of neighbors of all nodes in $\mathcal{S}$. Intuitively a bipartite graph is an expander if every 'not too large' subset has a 'large' boundary. It can be proved that a randomly generated left-regular bipartite graph with $|\mathcal{V}_\mathcal{O}| \geq O(k \log n)$, $n = |\mathcal{V}_\mathcal{I}|$ is an expander, with high probability [35], [36]. Moreover, it has been shown in [25, Thm. 1] that the scaled adjacency matrix A/d of a (k, ϵ)-unbalanced expander graph obeys RIP-1 (Defn. 1) of order k. Here columns of A correspond to vertices in $\mathcal{V}_\mathcal{I}$, and rows correspond to vertices in $\mathcal{V}_\mathcal{O}$. That is, for any k-sparse vector x, the following relationship holds: $\|x\|_1 \leq \|Ax\|_1/d \leq (1 + C\epsilon)\|x\|_1$ for some absolute constant $C > 1$. This property again implies that the null-space of A cannot contain vectors that are 'too sparse' (apart from the zero-vector). This summarizes the motivation behind the use of expanders in compressive recovery of sparse vectors, and also in group testing [25].
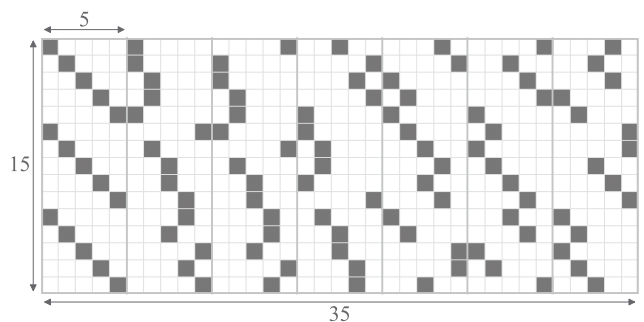

FIGURE 1. A full Kirkman matrix with $m = 15$ rows and $n = \binom{m}{2}/3 = 35$ columns. Each cell denotes an entry of the matrix, with white cells denoting the location of a 0 entry and the greyed out cells indicating the location of a 1 entry. Each column has exactly 3 entries with value 1. Each row has 7 entries with value 1. There are $(m - 1)/2 = 7$ groups of columns, each consisting of $m/3 = 5$ columns. Each row in a column group has exactly one 1 entry. Matrices of size 15×20, 15×25, 15×30 or 15×35 may be served by choosing the first 4, 5, 6, or 7 column groups, while keeping the number of 1 entries in each row equal.

3) MATRICES DERIVED FROM KIRKMAN TRIPLE SYSTEMS

Although randomly generated left-regular bipartite graphs are expanders, we would need to verify whether a *particular* such graph is a good expander, which may take prohibitively long in practice [35]. In the application at hand, this can prove to be a critical limitation since matrices of various sizes may have to be served, depending on the number of samples arriving in that batch at the testing centre, and the number of tests available to be performed. Hence, we have chosen to employ deterministic procedures to design such matrices, based on objects from combinatorial design theory known as **Kirkman triples** (see [8], [9]).

We first recall Kirkman Triple Systems (an example of which is illustrated in Fig. 1) which are Steiner Triple Systems with an extra property. Steiner Triple Systems consist of $n = \binom{m}{2}/3$ column vectors with m elements each, with each entry being either 0 or 1 such that each column has exactly three 1 s, every pair of rows has dot product equal to 1 and every pair of columns has dot product at most 1 [37]. This means that each column of a Steiner Triple System corresponds to a triplet of rows (i.e. contains exactly three 1 s), and every pair of rows occurs together in exactly one such triplet (i.e. for every pair of rows indexed by i, j, there exists exactly one column index k for which $A_{ik} = A_{jk} = 1$). If the columns of a Steiner Triple System can be arranged such that the sum of columns from i to $i + m/3 - 1$ equals $\mathbf{1} \in \mathbb{R}^m$ for every $i \equiv 1$ modulo $m/3$ then the Steiner Triple System is said to be **resolvable**, and is known as a **Kirkman Triple System** [8]. That is, the set of columns of a Kirkman Triple System can be partitioned into $(m - 1)/2$ disjoint groups, each consisting of $m/3$ columns, such that each row has exactly one 1 entry in a given such group of columns. Because of this property, we may choose any l such groups of columns of a Kirkman Triple System to form a $m \times n$ matrix, $n > m$, with $n = lm/3$, and $3 < l \leq (m - 1)/2$, while keeping the number of 1 entries in each row the same. From here on, we refer to such matrices as

Kirkman matrices. If $l = (m-1)/2$, then we refer to it as a **full** Kirkman matrix, else it is referred to as a **partial** Kirkman matrix. Note that in a partial Kirkman matrix, the dot product of any two rows may be at most 1, whereas in a full Kirkman matrix, it must be equal to 1.

Notice that $m = 6t + 3$ for some $t \in \mathbb{Z}_{\geq 0}$ for a Kirkman Triple System to exist, since $m - 1$ must be divisible by 2, and m must be divisible by 3. This, and the existence of Kirkman Triple Systems for all $t \in \mathbb{Z}_{\geq 0}$ have been proven in [9]. Explicit constructions of Kirkman Triple Systems for $m \leq 99$ exist [8]. Generalizations of Kirkman Triple Systems under the name of the Social Golfer Problem is an active area of research (see [38], [39]). The Social Golfer Problem asks if it is possible for $g \times p$ golfers to play in g groups of p players each for w weeks, such that no two golfers play in the same group more than once [40, Sec. 1.1]. Kirkman Triple Systems with m rows and $\binom{m}{2}/3$ columns are a solution to the Social Golfer Problem for the case when $p = 3$, $g = m/3$ and $w = (m-1)/2$. Full or partial Kirkman matrices may be constructed via greedy search techniques used for solving the Social Golfer Problem (such as in [41]). Previously, Kirkman matrices have been proposed for use as Low-Density Parity Check codes in [42], due to high girth[2] of Kirkman matrix bipartite graphs and the ability to serve only part of the matrix while keeping the row weights[3] equal. Matrices derived from Steiner Triple Systems have previously been used for pooled testing for transcription regulatory network mapping in [43]. Further, matrices derived from Steiner Systems [44], a generalization of Steiner Triple Systems, have been proposed for optimizing 2-stage binary group testing in [45].

4) RIP-1 AND EXPANSION PROPERTIES OF KIRKMAN MATRICES

We show that Kirkman matrix bipartite graphs are (k, ϵ)-unbalanced expanders, with $\epsilon = (k-1)/2\tilde{}d$, where d is the left-degree of the graph and is 3 for Kirkman matrices. Given a set S of column vertices such that $|S| \leq k$, we note that the size of the union set of neighbours of S, $|N(S)|$, is at least $|S|d - pr$, where $p = \binom{|S|}{2}$ is the number of (unordered) pairs of columns in S, and r is the maximum number of row vertices in common between any two column vertices. For a Kirkman matrix, since any two columns have dot product at most 1, hence $r = 1$. Therefore, $|N(S)| \geq d|S|(1 - (|S| - 1)/2\tilde{}d)$. Since $|S| \leq k$, therefore $|N(S)| \geq d|S|(1 - (k-1)/2\tilde{}d)$. This implies that Kirkman matrix biparite graphs are (k, ϵ)-unbalanced expanders, with $\epsilon = (k-1)/2\tilde{}d$. If we put in the requirement that $d = 3$ for Kirkman matrices and $\epsilon < 1$, we find that $k < 7$. Hence it follows from [25, Thm. 1] that the scaled Kirkman matrix has RIP-1 of order k for $k < 7$ and $\epsilon = (k-1)/6$. This suggests exact recovery for upto 3 infected samples using CS. However, in practice, we observe that using our method we are able to recover much higher number of positives, at the cost of an

acceptable number of false positives and rare false negatives (Section IV).

5) OPTIMALITY OF GIRTH 6 MATRICES

A Steiner Triple System bipartite graph does not have a cycle of length 4. If it did, then there would exist two rows a and b, and two columns u and v of the Steiner Triple System matrix A such that $A_{au} = A_{bu} = 1$ and $A_{av} = A_{bv} = 1$. This would violate the property that dot product of any two rows of the Steiner Triple System must be equal to 1. Furthermore, [42, Lemma 1] show that Steiner Triple System bipartite graphs have girth equal to 6. Since Kirkman Triple Systems are resolvable Steiner Triple Systems (see definitions earlier in this section), their bipartite graphs also have girth equal to 6. For a bipartite graph constructed from a partial Kirkman matrix, the girth is at least 6, since dropping some column vertices will not introduce new cycles in the graph. Furthermore, it is shown in [23, Thm. 10] that adjacency matrices of left-regular graphs with girth at least 6 satisfy RNSP (Defn. 2) of order k (for suitable k). Consequently, they may be used for CS decoding [23, Thm. 5]. They also give lower bounds on the number of rows m of left-regular bipartite graph matrices whose column weight[4] is more than 2, for them to have high girth and consequently satisfy RNSP of order k, given k and n [23, Eqn. 32, 33]. Given k and n, these lower bounds are minimized for graphs of girth 6 and 8, and the bounds are, respectively, $m \geq k\sqrt{n}$ and $m \geq k^{3/2}\sqrt{n}$ ([23, Eqn. 37]). However, with the additional requirement that $m < n$ for CS, it is found that girth 6 matrices can recover $k < \sqrt{n}$ defects, while girth 8 matrices can only recover $k < \sqrt[3]{n}$ defects. Hence, matrices whose bipartite graphs have girth equal to 6 are optimal in this sense. Full Kirkman matrix bipartite graphs are left-regular and have girth 6, as argued earlier, and hence they satisfy RNSP, may be used for compressive sensing, and are optimal in the sense of being able to handle most number of defects while minimizing the number of measurements. We note that since we employ Kirkman triples, each column has only three 1 s. The theoretical guarantees for such matrices hold for signals with ℓ_0 norm less than or equal to 2. However, we have obtained acceptable false positive and false negative rates in practice for much larger sparsity levels, as will be seen in Section IV.

6) DISJUNCTNESS PROPERTY OF KIRKMAN MATRICES

In order for a matrix to be suitable for our method, it should not only be good for CS decoding algorithms, but also for COMP. Kirkman matrices are 2-disjunct, and can recover up to 2 defects exactly using COMP. In a k-disjunct matrix, there does not exist any column such that its support is a subset of the union of the support of k other columns [15]. Matrices which are k-disjunct have exact support recovery guarantee for k-sparse vectors, using COMP (see [15]). Disjunctness follows from the following properties of Kirkman matrices

---

[2] The girth of a graph is equal to the length of the shortest cycle in it.
[3] defined as the number of 1 entries in a row

[4] defined as the number of 1 entries in a column

TABLE 1. Performance of COMP and D$_D$ (On Synthetic Data) for 45 × 105 Kirkman Triple Matrix. For Each Criterion and Each *k* Value, Mean and Standard Deviation Values are Reported Across 1000 Signals

k	RMSE	#FN	#FP	Sens.	Spec.	#$\mathcal{HCP}$
5	1.000 ± 0.000	0.0 ± 0.0	1.6 ± 1.2	1.0000 ± 0.0000	0.9983 ± 0.0013	4.7
8	1.000 ± 0.000	0.0 ± 0.0	7.9 ± 3.0	1.0000 ± 0.0000	0.9918 ± 0.0031	4.3
10	1.000 ± 0.000	0.0 ± 0.0	15.3 ± 4.5	1.0000 ± 0.0000	0.9839 ± 0.0048	2.5
12	1.000 ± 0.000	0.0 ± 0.0	25.3 ± 6.7	1.0000 ± 0.0000	0.9733 ± 0.0070	1.1
15	1.000 ± 0.000	0.0 ± 0.0	46.1 ± 10.3	1.0000 ± 0.0000	0.9512 ± 0.0109	0.2
17	1.000 ± 0.000	0.0 ± 0.0	62.3 ± 13.7	1.0000 ± 0.0000	0.9340 ± 0.0146	0.1
20	1.000 ± 0.000	0.0 ± 0.0	91.5 ± 18.1	1.0000 ± 0.0000	0.9028 ± 0.0192	0.0

– that two columns in a Kirkman matrix have at most one row in common with an entry of 1, and that each column has exactly three 1 entries. Consider R_a, R_b, and R_c, the sets of rows for which the three columns *a*, *b* and *c* respectively have a 1 entry. Note that $|R_a| = |R_b| = |R_c| = 3$, and $|R_p \cap R_q| \leq 1$ for $p, q \in \{a, b, c\}$, $p \neq q$. If $R_c \subseteq R_a \cup R_b$, then either $|R_c \cap R_a| > 1$ or $|R_c \cap R_b| > 1$, which presents a contradiction.

Empirically we find that even for $k > 2$, COMP reports only a small fraction of the total number of samples as positives when using Kirkman matrices (Table 1). In Section S.XIV (Proposition 6) of the supplemental material, we prove that *if a fraction $f \in (0, 1)$ of the tests come out to be positive, then* COMP*reports strictly less than fraction f^2 of the samples as positive for a full Kirkman matrix*. This provides intuition behind why Kirkman matrices may be well-suited for our combined COMP + CS method, since most samples are already eliminated by COMP. On the other hand, CS decoding (without the earlier COMP step) on the full Kirkman matrix does not perform as well, as shown in the supplemental Section S.IX.

7) ADVANTAGES OF USING KIRKMAN MATRICES

As we have seen in earlier sections, Kirkman matrices are suitable for use in compressed sensing due to their expansion, RIP-1 and high girth properties, as well as for binary group testing due to disjunctness. Furthermore, the dot product between two columns of a Kirkman matrix being at most 1 ensures that no two samples participate in more than one test together. This has favourable consequences in terms of placing an upper bound on the **mutual coherence** of the matrix, defined as:

$$\mu(A) := \max_{i \neq j} \frac{|A_i{}^t A_j|}{\|A_i\|_2 \|A_j\|_2}, \qquad (20)$$

where A_i refers to the ith column of A. Matrices with lower $\mu(A)$ values have lower values of worst case upper bounds on the reconstruction error [46]. These bounds are looser than those based on the RIC that we saw in previous sections. However, unlike the RIC, the mutual coherence is efficiently computable.

A practical benefit of Kirkman triples that is not shared by Steiner triples is that the former can be served for number of samples far less than $n = \binom{m}{2}/3$ while keeping pools balanced (i.e. ensuring that each pool is created from the same number of samples). In fact, we can choose n to be any integer multiple of $m/3$, and ensure that every pool gets the same number of samples, as discussed in section III-F3. Notice that

the expansion, RIP-1, high girth and disjunctness properties hold for full as well as partial Kirkman matrices, as proven in previous sections. This allows us to characterize the properties of the full Kirkman matrix, and use that analysis to predict how it will behave in the clinical situation where the pooling matrix to be served may require very specific values of m, n depending on the prevalence rate.

Column weight: Kirkman matrices have column weight equal to 3 - that is, each sample goes to 3 pools. It is possible to construct matrices with higher number of pools per sample (such as those derived from the Social Golfer Problem [38], which will retain several benefits of the Kirman matrices: (1) They would have the ability to serve only part of the matrix; (2) They would retain the the expander and RIP-1 properties, following a proof similar to the one in Section III-F4; (3) They would not have any 4-cycles in the corresponding bipartite graph, following a similar argument as in Section III-F5; and (4) They would possess the disjunctness property following a proof similar to the one in Section III-F6). Nevertheless, the time and effort needed for pooling increases with more pools per sample. Further, higher pools per sample will come at the cost of a larger number of tests (if pool size is kept constant), or larger pool size (if number of tests is kept constant). Higher number of tests is undesirable for obvious reasons, while larger pool size may lead to dilution of the sample within a pool, leading to individual RT-PCR tests failing.

8) OPTIMAL BINARY SENSING MATRICES WITH RANDOM CONSTRUCTION

While Kirkman matrices which satisfy RNSP of order k must have at least $k\sqrt{n}$ measurements, we can get much better bounds in theory if we use random constructions. From [7, Prop. 10] we see that with high probability, 0/1 Bernoulli(p) matrices need only $O(k \log n)$ measurements in order to satisfy ℓ_2-RNSP (Defn. 3) of order k, with $p \in (0, 1)$ being the probability with which each entry of the matrix is independently 1.

In the supplemental Section S.V, we prove that ℓ_2-RNSP is preserved by COMP. That is, the reduced matrix $A_{\bar{x}, \bar{y}}$ obeys ℓ_2-RNSP of order k with the same parameters as the original matrix A. Hence our method only needs $O(k \log n)$ measurements for robust recovery of k-sparse vectors with such random matrix constructions. Bernoulli(p) matrices are also good for COMP – [12, Thm. 4] shows that Bernoulli(p) matrices with $p = 1/k$ need only $O(k \log n)$ measurements for

exact support recovery of k-sparse vectors with COMP with vanishingly small probability of error.

In practice, we observe that Kirkman matrices perform better than Bernoulli(p) matrices using our method in the regime of our problem size. This gap between theory and practice may be arising due to the following reasons: (1) The $k\sqrt{n}$ lower bound for Kirkman triples is for a sufficient but not necessary condition for sparse recovery; (2) The $O(k \log n)$ may be ignoring a very large constant factor which affects the performance of moderately-sized problems such as the ones reported in this paper; and (3) The theoretical bounds are for exact recovery with vanishingly small error, whereas we allow some false positives and rare false negatives in our experiments. Similar comparisons between binary and Gaussian random matrices have been recently put forth in [23]. Moreover, the average column weight of Bernoulli(p) matrices is pm, where m is the number of measurements. This is typically much higher than column weight 3 of Kirkman matrices and hence undesirable (see Section III-F7). In the supplemental Section S.VI, we compare the performance of Kirkman matrices with Bernoulli(0.1) and Bernoulli(0.5) matrices.

9) MUTUAL COHERENCE OPTIMIZED SENSING MATRICES

As mentioned earlier, the mutual coherence from Eqn. (20) is efficient to compute and optimize over. Hence, there is a large body of literature on designing CS matrices by minimizing $\mu(A)$ w.r.t. A, for example [47]. We followed such a procedure for designing sensing matrices for some of our experimental results in Section IV-B. For this, we follow simulated annealing to update the entries of A, starting with an initial condition where A is a random binary matrix. For synthetic experiments, we compared such matrices with Bernoulli(p) random matrices, adjacency matrices of biregular random sparse graphs (i.e. matrices in which each column has the same weight, and each row has the same weight - which may be different than the column weight), and Kirkman matrices. We found that matrices of Kirkman triples perform very well empirically in the regime of sizes we are interested in, besides facilitating easy pipetting, and hence the results are reported using only Kirkman matrices.

IV. EXPERIMENTAL RESULTS

In this section, we show a suite of experimental results on synthetic data as well as on real data.

A. RESULTS ON SYNTHETIC DATA

1) CHOICE OF SENSING MATRIX

Recall from section II that a typical RT-PCR setup can test 96 samples in parallel. Three of these tests are used as control by the RT-PCR technician in order to have confidence that the RT-PCR process has worked. Hence, in order to optimize the available test bandwidth of the RT-PCR setup, the number of tests we perform in parallel should be ≤ 93, and as close to 93 as possible. Since in Kirkman matrices, the number of rows must be $6t + 3$ for some $t \in \mathbb{Z}_{\geq 0}$, hence we choose 93.

With this choice, the number of samples tested n has to be a multiple of $93/3 = 31$, hence we chose $n = 961$. This matrix is not a full Kirkman matrix – a full matrix with 93 rows will have 1426 columns. However, we keep the number of columns of the matrix under 1000 due to challenges in pooling large number of samples. Furthermore, $n = 961$, $m = 93$ satisfies more than 10x factor improvement in testing while detecting 1% infected samples with reasonable sensitivity and specificity and is in a regime of interest for widespread screening or repeated testing.

We also present results with a 45×105 partial Kirkman matrix in the supplemental Section S.VIII. This matrix gives 2.3x improvement in testing while detecting 9.5% infected samples with reasonable sensitivity and specificity. Further, two such batches of 105 tests in 45 pools may be run in parallel in a single RT-PCR setup.

2) SIGNAL/MEASUREMENT GENERATION

For the case of synthetic data, we generated k-sparse signal vectors $\boldsymbol{x}$ of dimension $n = 961$, for each k in $\{5, 8, 10, 12, 15, 17, 20\}$. We choose a wide range of k in order to demonstrate that not only do our algorithms have high sensitivity and specificity for large values of k, they also keep performing reasonably, well beyond the typical operating regime. The support of each signal vector $\boldsymbol{x}$ – given k – was chosen by sampling a k-sparse binary vector uniformly at random from the set of all k-sparse binary vectors. The magnitudes of the non-zero elements of $\boldsymbol{x}$ were picked uniformly at random from the range [1, 32 768]. This high dynamic range in the value of $\boldsymbol{x}$ was chosen to reflect a variance in the typical threshold cycle values (C_t) of real PCR experiments, which can be between 16 and 32. From Eqn. (6), we can infer that viral loads vary roughly as 2^{-C_t} (setting $q = 1$), up to constant multiplicative terms. In all cases, $m = 93$ noisy measurements in $\boldsymbol{y}$ were simulated following the noise model in Eqn. (3) with $\sigma = 0.1$ and $q = 0.95$. A 93×961 Kirkman sensing matrix was used for generating the measurements. The Poisson nature of the elements of $\boldsymbol{x}$ in Eqn. (3) was ignored. This approximation was based on the principle that if $X \sim \text{Poisson}(\lambda)$, then $\text{Std. Dev.}(X)/E(X) = \sqrt{\lambda}/\lambda = 1/\sqrt{\lambda}$ which becomes smaller and smaller as λ increases. The recovery algorithms were tested on $Q = 1000$ randomly generated signals for each value of k.

3) ALGORITHMS TESTED

The following algorithms were compared:
1) COMP (see Table 1)
2) COMP followed by NNLASSO (see Table 2)
3) COMP followed by SBL (see Table 3)
4) COMP followed by NNOMP (see Table 4)
5) COMP followed by NNLAD (see Table V)
6) COMP followed by NNLS (see Table S.VI in the Supplementary)

For each algorithm any positives missed during the CS stage but caught by DD were declared as positives, as mentioned in Section III-D. For small sample sizes we also tested

TABLE 2. Performance of Comp Followed by Nnlasso (On Synthetic Data) for 93 × 961 Kirkman Triple Matrix. For Each Criterion and Each *k* Value, Mean and Standard Deviation Values are Reported, Across 1000 Signals

k	RMSE	#FN	#FP	Sens.	Spec.
5	0.047 ± 0.020	0.0 ± 0.1	0.8 ± 0.9	0.9990 ± 0.0141	0.9991 ± 0.0009
8	0.069 ± 0.028	0.1 ± 0.2	4.0 ± 2.1	0.9925 ± 0.0307	0.9958 ± 0.0022
10	0.100 ± 0.049	0.2 ± 0.5	7.9 ± 3.4	0.9780 ± 0.0454	0.9917 ± 0.0035
12	0.149 ± 0.092	0.6 ± 0.7	12.9 ± 5.0	0.9538 ± 0.0591	0.9864 ± 0.0052
15	0.295 ± 0.166	1.0 ± 1.1	28.5 ± 16.5	0.9316 ± 0.0722	0.9699 ± 0.0175
17	0.404 ± 0.184	1.1 ± 1.4	46.3 ± 23.5	0.9355 ± 0.0817	0.9509 ± 0.0249
20	0.563 ± 0.172	1.1 ± 1.8	78.4 ± 26.7	0.9452 ± 0.0923	0.9167 ± 0.0284

TABLE 3. Performance of Comp Followed by Sbl (On Synthetic Data) for 93 × 961 Kirkman Triple Matrix. For Each Criterion and Each *k* Value, Mean and Standard Deviation Values are Reported, Across 1000 Signals

k	RMSE	#FN	#FP	Sens.	Spec.
5	0.043 ± 0.017	0.0 ± 0.0	0.9 ± 0.9	0.9998 ± 0.0063	0.9991 ± 0.0010
8	0.058 ± 0.021	0.0 ± 0.2	4.3 ± 2.1	0.9958 ± 0.0227	0.9955 ± 0.0023
10	0.071 ± 0.025	0.1 ± 0.2	8.2 ± 3.1	0.9937 ± 0.0247	0.9913 ± 0.0033
12	0.094 ± 0.035	0.1 ± 0.4	13.6 ± 4.4	0.9886 ± 0.0310	0.9856 ± 0.0046
15	0.123 ± 0.108	0.3 ± 0.6	25.1 ± 6.9	0.9804 ± 0.0396	0.9735 ± 0.0073
17	0.165 ± 0.179	0.5 ± 0.8	35.1 ± 9.9	0.9713 ± 0.0491	0.9628 ± 0.0105
20	0.318 ± 0.305	1.3 ± 1.6	54.5 ± 13.2	0.9349 ± 0.0803	0.9420 ± 0.0140

TABLE 4. Performance of Comp Followed by Nnomp (On Synthetic Data) for 93 × 961 Kirkman Triple Matrix. For Each Criterion and Each *k* Value, Mean and Standard Deviation Values are Reported, Across 1000 Signals

k	RMSE	#FN	#FP	Sens.	Spec.
5	0.043 ± 0.019	0.0 ± 0.1	0.3 ± 0.6	0.9982 ± 0.0209	0.9997 ± 0.0006
8	0.060 ± 0.025	0.1 ± 0.4	1.8 ± 2.0	0.9831 ± 0.0472	0.9981 ± 0.0021
10	0.077 ± 0.035	0.3 ± 0.5	3.7 ± 3.2	0.9739 ± 0.0541	0.9961 ± 0.0034
12	0.115 ± 0.067	0.5 ± 0.7	7.8 ± 4.9	0.9565 ± 0.0560	0.9918 ± 0.0051
15	0.242 ± 0.190	1.5 ± 1.4	15.6 ± 6.0	0.9013 ± 0.0951	0.9835 ± 0.0064
17	0.361 ± 0.243	2.8 ± 2.2	20.8 ± 5.6	0.8329 ± 0.1268	0.9780 ± 0.0059
20	0.589 ± 0.282	6.1 ± 3.0	27.0 ± 5.2	0.6941 ± 0.1520	0.9713 ± 0.0055

TABLE 5. Performance of Comp Followed by Nnlad (On Synthetic Data) for 93 × 961 Kirkman Triple Matrix. For Each Criterion and Each *k* Value, Mean and Standard Deviation Values are Reported, Across 1000 Signals

k	RMSE	#FN	#FP	Sens.	Spec.
5	0.050 ± 0.021	0.0 ± 0.0	1.0 ± 1.0	0.9996 ± 0.0089	0.9990 ± 0.0010
8	0.077 ± 0.034	0.0 ± 0.2	4.9 ± 2.3	0.9939 ± 0.0270	0.9949 ± 0.0024
10	0.107 ± 0.050	0.2 ± 0.4	9.3 ± 3.0	0.9809 ± 0.0441	0.9903 ± 0.0032
12	0.167 ± 0.095	0.5 ± 0.7	14.4 ± 5.1	0.9574 ± 0.0586	0.9848 ± 0.0054
15	0.296 ± 0.160	0.9 ± 1.0	29.7 ± 16.7	0.9393 ± 0.0687	0.9686 ± 0.0176
17	0.401 ± 0.171	0.8 ± 1.2	50.2 ± 23.9	0.9549 ± 0.0717	0.9468 ± 0.0253
20	0.530 ± 0.166	0.2 ± 0.9	87.9 ± 25.1	0.9884 ± 0.0427	0.9066 ± 0.0267

COMP-BF, i.e. COMP followed by brute-force search for samples in x with non-zero values. Details of this algorithm and experimental results with it are presented in the supplemental Section S.IV.

4) COMPARISON CRITERIA

In the following, $\hat{x}$ denotes the estimate of x. Most numerical algorithms do not produce vectors that are exactly sparse and have many entries with very tiny magnitude, due to issues such as choice of convergence criterion. Since in this application, support recovery is of paramount importance to identify which samples in x were infected, we employed the following post-processing step: All entries in $\hat{x}$ whose magnitude fell below a threshold $\tau := 0.2 \times x_{min}$ were set to zero, yielding a vector $\bar{x}$. Here x_{min} refers to the least possible value of the viral load, and this can be obtained offline from practical experiments on individual samples. In these synthetic experiments, we simply set $x_{min} := 1$. We observed that varying the value of τ over a fairly wide range had negligible impact on the results, as can be observed from Section S.XII of the supplemental material. For SBL, we set τ to 0 and also set also negative entries in the estimate to 0. For NNOMP, such thresholding was inherently not needed. The various algorithms were compared with respect to the following criteria:

1) RMSE := $\|x - \bar{x}\|_2 / \|x\|_2$
2) Number of false positives (FP) := $|\{i : x_i = 0, \hat{x}_i > 0\}|$
3) Number of false negatives (FN) := $|\{i : x_i > 0, \hat{x}_i = 0\}|$
4) Sensitivity (also called Recall or True Positive rate) := #correctly detected positives/#actual positives

5) Specificity (also called True Negative Rate) :=
#correctly detected negatives/#actual negatives.

5) MAIN RESULTS

It should be noted that all algorithms were evaluated on 1000 randomly generated sparse signals, given the same sensing matrix. The average value as well as standard deviation of all quality measures (over the 1000 signals) are reported in the Tables I, II, III, IV, V, S.VI. A comparison of Table 1 to Tables II, III, IV, V, S.VI indicates that COMP followed by NNLASSO/SBL/NNOMP/NNLAD/NNLS significantly reduces the false positives at the cost of a rare false negative. The RMSE is also significantly improved, since COMP does not estimate viral loads. At the same time, COMP significantly reduces the size of the problem for the CS stage. For example, for the 93×961 Kirkman matrix, when number of infected samples k is 12, the average size of the matrix after COMP filtering is $\sim 30 \times 37$. From Table 1 we see that Definite Defectives classifies many positives as high-confidence positives, for k upto 8. We note that the experimental results reported in these tables are quite encouraging, since these experiments are challenging due to small m and fairly large k, n, albeit with testing on synthetic data. We noticed that running the CS algorithms without the COMP step did not perform as well, results for which are presented in the supplemental Section S.IX. We observed that the advantages of our combined group testing and compressed sensing approach holds regardless of the sensing matrix size. For comparison, results of running our algorithms using a 45×105 Kirkman matrix instead of the 93×961 Kirkman matrix are presented in supplemental Section S.VIII.

6) PARAMETER SELECTION

As mention earlier, the regularization parameters in various estimators such as COMP-NNLASSO, COMP-NNLAD, COMP-NNOMP, etc. are estimated via cross-validation. For these estimators, we therefore do not require knowledge of the σ parameter in the noise model from Eqn. (3). The q parameter in the noise model is set to 0.95 in all our experiments. It is a reasonable choice as the molecule count is known to roughly double in each cycle of RT-PCR [14]. Moreover, variation of q in the range from 0.7 to 1 showed negligible variation in the results of our wet-lab experiments as can be seen in Section S.XI and Table S.VIII of the supplemental material. Also note that we only report viral loads relative to $y_{\min}$ (see Eqn. (8)) - we do not attempt to estimate $y_{\min}$. These relative viral loads are interpretable by the RT-PCR technicians since they know $t_{\min}$, the minimum C_t (threshold cycle) value observed in that experiment. Note as well that since $y_{\min}$ is the viral load of the pool with the minimum C_t value – it corresponds to the pool with the maximum viral load in that experiment.

7) COMPARISON WITH DORFMAN POOLING

We also performed a comparison of our algorithms with the popular two-stage Dorfman pooling method (an adaptive

TABLE 6. Expected Number of Tests Needed by Optimal Dorfman Testing for Number of Samples (*n*) 105 and 961 for Various *k*. Note That Our Proposed Methods Based on CS Require Much Fewer Tests (45 and 93) Typically, and Do Not Require Two Rounds of Testing

$N = 105$			$N = 961$		
k	# Tests	Pool Size	k	# Tests	Pool Size
5	43.7	5	5	136.5	14
8	55.3	4	8	172.2	11
10	61.3	4	10	192.2	11
12	67.0	4	12	209.6	9
15	73.9	3	15	233.7	9
17	78.2	3	17	248.7	8
20	84.3	3	20	269.4	7

method), with regard to the number of tests required. In the first stage of the Dorfman pooling technique, the n samples are divided into n/g pools, each of size g. Each of these n/g pools are tested, and a negative result leads to all members of that pool being considered negative (i.e. non-infected). However, the pools that are tested positive are passed onto a second stage, where all members of those pools are individually tested. The optimal pool size g^* will minimize the expected number of tests taken by this process (given that the membership in each pool is decided randomly). A formula for the expected number of tests taken by Dorfman testing is derived in [2]. The derivation in [2] assumes the following: (1) Any given sample may be positive with probability p, independently of the other samples; (2) The number of samples n is divisible by the pool size g. We modify the formula from [2] for the case that n is not divisible by g (supplemental section S.XIII), and find g^* by choosing the value of g which minimizes this number. We set $p = k/n$, so that out of n samples, the number of infected samples is k in expectation. Table VI shows the expected number of tests computed from the formula in supplemental section S.XIII, assuming that the expected number of infected samples k (and thus the optimal pool size g^*) is known in advance. We also empirically verified the expected number of tests by performing 1000 Monte Carlo simulations of Dorfman testing with the optimal pool size g^* for each case, and did not observe much deviation from the numbers reported in Table VI. Comparisons of Tables I, II, III, IV with the two-stage Dorfman pooling method in VI show that our methods require much fewer tests, albeit with a slight increase in number of false negatives. Moreover, all our methods are single-stage methods and therefore require less time for testing, unlike the Dorfman method which requires two stages of testing.

8) ESTIMATION OF NUMBER OF INFECTED SAMPLES

The number of CS measurements for successful recovery depends on the number of non-zero elements (ℓ_0 norm) of the underlying signal. For example, this varies as $O(k \log n)$ for randomized sensing matrices [5] or as $O(\max(k^2, \sqrt{n})$ for deterministic designs [22]. There is a lower bound of $k\sqrt{n}$ measurements for certain types of expander matrices to satisfy a sufficient (but not necessary) condition for recovery [23]. However, in practice k is always unknown, which leads to

TABLE 7. True Sparsity *k* Versus Estimated Sparsity *k_est* (On Synthetic Data) for 93 × 961 Kirkman Matrix. Mean and Standard Deviation of Estimated Sparsity is Computed Over 1000 Signals for Each *k*

k	k_{est}
5	5.01 ± 0.33
10	10.07 ± 0.69
15	15.13 ± 1.17
20	20.26 ± 1.63
25	25.54 ± 2.03
30	30.53 ± 2.61

TABLE 8. Comparison of Mean Number of False Negative and False Positives for COMP, COMP-SBL and COMP-SBL With Graceful Failure Mode for High Values of *k* for the 93 × 961 Kirkman Matrix. The Algorithm Goes Into Graceful Failure Mode When Estimated Sparsity is Greater Than or Equal to 20

	COMP		COMP-SBL		COMP-SBL-graceful	
k	#FN	#FP	#FN	#FP	#FN	#FP
15	0	45.3	0.3	24.9	0.3	24.8
20	0	92.7	1.3	55.4	0.4	80.2
25	0	151.2	4	97.5	0	151.2
30	0	212.1	6.9	140.6	0	212.1

the question as to how many measurements are needed as a minimum for a *particular* problem instance. To address this, we adopt the technique from [48] to estimate k on the fly from the compressive measurements. This technique does not require signal recovery for estimating k. The relative error in the estimate of k is shown to be $O(\sqrt{\log m/m})$ [49], which diminishes as m increases (irrespective of the true k). Table VII shows the accuracy of our sparsity estimate on synthetic data.

The advantage of this estimate of k is that it can drive the COMP-BF algorithm, as well as act as an indicator of whether there exist any false negatives. We can use this knowledge to enable a *graceful failure mode*. In this mode, if our estimate of k is larger than what the CS algorithms can handle, we return only the output of the COMP stage. Hence in such rare cases, it minimizes the number of false negatives, at the cost of many false positives. In these cases a second stage of individual testing must be done on the samples which were declared positive. Table VIII shows the effect of using graceful failure mode with COMP followed by SBL for large values of k. In these experiments, output of COMP is returned if the estimated sparsity, k_{est}, is greater than or equal to 20. We see that COMP-SBL with graceful failure mode matches the behaviour of COMP-SBL at sparsity value lower than 20, and that of COMP at sparsity value greater than 20. At sparsity equal to 20, it compromises between the high false positives of COMP, and the high false negatives of COMP-SBL. This is because of the variability in k_{est}, which can occasionally be less than 20 even if k is equal to 20.

B. RESULTS ON REAL DATA

We acquired real data in the form of test results on pooled samples from two labs: one at the National Center of Biological Sciences (NCBS) in India, and the other at the Wyss Institute at the Harvard Medical School, USA. In both cases, viral RNA was artificially injected into k of the n samples

where $k \ll n$. From these n samples, a total of m mixtures were created. For the datasets obtained from NCBS that we experimented with, we had $m = 16$, $n = 40$, $k \in \{1, 2, 3, 4\}$. For the data from the Wyss Institute, we had $m = 24$, $n = 60$, $k = 2$ and $m = 30$, $n = 120$, $k = 2$. The results for all these datasets are presented in Table IX. The 16×40 and 24×60 pooling matrices were obtained by performing a simulated annealing procedure to minimize the mutual coherence (see Section III-F9), starting with a random sparse binary matrix as initial condition. The 30×120 pooling matrix was a Kirkman matrix. We used $q = 0.95$ in all cases to obtain relative viral loads from C_t values, using Eqn. (8). While q may be estimated from raw RT-PCR data (Section S.XI, supplemental material), we found $q = 0.95$ to be a reasonable choice, and did not observe any variation in the number of reported positives when this parameter was changed between 0.7 to 1. For NNLASSO, NNLS and NNLAD, we use $\tau = 0.2 \times \tilde{y}_{\max}$ as the threshold below which an estimated relative viral load is set to 0, since value of $x_{\min}$ may not always be available for real experiments. Here $\tilde{y}_{\max}$ is the relative viral load of the pool with the largest C_t value, and consequently the smallest viral amount. We see that the CS algorithms reduce the false positives, albeit with an introduction of occasional false negatives for higher values of k. We also refer the reader to our work in [6] for a more in-depth description of results on real experimental data.

C. DISCUSSION

Each algorithm we ran presented a different set of tradeoffs between sensitivity and specificity. While COMP provides us with sensitivity equal to 1, it suffers many false positives, especially for higher k. For other algorithms, in general both the sensitivity and the specificity decrease as k is increased. COMP-NNOMP (Table 4) has the highest specificity, but it comes at the cost of sensitivity. COMP-SBL (Table 3) has the best sensitivity for most values of k amongst the CS algorithms. COMP-NNLASSO (Table 2) has better specificity than COMP-SBL for small values of k, but loses out for $k \geq 15$. COMP-NNLAD and COMP-NNLS (Tables 5 and S.VI) start behaving like COMP for higher values of k, effectively bounding the number of false negatives. However, their number of false positives is almost as much as those with COMP.

Ideally, we want both high sensitivity and high specificity while catching a large number of infected samples. Hence, we look at k^*, which is the maximum number of infected samples k for which the sensitivity and specificity of the algorithm are greater than or equal to some threshold values. For the 45×105 Kirkman matrix, we chose the sensitivity threshold as 0.99 and the specificity threshold as 0.95. For the 93×961 Kirkman matrix, we chose both thresholds to be 0.99, since a specificity threshold of 0.95 gives too many false positives for 961 samples. We observed that COMP-SBL has $k^* = 10$ for both matrices, which is the highest amongst all algorithms tested. Typically we do not know the number of infections, but a prevalence rate of infection. The number of infected samples out of a given set of n samples may be treated as a

TABLE 9. Results of Lab Experiments With Each Algorithm

Dataset	Algorithm	# true pos	# false neg	#false pos
Harvard $24 \times 60, k = 2$	COMP	2	0	1
	COMP-SBL	2	0	1
	COMP-NNOMP	2	0	0
	COMP-NNLASSO	2	0	1
	COMP-NNLAD	2	0	1
	COMP-NNLS	2	0	1
Harvard $30 \times 120, k = 2$	COMP	2	0	1
	COMP-SBL	2	0	1
	COMP-NNOMP	2	0	1
	COMP-NNLASSO	2	0	1
	COMP-NNLAD	2	0	1
	COMP-NNLS	2	0	1
NCBS-0 $16 \times 40, k = 0$	COMP	0	0	0
	COMP-SBL	0	0	0
	COMP-NNOMP	0	0	0
	COMP-NNLASSO	0	0	0
	COMP-NNLAD	0	0	0
	COMP-NNLS	0	0	0
NCBS-1 $16 \times 40, k = 1$	COMP	1	0	0
	COMP-SBL	1	0	0
	COMP-NNOMP	1	0	0
	COMP-NNLASSO	1	0	0
	COMP-NNLAD	1	0	0
	COMP-NNLS	1	0	0
NCBS-2 $16 \times 40, k = 2$	COMP	2	0	0
	COMP-SBL	2	0	0
	COMP-NNOMP	2	0	0
	COMP-NNLASSO	2	0	0
	COMP-NNLAD	2	0	0
	COMP-NNLS	2	0	0
NCBS-3 $16 \times 40, k = 3$	COMP	3	0	1
	COMP-SBL	2	1	1
	COMP-NNOMP	2	1	0
	COMP-NNLASSO	2	1	1
	COMP-NNLAD	3	0	1
	COMP-NNLS	2	1	1
	COMP-BF	2	1	1
NCBS-4 $16 \times 40, k = 4$	COMP	4	0	3
	COMP-SBL	3	1	2
	COMP-NNOMP	2	2	2
	COMP-NNLASSO	3	1	2
	COMP-NNLAD	2	2	2
	COMP-NNLS	3	1	2
	COMP-BF	2	2	2

Binomial random variable with probability of success equal to the prevalence rate. Under this assumption, using COMP-SBL with the 93×961 Kirkman matrix, we observed that the maximum prevalence rate for which sensitivity and specificity are both above 0.99 is 1%. Similarly, using COMP-SBL with the 45×105 Kirkman matrix, we observed that the maximum prevalence rate for which sensitivity is above 0.99 and specificity is above 0.95 is 9.5%. Thus, Tapestry is viable at prevalence rates as high as 9.5%, while reducing testing cost by a factor of 2.3. On the other hand, if the prevalence rate is only 1% or less, it can reduce testing cost by a factor of 10.3.

Comments about sensitivity and specificity: We observe that the sensitivity and specificity of our method on synthetic data is *within the recommendations of the U.S. Food and Drugs Administration (FDA)*, as provided in this document [50]. The document provides recommendations for percent positive agreement (PPA) and percent negative agreement (PNA) of a COVID-19 test with a gold standard test (such as RT-PCR done on individual samples). PPA and PNA are used instead of sensitivity and specificity when ground-truth positives are not known. Since for synthetic data we know the ground truth positives, we compare their PPA and PNA recommendations with the sensitivity and specificity observed by us. We use COMP-SBL for comparison, since we consider it to be our best method.

For 'Testing patients suspected of COVID-19 by their healthcare provider' (point G.4.a, page 7 of [50]), the document considers positive and negative agreement of $\geq 95\%$ as acceptable clinical performance (page 9, row 2 of table

in [50]). The sensitivity and specificity of our method on the 93×961 Kirkman matrix is within this range for $k \leq 17$ infected samples (Table 3). For the 45×105 matrix, it is within this range for $k \leq 10$ infected samples (Table S.XII).

For 'Screening individuals without symptoms or other reasons to suspect COVID-19 with a previously unauthorized test' (point G.4.c, page 10 of [50]), the document considers positive agreement of $\geq 95\%$ and negative agreement of $\geq 98\%$ as acceptable (along with the lower bounds of two-sided 95% confidence interval to be $> 76\%$ and $> 95\%$ respectively). Similarly, for 'Adding population screening of individuals without symptoms or other reasons to suspect COVID-19 to an authorized test' (point G.4.d, page 12 of [50]) the document has the same criterion as for point G.4.c. Our sensitivity and specificity are within the ranges specified for the 93×961 Kirkman matrix for $k \leq 12$ (Table 3). While we do not report confidence intervals (as suggested for point G.4.c and G.4.d of [50]), the standard deviation of sensitivity and specificity reported by us are fairly low, and we believe the performance of our method is within the recommendations of [50]. Since our numbers are on synthetic data - these numbers may vary upon full clinical validation, especially considering that there may be more sources of error in a real test. Nonetheless, we find these numbers to be encouraging.

Further, we note that while our method incurs an occasional false negative, the viral loads of these false negative values are fairly small. This means that super-spreaders (who are believed to have high viral load [10]) will almost always be caught by our method. In the supplemental material, we discuss this in more detail in Sec. S.X, and provide a table of mean and standard deviations of viral loads of false negatives (Table S.VII) for all our methods on synthetic data.

Tapestry can detect certain errors caused by incorrect pipetting, pool contamination, or failed RT-PCR amplification of some pools. This is done by performing a consistency check after the COMP stage. If there is a pool which is positive, but all of the samples involved in it have been declared negative by COMP, this is indicative of error. In case of error, we list all samples categorized by the number of tests that they are positive in. However, the COMP consistency check will not catch all errors. Alternately, the noisy COMP [12] algorithm may be used to correct for errors in the COMP stage. A full exposition on detection and correction of errors is left as future work.

Although Tapestry can work with a variety of sensing matrix designs, we found Kirkman matrices to be most suitable for our purposes. This is due to lower sparsity and smaller pool sizes presented by Kirkman matrices. Our algorithms also exhibit a more stable behaviour over a wide range of the number of infected samples k when using Kirkman matrices. We compare some alternative matrix designs in Section S.VI.

V. RELATION TO PREVIOUS WORK
We review some recent work which apply CS or combinatorial group testing for COVID-19 testing. The works in [51]–[53] adopt a nonadaptive CS based approach. The works in [54]–[56] use combinatorial group testing. Compared to these methods, our work is different in the following ways (also see [6]):

1) *Real/Synthetic data:* Our work as well as that in [52] have tested results on real data, while the rest present only numerical or theoretical results.

2) *Quantitative Noise model:* Our work uses the physically-derived noise model in Eqn. (3) (as opposed to only Gaussian noise). This noise model is not considered in [51]. The work in [53] considers unknown noise. Combinatorial group testing methods [54]–[56] do not make use of quantitative information. The work in [52] uses only binary test information, even though the decoding algorithm is based on CS.

3) *Algorithms:* The work in [51] adopts the BPDN technique (i.e P1 from Eqn. (11)) as well as the brute-force search method for reconstruction. The work in [52], [57] uses the LASSO, albeit with a ternary representation for the viral loads. The work in [53] uses NNLAD. We use the LASSO with a non-negative constraint, the brute-force method, NNLAD, as well as other techniques such as SBL and NNOMP, all in combination with COMP. The work in [51] assumes knowledge of the (Gaussian) noise variance for selection of ε in the estimator in Eqn. (11), whereas we use cross-validation for all our estimators. The technique in [52] uses a slightly different form of cross-validation for selection of the regularization parameter in LASSO. Amongst combinatorial algorithms, [56] uses COMP, while [54] and [55] use message passing.

4) *Sensing matrix design:* The work in [51] uses randomly generated expander graphs, whereas we use Kirkman matrices. The work in [52] uses randomly generated sparse Bernoulli matrices or Reed-Solomon codes, while [55] uses Low-Density Parity Check (LDPC) codes [58]. The work in [53] uses Euler square matrices [59], and the work in [56] uses the Shifted Transversal Design [60]. Both are deterministic disjunct matrices like Kirkman matrices. Each sample in our matrix participates in 3 pools as opposed to 5 pools as used in [55], 6 pools as used in [52] and [56], and 8 pools as used in [53], which is advantageous from the point of view of pipetting time.

5) *Sparsity estimation:* Our work uses an explicit sparsity estimator and does not rely on any assumption regarding the prevalence rate.

6) *Numerical comparisons:* We found that COMP-NNLAD works better than the NNLAD method used in [53] on our matrices (see Tables 5 and S.XIX). We also found that COMP-NNLASSO and COMP-SBL have better sensitivity and specificity than COMP-NNLAD (see Tables 2, 3, and V). The method in [52] can correctly identify up to 5/384 (1.3%) of samples with 48 tests, with an average number of false positives that was less than 2.75, and an average number of false negatives that was less than 0.33. On synthetic simulations with their

48×384 Reed-Solomon code based matrix (released by the authors) for a total of 100 $\boldsymbol{x}$ vectors with ℓ_0 norm of 5 using COMP-NNLASSO, we obtained 1.51 false positives and 0.02 false negatives on an average with a standard deviation of 1.439 and 0.14 respectively. Using COMP-SBL instead of COMP-NNLASSO with all other settings remaining the same, we obtained 1.4 false positives and 0.0 false negatives on an average with a standard deviation of 1.6 and 0.1 respectively. As such, a direct numerical comparison between our work and that in [52] is not possible, due to lack of available real data, however these numbers yield some indicator of performance.

7) *Number of Tests:* We use 93 tests for 961 samples while achieving more than 0.99 sensitivity and specificity for $k = 10$ infections using COMP-SBL. In a similar setting, [55] use 108 tests for $Q = 1000$ samples under prevalence rate 0.01 for exact 2-stage recovery. The work in [56] uses 186 tests for 961 samples under the same prevalence rate, albeit for sensitivity equal to 1 and very high specificity. Matrix sizes studied in other work are very different than ours. The work in [61] builds on top of our Tapestry scheme to reduce the number of tests, but it is a two-stage adaptive technique and hence will require much more testing time.

VI. CONCLUSION

We have presented a non-adaptive, single-round technique for prediction of infected samples as well as the viral loads, from an array of *n* samples, using a compressed sensing approach. We have empirically shown on synthetic data as well as on some real lab acquisitions that our technique can correctly predict the positive samples with a very small number of false positives and false negatives. Moreover, we have presented techniques for appropriate design of the mixing matrix. Our single-round testing technique can be deployed in many different scenarios such as the following:

1) Testing of 105 symptomatic individuals in 45 tests.
2) Testing of 195 asymptomatic individuals in 45 tests assuming a low rate of infection. A good use case for this is airport security personnel, delivery personnel, or hospital staff.
3) Testing of 399 individuals in 63 tests. This can be used to test students coming back to campuses, or police force, or asymptomatic people in housing blocks and localities currently under quarantine.
4) Testing of 961 people in 93 tests, assuming low infection rate. This might be suitable for airports and other places where samples can be collected and tested immediately, and it might be possible to obtain liquid handling robots.

Outputs: We have designed an Android app named Byom Smart Testing to make our Tapestry protocol easy to deploy in the future. The app can be accessed at [62]. We are also sharing our code and some amount of data at [63]. More information is also available at our website [64].

Future work: Future work will involve extensive testing on real COVID-19 data, and extensive implementation of a variety of algorithms for sensing matrix design as well as signal recovery, keeping in mind the accurate statistical noise model and accounting for occasional pipetting errors.

REFERENCES

[1] D. Benatia, R. Godefroy, and J. Lewis, "Estimating COVID-19 prevalence in the United States: A sample selection model approach," [Online]. Available: https://www.medrxiv.org/content/10.1101/2020.04.20.20072942v1

[2] R. Dorfman, "The detection of defective members of large populations," *Ann. Math. Statist.*, vol. 14, no. 4, pp. 436–440, 1943.

[3] "Israelis introduce method for accelerated Covid-19 testing," Accessed: Apr. 8, 2021. [Online]. Available: https://www.israel21c.org/israelis-introduce-method-for-accelerated-covid-19-testing/

[4] "Corona 'pool testing' increases worldwide capacities many times over," Accessed: Apr. 8, 2021. [Online]. Available: https://healthcare-in-europe.com/en/news/corona-pool-testing-increases-worldwide-capacities-many-times-over.html

[5] E. Candes and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[6] S. Ghosh *et al.*, "Tapestry: A single-round smart pooling technique for COVID-19 testing," 2020, *medRxiv*. [Online]. Available: https://www.medrxiv.org/content/early/2020/05/02/2020.04.23.20077727

[7] R. Kueng and P. Jung, "Robust nonnegative sparse recovery and the nullspace property of 0/1 measurements," *IEEE Trans. Inf. Theory*, vol. 64, no. 2, pp. 689–703, Feb. 2018.

[8] E. W. Weisstein, "Kirkman's schoolgirl problem," from MathWorld-A Wolfram Web Resource. Accessed: Apr. 8, 2021. [Online]. Available: https://mathworld.wolfram.com/KirkmansSchoolgirlProblem.html

[9] D. K. Ray-Chaudhuri and R. M. Wilson, "Solution of Kirkman's schoolgirl problem," in *Proc. Symp. Pure Math*, 1971, vol. 19, pp. 187–203.

[10] P. M. Beldomenico, "Do superspreaders generate new superspreaders? A hypothesis to explain the propagation pattern of COVID-19," *Int. J. Infect. Dis.*, vol. 96, pp. 461–463, 2020.

[11] Y. Liu *et al.*, "Viral dynamics in mild and severe cases of COVID-19," *Lancet Infect. Dis.*, vol. 20, no. 6, pp. 656–657, 2020.

[12] C. L. Chan *et al.*, "Non-adaptive probabilistic group testing with noisy measurements: Near-optimal bounds with efficient algorithms," in *Proc. 49th Annu. Allerton Conf. Commun., Control, Comput.*, 2011, pp. 1832–1839.

[13] N. Jawerth, "How is the COVID-19 virus detected using real time RT-PCR?," [Online]. Available: https://www.iaea.org/newscenter/news/how-is-the-covid-19-virus-detected-using-real-time-rt-pcr

[14] "Efficiency of real-time PCR," Accessed: Apr. 5, 2021. [Online]. Available: https://www.thermofisher.com/in/en/home/life-science/pcr/real-time-pcr/real-time-pcr-learning-center/real-time-pcr-basics/efficiency-real-time-pcr-qpcr.html

[15] M. Aldridge, L. Baldassini, and O. Johnson, "Group testing algorithms: Bounds and simulations," *IEEE Trans. Inf. Theory*, vol. 60, no. 6, pp. 3671–3687, Jun. 2014.

[16] A. Gilbert, M. Iwen, and M. Strauss, "Group testing and sparse signal recovery," in *Proc. Asilomar Conf. Signals, Syst. Comput.*, 2008, pp. 1059–1063.

[17] N. Zhao, D. O'Connor, A. Basarab, D. Ruan, and K. Sheng, "Motion compensated dynamic MRI reconstruction with local affine optical flow estimation," *IEEE Trans. Biomed. Eng.*, vol. 66, no. 11, pp. 3050–3059, Nov. 2019.

[18] Z. Zhang, T. Jung, S. Makeig, and B. D. Rao, "Compressed sensing for energy-efficient wireless telemonitoring of noninvasive fetal ECG via block sparse bayesian learning," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 2, pp. 300–309, Feb. 2013.

[19] Y. Liu, M. D. Vos, and S. V. Huffel, "Compressed sensing of multichannel EEG signals: The simultaneous cosparsity and low-rank optimization," *IEEE Trans. Biomed. Eng.*, vol. 62, no. 8, pp. 2055–2061, Aug. 2015.

[20] E. Candes, "The restricted isometry property and its implications for compressive sensing," *Comptes Rendus Mathematiques*, vol. 346, no. 9-10, pp. 589–592, 2008.

[21] R. Baraniuk *et al.*, "A simple proof of the restricted isometry property for random matrices," *Constructive Approximation*, vol. 28, pp. 253–263, 2008.

[22] R. DeVore, "Deterministic construction of compressed sensing matrices," *J. Complexity*, vol. 23, pp. 918–925, 2007.

[23] M. Lotfi and M. Vidyasagar, "Compressed sensing using binary matrices of nearly optimal dimensions," *IEEE Trans. Signal Process.*, vol. 68, pp. 3008–3021, 2020, doi: 10.1109/TSP.2020.2990154.

[24] M. Davenport *et al.*, "Introduction to compressed sensing," in *Compressed Sensing: Theory and Applications*, Y. Eldar and G. Kutyniok, Eds. Cambridge, U.K.: Cambridge Univ. Press, 2012, pp. 1–64.

[25] R. Berinde *et al.*, "Combining geometry and combinatorics: A unified approach to sparse signal recovery," in *Proc. 46th Annu. Allerton Conf. Commun., Control, Comput.*, 2008, pp. 798–805.

[26] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning With Sparsity: The LASSO and Generalizations*. Boca Raton, FL, USA: CRC Press, 2015.

[27] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with application to wavelet decomposition," in *Proc. Asilomar Conf. Signals, Syst. Comput.*, 1993, pp. 40–44.

[28] T. T. Cai and L. Wang, "Orthogonal matching pursuit for sparse signal recovery with noise," *IEEE Trans. Inf. Theory*, vol. 57, no. 7, pp. 4680–4688, Jul. 2011.

[29] M. Yaghoobi, D. Wu, and M. Davies, "Fast non-negative orthogonal matching pursuit," *IEEE Signal Process. Lett.*, vol. 22, no. 9, pp. 1229–1233, Sep. 2015.

[30] M. Tipping, "Sparse bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, 2001.

[31] D. Wipf and B. D. Rao, "Sparse bayesian learning for basis selection," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2153–2164, Aug. 2004.

[32] E. Crespo Marques *et al.*, "A review of sparse recovery algorithms," *IEEE Access*, vol. 7, pp. 1300–1322, 2019.

[33] A. Nalci, I. Fedorov, M. Al-Shoukairi, T. T. Liu, and B. D. Rao, "Rectified Gaussian scale mixtures and the sparse non-negative least squares problem," *IEEE Trans. Signal Process.*, vol. 66, no. 12, pp. 3124–3139, Jun. 2018.

[34] H. Petersen, B. Bah, and P. Jung, "Efficient noise-blind ℓ_1-regression of nonnegative compressible signals," 2020, *arXiv:2003.13092*.

[35] M. Lotfi and M. Vidyasagar, "A fast noniterative algorithm for compressive sensing using binary measurement matrices," *IEEE Trans. Signal Process.*, vol. 66, no. 15, pp. 4079–4089, Aug. 2018.

[36] M. Raginsky, S. Jafarpour, Z. T. Harmany, R. F. Marcia, R. M. Willett, and R. Calderbank, "Performance bounds for expander-based compressed sensing in poisson noise," *IEEE Trans. Sig. Process.*, vol. 59, no. 9, pp. 4139–4153, Sep. 2011.

[37] Wikipedia contributors, "Steiner triple systems," 2021. Accessed: Apr. 8, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Steiner_system#Steiner_triple_systems

[38] E. J. Pegg, "Social golfer problem," from MathWorld-A Wolfram Web Resource, created by E. W. Weisstein. Accessed: Apr. 8, 2021. [Online]. Available: https://mathworld.wolfram.com/SocialGolferProblem.html

[39] "Math games: Social golfer problem," [Online]. Available: http://www.mathpuzzle.com/MAA/54-Golf%20Tournaments/mathgames_08_14_07.html

[40] M. Triska, *Solution Methods for the Social Golfer Problem*. Citeseer, 2008.

[41] I. Dotú and P. Van Hentenryck, "Scheduling social golfers locally," in *Proc. Int. Conf. Integration Artif. Intell. (AI) Operations Res. (OR) Techn. Constraint Program.*, Springer, 2005, pp. 155–167.

[42] S. J. Johnson and S. R. Weller, "Construction of low-density parity-check codes from Kirkman triple systems," in *Proc. IEEE Global Telecommun. Conf.*, 2001, vol. 2, pp. 970–974.

[43] V. Vermeirssen *et al.*, "Matrix and steiner-triple-system smart pooling assays for high-performance transcription regulatory network mapping," *Nature Methods*, vol. 4, no. 8, pp. 659–664, 2007.

[44] Wikipedia contributors, "Steiner systems," 2021. Accessed: Apr. 8, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Steiner_system

[45] V. D. Tonchev, "Steiner systems for two-stage disjunctive testing," *J. Combinatorial Optim.*, vol. 15, no. 1, pp. 1–6, 2008.

[46] C. Studer and R. Baraniuk, "Stable restoration and separation of approximately sparse signals," *Appl. Comput. Harmon. Anal.*, vol. 37, no. 1, pp. 12–35, 2014.

[47] V. Abdoghasemi *et al.*, "On optimization of the measurement matrix for comprcsive sensing," in *Proc. 18th Eur. Signal Process. Conf.*, 2010, pp. 427–431.

[48] V. Bioglio, T. Bianchi, and E. Magli, "On the fly estimation of the sparsity degree in compressed sensing using sparse sensing matrices," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2015, pp. 3801–3805.

[49] C. Ravazzi *et al.*, "Sparsity estimation from compressive projections via sparse random matrices," *EURASIP J. Adv. Signal Process.*, vol. 56, 2018. [Online]. Avilable: https://doi.org/10.1186/s13634-018-0578-0

[50] "In vitro diagnostics euas, section: Templates for eua submissions/diagnostic templates (molecular and antigen), bullet: Molecular diagnostic template for laboratories," Accessed: Mar. 27, 2021. [Online]. Available: https://www.fda.gov/medical-devices/coronavirus-disease-2019-covid-19-emergency-use-authorizations-medical-devices/in-vitro-diagnostics-euas#covid19ivdTemplates

[51] J. Yi, R. Mudumbai, and W. Xu, "Low-cost and high-throughput testing of COVID-19 viruses and antibodies via compressed sensing: System concepts and computational experiments," 2020. [Online]. Available: https://arxiv.org/abs/2004.05759

[52] N. Shental *et al.*, "Efficient high-throughput SARS-CoV-2 testing to detect asymptomatic carriers," *Sci. Adv.*, 2020. [Online]. Available: https://advances.sciencemag.org/content/early/2020/08/20/sciadv.abc5961

[53] H. Petersen, B. Bah, and P. Jung, "Practical high-throughput, non-adaptive and noise-robust SARS-CoV-2 testing," 2020. [Online]. Available: https://arxiv.org/abs/2007.09171

[54] J. Zhu, K. Rivera, and D. Baron, "Noisy pooled PCR for virus testing," 2020. [Online]. Available: https://arxiv.org/abs/2004.02689

[55] J.-T. Seong, "Group testing-based robust algorithm for diagnosis of COVID-19," *Diagnostics*, vol. 10, no. 6, 2020, Art. no. 396, doi: 10.3390/diagnostics10060396

[56] M. Täufer, "Rapid, large-scale, and effective detection of COVID-19 via non-adaptive testing," *J. Theor. Biol.*, vol. 506, 2020, Art. no. 110450.

[57] H. Nida *et al.*, "Highly efficient de novo mutant identification in a sorghum bicolor tilling population using the comseq approach," *Plant J.*, vol. 86, no. 4, pp. 349–359, 2016.

[58] D. J. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[59] R. R. Naidu, P. Jampana, and C. S. Sastry, "Deterministic compressed sensing matrices: Construction via euler squares and applications," *IEEE Trans. Signal Process.*, vol. 64, no. 14, pp. 3566–3575, Jul. 2016.

[60] N. Thierry-Mieg, "A new pooling strategy for high-throughput screening: The shifted transversal design," *BMC Bioinf.*, vol. 7, no. 28, 2006. [Online]. Avilable: https://doi.org/10.1186/1471-2105-7-28

[61] A. Heidarzadeh and K. R. Narayanan, "Two-stage adaptive pooling with RT-QPCR for COVID-19 screening," 2020, *arXiv:2007.02695*.

[62] "Byom app," [Online]. Available: https://rebrand.ly/byom-app

[63] "Tapestry code," [Online]. Available: https://github.com/atoms-to-intelligence/tapestry

[64] "Tapestry website," [Online]. Available: https://www.tapestry-pooling.com/

[65] D. Du, F. K. Hwang, and F. Hwang, *Combinatorial Group Testing and Its Applications*. Singapore: World Scientific, 2000.

[66] J. Zhang *et al.*, "On the theoretical analysis of cross validation in compressive sensing," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2014, pp. 3370–3374.

[67] Y. Li and G. Raskutti, "Minimax optimal convex methods for poisson inverse problems under ℓ_q-ball sparsity," *IEEE Trans. Inf. Theory*, vol. 64, no. 8, pp. 5498-5512, Aug. 2018.