AI for Pooled Testing of COVID-19 Samples

Ajit Rajwade, Nir Shlezinger, Yonina C. Eldar

July 20, 2022

Abstract

The COVID-19 pandemic has adversely affected millions all over the world. Efficient and effective testing of individuals for COVID-19, via modalities such as reverse transcription polymerase chain reaction (RT-PCR) is a crucial factor in combating this menace. Given the widespread scarcity of testing resources including testing kits, reagents, skilled manpower and available time, pooled testing has been advocated as a method of speed-up. Pooling involves mixing together small portions of 'samples' of different individuals, followed by testing the pools instead of the individual samples. It has been observed that a much smaller number of pools, as compared to the number of samples, is sufficient to allow for accurate prediction of the health status of the constituent samples, under the common and reasonable assumption that only a small number of the samples were infected. Artificial Intelligence (AI) has emerged as a key tool in improving the prediction accuracy as well as efficiency of pooled testing. Such algorithmic tools are often studied within the frameworks of group testing and compressed sensing. In this chapter, we present algorithmic tools for pooled testing and recovery, giving a broad description of the use of AI for pooled testing in the context of COVID-19.

Keywords: pooled testing, group testing, compressed sensing, RT-PCR, sensing matrix design, sparse regression, Dorfman testing, adaptive and non-adaptive testing

1 Introduction

The Coronavirus disease 2019 (COVID-19) pandemic has already forced lockdowns all over the globe, and has claimed more than five million lives worldwide. In order to handle and contain pandemics, and particularly COVID-19, large portions of the population should be frequently tested [1]. One of the main difficulties in doing so stems from the limited testing resources and the lengthy duration required to identify the presence of an infection [2].

The main diagnosis tool for COVID-19 tests is based on Ribonucleic acid (RNA) extraction via qualitative reverse transcription polymerase chain reaction (RT-PCR). Although various technological alternatives have been proposed [3,4], the RT-PCR process remains the leading and most widely spread method for COVID-19 testing. The output of this procedure represents an estimate of the viral load in the tested sample [5]. A major bottleneck associated with this form of COVID-19 testing follows from the fact that each RT-PCR machine can simultaneously process a fixed number of samples, and its procedure tends to be quite lengthy, on the order of a few hours for each test.

A leading method to tackle the time-consuming nature of RT-PCR, whose duration often lasts between $3 \sim 4$ hours, and the limited capacity associated with the detection scheme, is to increase the efficiency of COVID-19 tests using *pooling* techniques [6,7]. Pooling builds on the fact that empirical observations suggest a low prevalence rate, namely, most tests are wasteful as most individual samples will return as negative. Consequently, for large populations, testing grouped samples can facilitate the aim of providing quick yet accurate results economically at an increased throughput. In pooling, each sample processed, i.e., each input to the RT-PCR machine, is comprised of a mixture of small portions of several samples taken from different patients (one sample per patient). When the infected patients constitute a small subset of the overall tested individuals, pooling-based schemes allow accurate recovery using a number of tests which is notably smaller than the number of patients [8]. While pooling can significantly increase the throughput of testing and thus

boost rapid and accessible diagnosis for tracking of the pandemic, the fact that it relies on the examination of mixed samples implies that dedicated artificial intelligence (AI) methods are required for the automated and rapid identification of the infected subjects and the recovery of their viral loads.

In this chapter we review recovery methods for pooled COVID-19 tests. These AI methods can be divided according to two main mathematical frameworks for such recovery procedures: group testing (GT) and compressed sensing (CS). To describe methods belonging to these families, we begin by presenting a mathematical model which formulates the problem of pooled testing recovery in Sec. 2. We elaborate on the subtleties and the specific assumptions and requirements which characterize COVID-19 testing. Based on the modelling of the pooled recovery setting, we review the application of GT, which was originally designed for testing infections in large populations, for COVID-19 pooled testing in Sec. 3. Our review examines different GT schemes along with their relevance in the context of COVID-19. We proceed to discuss recovery methods which arise from CS theory in Sec. 4. The framework of CS deals with the recovery of sparse signals [9, 10], and such methods were proposed for recovery from pooled COVID-19 tests in [11–16]. We conclude the chapter with a qualitative comparison between these families of AI methods in Sec. 6, and identify the setting in which one is likely to be preferable over the other for pooled COVID-19 tests.

2 System Model

In this section we present the system model for which we review pooled recovery algorithms in Sec. 3-4. As we focus on COVID-19 testing, we begin by identifying the specific characteristics that arise from this application of pooling in Sec. 2.3, based on which we formulate the recovery problem in Sec. 2.4.

2.1 The PCR Process

We begin with an overview of the RT-PCR process following [17]. We refer the reader to [18, 19] for a more detailed description.

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, following which it is mixed into a liquid medium. The RNA molecules of the virus (if present) in the original sample and thus also in the liquid medium, get transformed into *complementary DNA* (cDNA). This conversion from RNA to cDNA occurs by a process termed *reverse transcription*. To this mixture, DNA fragments called *primers* are added. These primers behave in a manner that is 'complementary' to the existing cDNA and get attached to certain specific sections of the cDNA (if the virus is present in the sample). This cDNA then undergoes a process of exponential amplification inside the RT-PCR machine during several cycles of alternate heating and cooling. Roughly per cycle, the cDNA is doubled in number. Fluorescent markers added to this mixture produce a visible fluorescence effect, observed almost immediately on a computer screen, in response to, and directly proportional to, the total amount of amplified cDNA. The time when the amount of fluorescence is observed to exceed a machine-specific threshold is known as the *cycle threshold* (CT). A smaller CT value indicates greater number of copies of the virus, and a larger CT value indicates a lower number (potentially even zero). Usually CT values lie anywhere between 16 to 35 cycles in real experiments. Statistics of typically observed CT values are reported in [20]. In particular, we note that RT-PCR can detect even single molecules. A single molecule typically would have a CT value of around 40 cycles. An RT-PCR machine can typically test 96 samples in parallel, and the testing time is between 3 to 4 hours.

2.2 Mathematical Model

The readings provided in RT-PCR testing provide an indication on the presence and level and infection. In some cases, one can approximate the relationship between the cycle time of a sample and its viral load using a statistical model [17, 19, 21]. Due to the fact that the growth of cDNA molecules is exponential [19] (as also explained in Sec. 2.1), the number of molecules of viral cDNA in a sample i at cycle time t is given by: $z_i(1+q_a)^t$, where z_i is the initial viral load in the sample (before the RT-PCR process began), and $q_a \in [0,1]$ is an amplification factor. Here t is a real number, and $\lfloor t \rfloor$ denotes the number of RT-PCR cycles that have passed. The fluorescence $\xi_i(t)$ of the i^{th} pool at time t is directly proportional to the number of

virus molecules. Hence we have:

$$\xi_i(t) = K \cdot z_i(1+q_a)^t, \tag{1}$$

where K is a constant of proportionality.

Let F be the threshold value of the fluorescence, and assume that the fluorescence of pool i reaches the value F at some cycle time τ_i, according to (1). Due to factors such as the stochasticity of the RT-PCR process reaction and the measurement error/quantization in the RT-PCR machine, the CT value that is output by the machine will not reflect this true cycle time. In other words, the true cycle time τ_i and the recorded cycle time t_i are statistically related, and this relationship can be modelled as $\tau_i = t_i + e_i$, where e_i is interpreted as the error in recording the value of τ_i. Here e_i is modeled as a zero-mean Gaussian random variable with variance σ_e^2. We assume $0 < \sigma_e^2 \ll 1$, as we expect this error to not be too high. Plugging in $\xi_i(\tau_i) = F$, we have from (1) that

$$F = K \cdot z_i(1+q_a)^{\tau_i} = K \cdot y_i(1+q_a)^{t_i}, \tag{2}$$

where y_i is defined to be the noisy initial viral load of sample i corresponding to a noisy observed CT value t_i. Hence,

$$y_i = z_i(1+q_a)^{\tau_i - t_i} = z_i(1+q_a)^{e_i}. \tag{3}$$

The above stochastic relationship constitutes an approximate model of the underlying statistics of RT-PCR measurements, which one may use for designing and analyzing pooling methods, as discussed in the sequel.

2.3 Pooled COVID-19 Tests

The common approach in testing for the presence of COVID-19 is based on the RT-PCR method. Here, a sample is collected, most commonly based on nasopharyngeal or oropharyngeal swabs or saliva [22]. The presence of infection is then examined by RNA extraction via RT-PCR measurements, quantifying the viral load in the sample. The RT-PCR process is quite time consuming (on the typical order of several hours), and can simultaneously examine up to a given number of m inputs (commonly on the order of several tens of samples). This results in a major bottleneck, particularly when the number of patients, denoted by n, is much larger than m.

A candidate approach to reduce the test duration utilizes pooling [6]. Pooling considers a small mixture of pooled samples rather than the total number of samples. To model this procedure in a generic manner, let $\boldsymbol{f} \in \mathcal{R}_+^n$ represent the samples vector, i.e., the viral loads of each of the subjects, whose entries $f_1, \ldots, f_n$ denote the viral loads of the n individuals. If $f_i = 0$, then the i^{th} individual is not infected. Pooled testing is comprised of two stages: pooling and measurement.

Pooling refers to the mapping of the n samples $\boldsymbol{f}$ into a lower-dimensional vector $\boldsymbol{z} \in \mathcal{R}_+^m$, with $m < n$. Here, each entry of $\boldsymbol{z}$ is obtained by mixing a subset of the samples in $\boldsymbol{f}$ determined by the binary valued pooling matrix $\boldsymbol{\Phi} \in \{0,1\}^{m \times n}$, such that

$$z_i = g_{\text{pool}}\left(\{f_j | \boldsymbol{\Phi}_{i,j} = 1\}\right), \quad i \in \{1, \ldots, m\}. \tag{4}$$

In (4), $g_{\text{pool}}(\cdot)$ represents how the viral loads of the mixed samples are mapped into the viral load of the pool. In direct (non-pooled) testing, one can write $z_i = f_i$ and $\boldsymbol{\Phi}$ as the $n \times n$ diagonal matrix. While pooling is commonly modelled as averaging, i.e., $g_{\text{pool}}(\mathcal{X}) = \frac{1}{|\mathcal{X}|}\sum_{x \in \mathcal{X}} x$, in many applications, including COVID-19 testing, the relationship between the pool and the mixed samples is more complex and includes stochasticity and dilution [6].

Measurement refers to reading of the pooled vector $\boldsymbol{z}$ into the observed vector $\boldsymbol{y} \in \mathcal{R}_+^m$. This is carried out via the element-wise mapping $g_{\text{meas}}(\cdot)$, such that $y_i = g_{\text{meas}}(z_i)$ for each $i \in \{1, \ldots, m\}$. For RT-PCR, the mapping $g_{\text{meas}}(\cdot)$ represents a procedure which involves the conversion of the sample into complementary DNA through reverse transcription, followed by examining its reaction in response to the heating and cooling cycle through a chemical process to quantify the number of virus particles (as earlier described in Sec. 2.1). The measurement procedure typically involves some level of uncertainty and distortion, and thus $g_{\text{meas}}(\cdot)$ is typically stochastic. Under some approximations detailed in Sec. 2.2, one can approximate $g_{\text{meas}}(\cdot)$ using the noise model in (3).

An illustration of the overall flow of pooled RT-PCR-based COVID-19 testing is depicted in Fig. 1. On the left side of the figure, we show the true viral loads of all n items. In particular, we see that the first

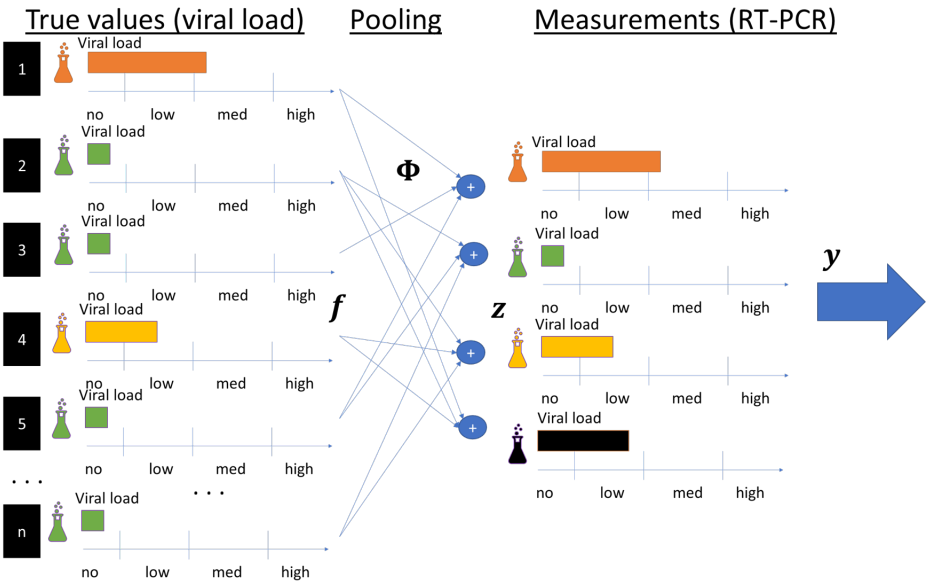

Figure 1: Pooled RT-PCR testing illustration.

item is infected in a medium level, the fourth item is infected in a low level, and all other items are not infected. Next, pooling is done based on a testing matrix, which is generated prior to obtaining the samples. For example, the first pooled test involves samples from the first, third, and fifth items. This results in a measurement vector, denoted by $\boldsymbol{z}$. This vector is fed to the recovery algorithm, which should be able to tell that the first item is infected (with some medium level viral load), the fourth item is infected (with some low level viral load), and all other items are not infected.

While the above description of the pooling and measurement procedure are generic, its application to RT-PCR-based COVID-19 tests gives rise to the following characteristics:

A1 The number of infected measurements, denoted by k, is much smaller than the number of tested individuals n. Typically $k \leq 0.1n$, i.e., up to 10% of the tested population is infected. This number often varies between non-symptomatic patients (where k is typically much smaller than 10%) and symptomatic ones.

A2 The RT-PCR measurements are noisy, i.e., some level of random distortion is induced in the overall process, encapsulating the randomness in the acquisition of the samples, their mixing, and the RT-PCR reading. As a result, the pooling and measurement procedures $g_{\text{pool}}(\cdot)$ and $g_{\text{meas}}(\cdot)$ may be noisy and unknown, with one possible model for $g_{\text{meas}}(\cdot)$ being the relationship formulated in Sec. 2.2.

A3 There is a limit, denoted by $L > 1$, on the number of subjects which can be mixed together in a single measurement. A typical limit on the number of subjects mixed together is $L = 32$ [6]. Furthermore, the portion taken from each sample for the pooled measurements is identical, e.g., one cannot mix 30% from one patient with 70% from another patient into a single pool.

While the characteristics are identified for swab-based RT-PCR tests, they also hold for other forms of contagious disease testing, including, e.g., serological tests [23].

2.4 Recovery from Pooled Tests

Pooling is based on mixing the samples of groups of patients together, forming m mixed samples out of the overall n patients. Then, the presence of COVID-19 for each of the tested individuals must be recovered from the mixed RT-PCR measurements, either directly, i.e., in a *one-shot* fashion, or in an adaptive manner, which involves additional tests [24]. When testing for COVID-19, the following requirements must also be accounted when designing the recovery procedure:

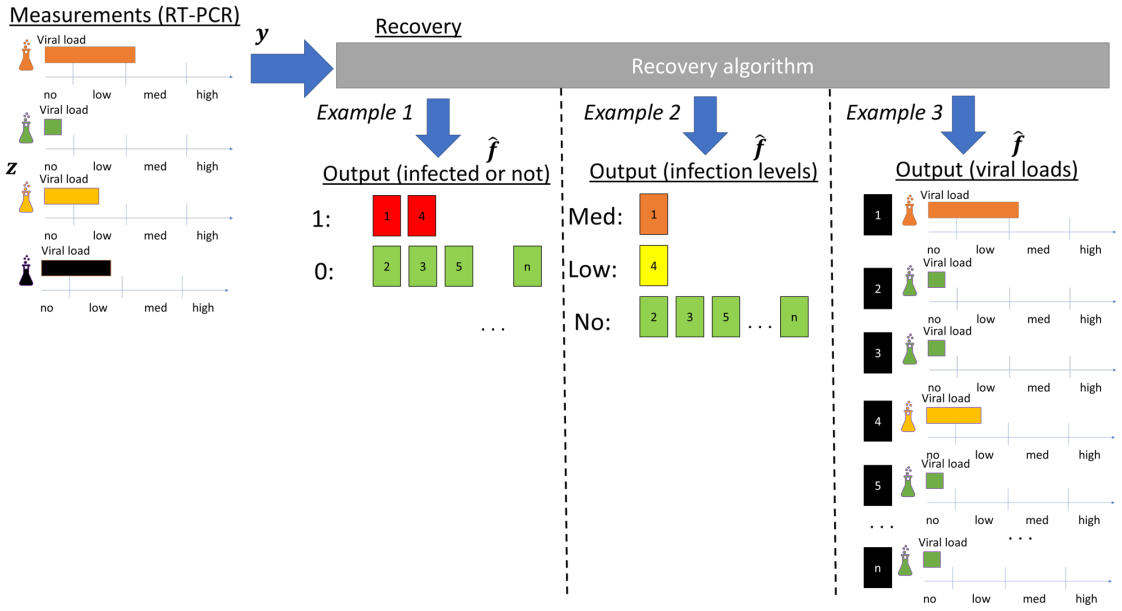

Figure 2: Pooled testing recovery illustration, divided into desired outputs following Examples 1-3.

R1 One-shot tests, where we fully identify all subjects from a single RT-PCR operation, are often preferable over adaptive techniques, which involve having to carry out additional tests based on the results. In particular, one-shot methods are simpler to automate, being less dependent on human intervention compared to adaptive schemes, and thus typically reduce the overall duration of the testing procedure (though not necessarily the number of pooled tests m).

R2 In some cases, one is interested not in only identifying whether a subject is infected or not, but also in some score on the viral load. This can be either the viral load itself, or a discrete score.

Per *R2* the objective of recovery from COVID-19 pooled tests is to recover a score on the infection of each subject. To formulate this, we define a mapping, $Q : \mathcal{R}_+ \mapsto \mathcal{Q}$, which translates a viral load into the corresponding score, while $\mathcal{Q}$ is a set describing all possible infection status. Some examples of such mappings of interest are:

Example 1. *When one is only interested in detecting whether a subject is infected or not, then* $\mathcal{Q} = \{0, 1\}$, *with* $Q(x) = 1$ *when* x *is larger than some infection threshold, and* $Q(x) = 0$ *otherwise.*

Example 2. *Often in practice, one is interested in a discrete score of the viral load, For example, possible discrete score outputs are* no *(no virus),* low *(borderline),* mid *(infected), and* high *(highly infected); this requirement implies that* $\mathcal{Q} = \{no, low, med, high\}$. *The decision regions are defined by the pre-specified thresholds* $\tau_1, \ldots, \tau_{|\mathcal{Q}|-1}$, *such that* $Q(x)$ *maps a value of* x *not larger than* τ_1 *is treated as not infected, while, e.g., a value in the range* $(\tau_1, \tau_2]$ *is treated as a low level of infection.*

Example 3. *When the desired output is the actual viral load, then* $\mathcal{Q} = \mathcal{R}_+$, *with* $Q(x) = x$.

Based on the characteristics of pooled COVID-19 tests detailed above, the recovery problem can be formulated as the estimation of $Q(\boldsymbol{f})$ (where $Q(\cdot)$ is applied element-wise) from $\boldsymbol{z}$, denoted by $\hat{\boldsymbol{f}} \in \mathcal{Q}^n$. In particular, for the subset of k infected items of a total of n inspected patients, the goal in pooled recovery is to formulate an algorithm for reconstructing $\hat{\boldsymbol{f}}$ from $\boldsymbol{y}$, possibly along with the design of the $m \times n$ measurement matrix $\boldsymbol{\Phi}$. The measurement matrix should guarantee that at most L subjects are mixed in each pool-test, such that the recovery algorithm can identify the subset of infected items and their representation of the viral load using the recovery algorithm, from the observed vector $\boldsymbol{y}$. The recovery procedure is visualized in Fig. 2.

AI-based recovery algorithms typically fall into either of the following mathematical frameworks: GT, which is concerned with detecting the presence of infection (e.g., Example 1), and CS, which originates from the recovery of sparse continuous-amplitude vectors, and is thus more naturally suitable for recovering the actual viral loads (e.g., Example 3). We discuss these approaches in Sec. 3-4, respectively. The recovery of a discrete yet non-binary score, as in Example 2, requires an extension of GT techniques, which we also present in Sec. 3.

3 Group Testing Methods for COVID-19

GT is a mathematical theory derived for detecting infections in large populations. It was introduced by Robert Dorfman in [25], who suggested GT for identifying syphilis-infected draftees during World War II. Since then, GT has been long studied and utilized in many fields, such as biology and chemistry [26, 27], communications [28–30], sensor networks [31], pattern matching [32], web services [33], and cyber security [34–36].

Broadly speaking GT is a framework for group detection problems of binary variables, i.e., each subject can either be infected or not infected [8]. GT is traditionally adaptive, requiring multiple sequential tests [25] in order to achieve a minimal number of overall tests from which the presence of infection can be inferred. Nonetheless, GT can also be applied in a one-shot (non-adaptive) manner [37], avoiding the need to mix new samples during the testing procedure, as studied for COVID-19 pooled testing in [38, 39]. Therefore, in the following we divide our description of GT methods into adaptive and non-adaptive. Unless stated otherwise, our description henceforth focuses on linear measurements, that is we consider $g_{\text{pool}}(\boldsymbol{f}) = \boldsymbol{\Phi}\boldsymbol{f}$ to be the common setting for GT studies.

3.1 Adaptive GT Methods

GT was originally proposed as an adaptive procedure. The simple scheme proposed by Dorfman partitions the patients into distinct sets of size B that are pooled together. Then, the subjects of each pool which is tested as positive are re-tested separately one-by-one, where the measurements are assumed to be noiseless, i.e., $g_{\text{meas}}(\boldsymbol{z}) = \boldsymbol{z}$. The resulting formulation is stated below as Algorithm 1.

Algorithm 1: Dorfman Adaptive GT [25]

Init: Subjects to be inspected $\boldsymbol{f} \in \mathcal{R}_+^n$, set size B

Pooling:

Set number of pools to $m = \lceil n/B \rceil$;

Set measurement matrix such that $(\boldsymbol{\Phi})_{i,j} = 1$ if $i \in [(j-1)\cdot B, j \cdot B)$ and $(\boldsymbol{\Phi})_{i,j} = 0$ otherwise;

Measurement:

Measure $\boldsymbol{y}$;

Recovery:

Set $\hat{\boldsymbol{f}} = 0$;

for *each* j *such that* $y_j \neq 0$ **do**

  Set each $\hat{f}_i$ for which $(\boldsymbol{\Phi})_{i,j} = 1$ by testing f_i individually;

end

Dorfman's method, which is simple to implement and highly intuitive, has been used for COVID-19 pooled testing in [6, 7, 40]. When one has prior knowledge of the number of defective items k, the setting of B which minimizes the maximal number of tests is $B = \sqrt{n/k}$ [41]. The adaptive nature of Algorithm 1 stems from the fact that it operates in two stages. Yet, adaptive GT can be more efficient in terms of the maximal number of tests when operating with multiple stages in a recursive manner. A representative implementation of recursive adaptive GT is the methods proposed in [42] which is based on binary splitting. Broadly speaking, binary splitting recursively partitions the subjects as long as they contain at least one infected item. The resulting procedure is formulated as Algorithm 2. Recursive adaptive GT based on binary splitting is known to be most beneficial when the k is much smaller compared with n, Furthermore, it was

conjectured in [43] that GT, and pooling in general, should only be applied when the subjects are indeed sparse (as per *A1*), and particularly when $k < n/3$.

Algorithm 2: Binary Splitting Adaptive GT [42]

Init: Subjects to be inspected $\boldsymbol{f} \in \mathcal{R}_+^n$, index of first subject i
Pooling:
Set measurement matrix $\boldsymbol{\Phi}$ to be the $1 \times n$ all-ones vector;
Measurement:
Measure (the scalar) y;
Recovery:
if $y \neq 0$ **then**
  if $n = 1$ **then**
    Set $\hat{f}_i$ as infected;
  end
  else
    Invoke Algorithm 2 for $f_1, \ldots, f_{\lceil n/2 \rceil}$ with first index i;
    Invoke Algorithm 2 for $f_{\lceil n/2 \rceil + 1}, \ldots, f_n$ with first index $i + \lceil n/2 \rceil$;
  end
end
else
  Set $\hat{f}_i, \ldots, \hat{f}_{i+n-1}$ as not infected;
end

The major drawback of adaptive GT for COVID-19 pooled testing stems from the fact that it requires additional procedures based on the results. This implies that laboratory technicians must save swabs taken from patients in addition to those mixed into the pools, and then possibly mix them again based on the outcome of the lengthy RT-PCR procedure. This can constitute a major bottleneck and notably burden laboratory technicians for both two-stage GT (as in Algorithm 1), and even more so for recursive GT (as in Algorithm 2).

3.2 Non-Adaptive GT Methods

Non-adaptive GT operates in a one-shot manner, detecting the presence of infections among all patients without requiring additional tests to be acquired based on previous results. Unlike the adaptive techniques detailed in the previous section, where the recovery algorithm typically dictates the measurement matrix $\boldsymbol{\Phi}$, non-adaptive methods can typically be carried out with different polling patterns. Therefore, in the following we first discuss how to set the pooling matrix $\boldsymbol{\Phi}$, after which we review some representative recovery methods designed for both noiseless and noisy settings.

3.2.1 Pooling Matrix

To determine the testing matrix $\boldsymbol{\Phi}$, one must first fix the number of the pool-tests m. GT theory dictates that m should satisfy $m = (1 + \epsilon)k \log_2 n$, for some $\epsilon > 0$, as this is the sufficient number of pool-test for which reliable recovery in noiseless GT can be guaranteed [44]. The parameter ϵ controls the probability of error of the procedure [44]. In practice, the number of pools-tests is often dictated by the measurement setup, e.g., it may be constrained to be an integer multiple of the number of inputs accepted by an RT-PCR machine.

The traditional GT method of generating $\boldsymbol{\Phi}$ for non-adaptive recovery draws its elements in an i.i.d. fashion according to a Bernoulli distribution with parameter p. A common choice for p is $p = 1 - 2^{-1/k}$, for which the probability of each element in $\boldsymbol{z}$ to be zero is $1/2$. When k is unknown, p is chosen using a rough approximation of k. A major drawback of this approach is that *A3* is not necessarily satisfied, and there is some chance that too many patients will be mixed into the same pool causing dilution. This can be relieved by forcing the columns of $\boldsymbol{\Phi}$, as well as the rows of $\boldsymbol{\Phi}$, to be typical, such that every column/row to have

exactly $\lceil p \cdot m \rceil \leq L$ and $\lceil p \cdot n \rceil$ ones, respectively [38]. In practical testing setups, one is interested in using a fixed deterministic matrix, rather than having to work with random matrices. Thus, $\boldsymbol{\Phi}$ should be generated once before the pooling starts, after which it can be used for multiple pooling experiments.

3.2.2 Noiseless Linear Non-Adaptive Recovery

Noiseless recovery methods are designed for linear noiseless measurements, such that $\boldsymbol{y} = \boldsymbol{z} = \boldsymbol{\Phi f}$. For such settings, common strategies attempt to classify the subjects into DND and DD based on observing which are mixed into a non-infected pool and which are mixed into an infected one. The identification of the DND is carried out using COMP. The purpose of COMP is to characterize DND subjects as those pooled into a non-infected pool, while classifying the remaining patients as possible defective (PD). The resulting procedure is summarized below as Algorithm 3.

Algorithm 3: Combinatorial Orthogonal Matching Pursuit [45, 46]

Init: Pooled measurements $\boldsymbol{y} \in \mathcal{R}_+^m$, pooling matrix $\boldsymbol{\Phi}$, infection threshold τ

Mark each $\{\hat{f}_j\}_{j=1}^n$ as PD;
for $i = 1, \ldots, m$ **do**
  if $y_i \leq \tau$ **then**
    Set $\hat{f}_j$ as DND ($Q(0)$) for each j such that $(\boldsymbol{\Phi})_{i,j} \neq 0$;
  end
end

While Algorithm 3 only divides the patients into DND and PD, it can also be extended to identify some of the subjects as DD. The rationale here is that whenever a pool is tested as infected, one of the subjects mixed into that specific pool must be infected. Therefore, if only one PD subject is mixed into an infected pool, then this subject is DD. The resulting steps are summarized as Algorithm 4.

Algorithm 4: Definitely Defective Non-Adaptive Recovery [47]

Init: Pooled measurements $\boldsymbol{y} \in \mathcal{R}_+^m$, pooling matrix $\boldsymbol{\Phi}$, infection threshold τ

Apply COMP (Algorithm 3) to obtain DND and PD subjects;
for *each PD subject* $\hat{f}_j$ **do**
  if $\exists y_i > \tau$ *such that* $(\boldsymbol{\Phi})_{i,j} \neq 0$ *while* $(\boldsymbol{\Phi})_{i,j'} = 0$ *for each* $j' \neq j$ *with PD* $\hat{f}_{j'}$ **then**
    Set $\hat{f}_j$ as DD;
  end
end

The DD procedure in Algorithm 4 can be repeated sequentially, i.e., iteratively using previously identified DD subjects to repeat the detection procedure. This iterative operation is referred to as Sequential COMP [47]. It is noted though that these algorithms may result in some of the patients being classified as neither DD nor DND, implying that a policy for dealing with such cases is required (though it is likely to be infrequently applied).

Algorithms 3-4 can be used as preliminary processing, after which the remaining PD patients are detected using alternative recovery methods. In particular, for settings where one is interested in recovering the viral loads and not just which patients are infected, COMP can be used at pre-processing to identify the DND subjects and reduce the number of patients whose viral load must be estimated; the viral loads of these remaining PD subjects can then be recovered using methods that are not limited to detection, e.g., loopy belief propagation (LBP) [39], CS [12], and even least-squares estimation [38].

3.2.3 Noisy Non-Linear Non-Adaptive Recovery

The non-adaptive GT recovery algorithms detailed in the previous section are designed for settings where the measurements are linear and noiseless, such that $\boldsymbol{y} = \boldsymbol{z} = \boldsymbol{\Phi f}$. Some of these techniques can be extended

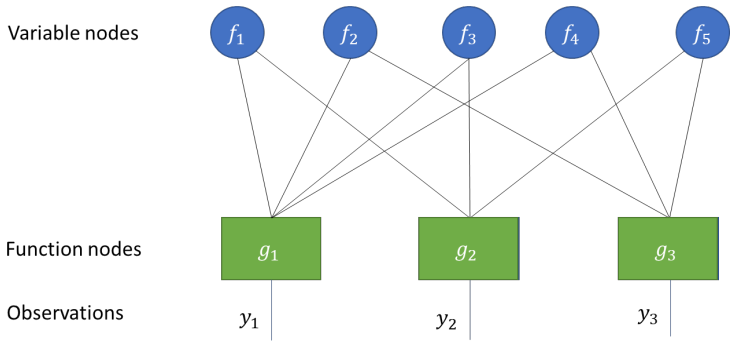

Figure 3: Representation of pooling procedure as a loopy factor graph.

to the presence of additive noise, i.e., when $\boldsymbol{y} = \boldsymbol{z} + \boldsymbol{w}$ where $\boldsymbol{w}$ is a noise signal, by increasing the number of pools, as in e.g., noisy COMP [37]. However, these methods do not naturally extend to setting where the relationship between the measurements and the pooled viral loads is not only noisy, but also non-linear (namely $\boldsymbol{z}$ is not equal to $\boldsymbol{\Phi f}$) as in (4), which is often the case in RT-PCR-based COVID tests.

When the pooling and measurement mapping are stochastic yet their distribution is known, the recovery rule which minimizes the error probability, and holds for both linear and non-linear measurements, is the maximum a-posteriori probability (MAP) rule

$$\hat{\boldsymbol{f}} = \arg\max_{\boldsymbol{s}\in\mathcal{Q}^n} P(\boldsymbol{s}|\boldsymbol{y}). \tag{5}$$

However, even when the underlying distribution is fully known, computing MAP estimates is a computationally prohibitive task whose burden grows exponentially with n. Nonetheless, its decision rule can be approached with complexity that only grows linearly with n via the LBP algorithm [48], [41, Ch. 3.3].

Consider our pooled testing setup, where the task is to estimate the discrete-valued vector $\boldsymbol{s} \in \mathcal{Q}^n$ with entries $s_i = Q(f_i)$ from the observed $\boldsymbol{y} \in \mathcal{R}_+^m$. Such settings correspond to Example 1 (where $|\mathcal{Q}| = 2$) and Example 2. Given a pooling pattern matrix $\boldsymbol{\Phi}$, the joint probability distribution of $\boldsymbol{y}, \boldsymbol{s}$ can be expressed as a partition

$$P(\boldsymbol{y}, \boldsymbol{q}) = P(\boldsymbol{y}|\boldsymbol{s})P(\boldsymbol{s}) = P(\boldsymbol{s})\prod_{i=1}^{m} P(y_i|\mathcal{S}_i), \tag{6}$$

where $\mathcal{S}_i = \{s_j|(\boldsymbol{\Phi})_{i,j} = 1\}$. Note that the factorization in (6) is dictated by the pooling pattern $\boldsymbol{\Phi}$, while the combined stochastic effect of the pooling and measurement procedures is encapsulated in the conditional distribution $P(y_i|\mathcal{S}_i)$. By defining $g_i(\mathcal{S}_i) := P(y_i|\mathcal{S}_i)$, (6) can be represented as a factor graph as illustrated in Fig. 3, with m *function nodes* $\{g_i\}$ and n *variable nodes* $\{s_i\}$.

LBP is an iterative algorithm that approximates the marginal a-posteriori probabilities by propagating messages in the factor graph domain according to a flooding schedule [49]. Messages are initialized $\{\mu^{(0)}_{g_i\to s_j}(s), \mu^{(0)}_{s_j\to g_i}(s)\}$ for each $s \in \mathcal{Q}$ and for every function node g_i and variable node s_j which share an edge in the factor graph, i.e., $\boldsymbol{\Phi}_{i,j} = 1$. Then, for each vertex, the messages are shared downstream with their neighbors in an iterative fashion. For iteration index t, the variable-to-function messages are computed via

$$\mu^{(t+1)}_{s_j\to g_i}(s) = \prod_{f\in\mathcal{N}(s_j)/g_i} \mu^{(t)}_{f\to s_j}(s), \tag{7}$$

and the function-to-variable messages are obtained as

$$\mu^{(t+1)}_{g_i\to s_j}(s) = \sum_{\mathcal{S}_i/s_j} g_i(\mathcal{S}_i) \prod_{z\in\mathcal{N}(g_i)/s_j} \mu^{(t)}_{z\to g_i}(z), \tag{8}$$

where $\mathcal{N}(v)$ denotes the set of neighbors of a given node v. Note that the neighbours of variable nodes are all function nodes, while the neighbors of function nodes are all variable nodes.

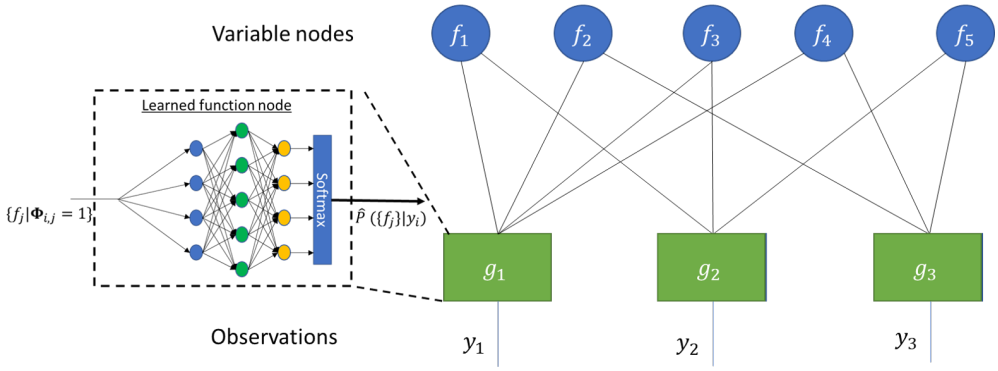

Figure 4: Pooling procedure with loopy learned factor graph.

Assuming convergence after T iterations, the a-posteriori probability can be approximated as the product of all incoming messages

$$\hat{P}(s_j = s|\boldsymbol{y}) \propto \prod_{f\in\mathcal{N}(s_j)} \mu^{(T)}_{f\to s_j}(s). \tag{9}$$

The resulting LBP algorithm is summarized below as Algorithm 5.

Algorithm 5: Loopy Belief Propagation [50, Ch. 26]

Init: Pooled measurements $\boldsymbol{y}\in\mathcal{R}_+^m$, pooling matrix $\mathbf{\Phi}$

Initialize $\{\mu^{(0)}_{f_i\to s_j}(s), \mu^{(0)}_{s_j\to f_i}(s)\}$ uniformly, i.e., setting $(\mu)_i = \frac{1}{|\mathcal{Q}|}$ for each $i\in\mathcal{Q}$;

for $t = 0, \cdots T-1$ **do**

  Update variable-to-function messages via (7);

  Update function-to-variable messages via (8) to compute $f(\mathcal{S}_i)$;

end

for $j = 1\ldots, n$ **do**

  Recover infection levels of possibly defective samples by

$$\hat{f}_j = \arg\max_{s\in\mathcal{Q}} \prod_{f\in\mathcal{N}(s_j)} \mu^{(T)}_{f\to s_j}(s). \tag{10}$$

end

Algorithm 5 is suitable for non-linear and noisy settings, and is not constrained to detecting infection, being allowed to produce estimates within any predefined finite dictionary. However, it can be computationally demanding, particularly when the number of patients n is relatively large. For this reason, it is often preferable to carry out COMP as a preliminary stage for reducing the dimensionality of the problem. Furthermore, it relies on knowledge of the stochastic model relating the viral loads and the measurements. In traditional GT, simplified models are often adopted [41, Ch. 3]. In COVID-19 testing, when the measurements correspond to the RT-PCR readings of the pool tests, one may either resort to approximated models, as suggested in Sec. 2.2. Alternatively, as proposed in [39], deep learning tools can be exploited to learn to compute the function nodes as a form of learned factor graphs [51], by training a neural classifier to predict $\mathcal{S}_i = \{s_j|\boldsymbol{A}_{i,j}=1\}\in\mathcal{Q}^{|\mathcal{S}_i|}$ from y_i. This form of AI-aided recovery allows to carry out LBP recovery with either binary levels (Example 1) or multi-level classification (Example 2) over the learned factor graph as illustrated in Fig. 4. This form of recovery preserves the suitability of LBP to exploit the statistical structures induced by known pooling pattern $\mathbf{\Phi}$ while leveraging to model-agnostic nature of neural networks to bypass the need to impose an approximated model on the pooling and RT-PCR measurement processes.

3.3 Summary

GT provides powerful AI methods for pooling and recovery from pooled measurements. These techniques are most suitable when one is interested in merely identifying the presence of infection, as in Example 1, and when the pooling and measurement procedures can be reliably modeled as linear (and preferably noiseless) operations. GT tools can be applied in both an adaptive and a non-adaptive manner, where the former is quite intuitive yet is less attractive for COVID-19 tests due to its associated overhead in terms of additional technician labour. Furthermore, GT methods, and particularly COMP, can be used as pre-processing steps combined with alternative recovery algorithms, including CS-based discussed in the following section, as means for reducing the dimensionality and complexity of automated recovery.

4 Compressed Sensing for Pooled Testing for COVID-19

CS is a popular sub-field in the areas of signal and image processing [9, 10]. It involves the acquisition of signals or images directly in compressed format, with a focus on saving acquisition or measurement time. This is different from conventional sensing where a signal or image is acquired in its entirety followed by possibly lossy compression using techniques such as JPEG, JPEG2000 or MPEG.

Consider a signal/image $\boldsymbol{f}$ in vectorized format with n elements. Instead of directly measuring every element of $\boldsymbol{f}$, it is sensed indirectly via a 'sensing matrix' $\boldsymbol{\Phi}$ of size $m \times n, m \ll n$ leading to the acquisition of m measurements in a vector $\boldsymbol{y}$. This is represented mathematically as:

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{f} + \boldsymbol{\eta}, \tag{11}$$

where $\boldsymbol{\eta}$ is a noise vector with m elements. We note that the matrix-vector operation $\boldsymbol{\Phi}\boldsymbol{f}$ is accomplished via hardware, in a specific manner dictated by the architecture of the compressive device. The aim is to recover $\boldsymbol{f}$ given $\boldsymbol{y}$ and $\boldsymbol{\Phi}$. This problem is ill-posed in general since $m \ll n$ and this forms an under-determined linear system, even if $\boldsymbol{\eta}$ were equal to the zero vector. However CS theory states that in the absence of measurement noise, this system of linear equations has a unique solution provided two sufficient conditions are met: $(Q1)$ the underlying signal $\boldsymbol{f}$ is sparse, i.e. most of its entries are zero, and $(Q2)$ the sensing matrix $\boldsymbol{\Phi}$ has a nullspace which contains no sparse vector other than the zero vector. These conditions are sufficient to ensure unique recovery of $\boldsymbol{f}$ from $\boldsymbol{y}, \boldsymbol{\Phi}$ [52]. Moreover, the recovery of $\boldsymbol{f}$ can be accomplished by efficient techniques such as linear programming [53] and various greedy methods. Furthermore, the error in the recovery of $\boldsymbol{f}$ is robust against noise and increases gradually as the noise variance increases.

In many imaging applications, $\boldsymbol{f}$ is not sparse in itself, but is expressible as a sparse or weakly sparse linear combination of column vectors from an orthonormal basis $\boldsymbol{\Psi}$ such as the discrete wavelet transform (DWT) or discrete cosine transform (DCT) in the following manner:

$$\boldsymbol{f} = \boldsymbol{\Psi}\boldsymbol{\theta} = \sum_{l=0}^{n} \boldsymbol{\Psi_l}\theta_l. \tag{12}$$

Here a weakly sparse linear combination refers to a linear combination in which most of the coefficients are equal to or very close to zero. That is, the coefficient vector $\boldsymbol{\theta}$ is sparse or weakly-sparse. In this case, the measurement model can be written as:

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{\Psi}\boldsymbol{\theta} + \boldsymbol{\eta}, \tag{13}$$

and we seek to recover $\boldsymbol{\theta}$ from $\boldsymbol{y}, \boldsymbol{\Phi}, \boldsymbol{\Psi}$. Upon recovering $\boldsymbol{\theta}$, it is easy to recover $\boldsymbol{f}$ since $\boldsymbol{f} = \boldsymbol{\Psi}\boldsymbol{\theta}$. Recovery can be efficiently performed using estimators such as the LASSO:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{\lambda}} = \text{argmin}_{\boldsymbol{\theta}} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\Psi}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1 \tag{14}$$

or basis pursuit denoising:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{\lambda}} = \text{argmin}_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|_1 \text{ s.t. } \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\Psi}\boldsymbol{\theta}\|_2 \leq \varepsilon \tag{15}$$

where λ and ε depend on the variance of the noise in $\boldsymbol{\eta}$. In both cases, there are theoretical guarantees for the recovery of $\boldsymbol{\theta}$ [53,54]. These guarantees extend to a variety of different noise models. Matrices generated randomly from the zero mean Gaussian distribution or the ± 1 Bernoulli distribution have been proved to

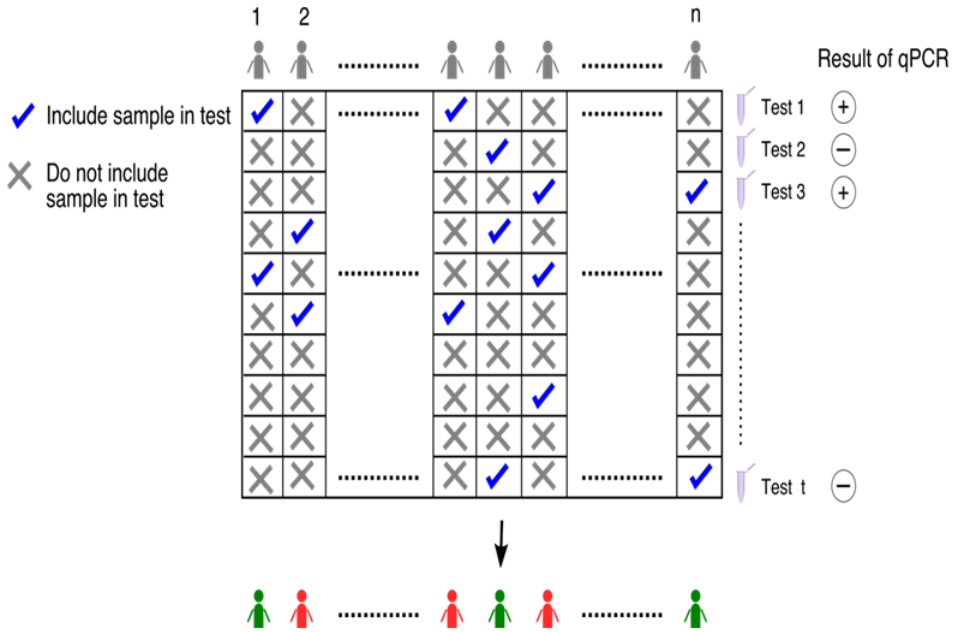

Figure 5: RT-PCR pooling process in a compressive framework. The results of $m = t$ RT-PCR pools are on the RHS in the form of vector $\boldsymbol{y}$, which is a noisy version of $\boldsymbol{\Phi f}$. The entries of $\boldsymbol{\Phi}$ are represented by crosses (0) and ticks (1).

obey a property called the Restricted Isometry Property (RIP) [55] with very high probability provided m is lower bounded by a quantity proportional to $s \log n$ where s is the number of non-zero entries in the underlying signal. The RIP implies the property $(Q2)$ mentioned earlier involving the null-space.

4.1 Compressed Sensing Forward Model for Pooled RT-PCR

A model similar to that in (11) can be presented for pooled inference in the case of RT-PCR. Using the notations introduced in Sec. 2, this can be expressed by writing $g_{\text{pool}}(\boldsymbol{f}) = \boldsymbol{\Phi f}$, such that the measurement output $\boldsymbol{y} = g_{\text{meas}}(\boldsymbol{\Phi f})$ is given by:

$$\boldsymbol{y} = \boldsymbol{\Phi f} \circ (1 + q_a)^{\boldsymbol{e}} \implies \forall i \in [m],\ y_i = \boldsymbol{\Phi^i f}(1 + q_a)^{e_i}, \tag{16}$$

where $\circ$ denotes the element-wise multiplication of two vectors. Here we note the following:

1. $\boldsymbol{f}$ is a real-valued but *sparse* vector of n elements where each entry f_i represents the viral load in the sample of the i^{th} individual. The sparsity assumption on $\boldsymbol{f}$ is essential for good quality recovery using CS algorithms and it is well justified due to the low prevalance rates in COVID-19 testing [56] as explained earlier.

2. $\boldsymbol{\Phi}$ is a $m \times n$ binary pooling matrix. A pool is created by mixing together small but equal portions of the samples of a subset of the n individuals. For the i^{th} pool, this information is contained in the non-zero entries of the i^{th} row of $\boldsymbol{\Phi}$, i.e. in $\boldsymbol{\Phi^i}$.

3. $\boldsymbol{y}$ is a vector of viral loads in the m different pools.

4. The noise model here is not additive unlike that in (11) which is typically used in CS, but multiplicative as explained in Sec. 2.2 with $q_a \in [0, 1]$ representing an amplification factor and each entry of $\boldsymbol{e}$ being distributed as per $\mathcal{N}(0, \sigma_e^2)$ where $0 < \sigma_e \ll 1$. Comparing with (3), the role of f_i is played by the appropriate viral load value in the i^{th} pool, given by $\boldsymbol{\Phi^i f}$.

The reader may refer to Fig. 5 for greater clarity in the pooling process.

4.2 CS Algorithms for Recovery

Referring to (16), the aim is to recover the sparse vector $\boldsymbol{f}$ from $\boldsymbol{y}, \boldsymbol{\Phi}$. Since $\boldsymbol{f}$ is real-valued (and also non-negative), such a procedure gives us viral-load values as opposed to just a binary indicator health status indicator ('infected' versus 'non-infected') as obtained from traditional group testing techniques surveyed in Sec. 3.

As the pandemic began spreading all over the world, a number of research works that used CS estimators for pooled testing emerged in the literature. These techniques include the Tapestry approach [17], the multi-stage approach in [38], the P-BEST method [57] and the initial work in [58]. We survey the algorithms employed in these techniques here below:

1. In the Tapestry approach [17, 59], a number of different CS algorithms were explored and compared – on synthetically generated as well as real data. These algorithms include the LASSO [54] with non-negativity constraints (NNLASSO) (see 14), sparse Bayesian learning [60] SBL, non-negative least squares NNLS or non-negative least absolute deviation NNLAD [61], and greedy algorithms such as orthogonal matching pursuit [62] with non-negativity constraints NNOMP [63]. The NNLS and NNLAD approaches were also experimented with in [14]. These algorithms were executed after the traditional group testing algorithm COMP described in Sec. 3 and outlined in Alg. 3. This was deemed particularly advantageous if there exist some pools with zero viral load, which immediately implies that the viral loads of the samples which contributed to this pool are all zero. Using this COMP step improves prediction performance and also improves computational efficiency as it reduces the problem size. As COMP is only a binary approach, the subsequent steps involving estimation of real-valued viral loads remain very important. Theoretical properties of this combined approach were explored in detail in [17]. It was empirically observed that COMP followed by SBL yielded the best performance among all other approaches such as NNLASSO, NNLS, NNLAD and NNOMP. An adaptive, multi-stage extension of the Tapestry system was presented in [64] (also see Sec. 6).

2. The multi-stage approach in [38] adopts a unique multi-stage strategy that beautifully combines three techniques: a variant of COMP called 'definite defectives' (DD) described in Sec. 3 and outlined in Alg. 4, followed by a maximum-likelihood approach from [65] (also called 'Boolean compressed sensing'), followed by an iterative viral loads estimation algorithm. Extensive experiments showcase the beneficial contribution of each of the three stages.

3. The P-BEST approach [57] followed the same strategy as their group's earlier work on pooled testing in agricultural applications [66, 67]. The estimator here is a form of NNLASSO but with the viral loads quantized to three values: 0,1,2 (see Eqn. 1 of [66]), representing zero viral loads, intermediate viral loads and high viral loads respectively.

4. To our best knowledge, the earliest piece of work on use of CS for COVID-19 testing appeared in [58] where a basis pursuit denoising (BPDN) approach (see 15) was adopted.

4.2.1 Details of Algorithms

For the sake of completeness, we briefly explain the various algorithms referred to in the previous section: NNLASSO, BPDN, NNLAD, NNLS, SBL and NNOMP.

1. NNLASSO: This approach works by minimizing the following cost function:

$$J_1(\boldsymbol{f}) \triangleq \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{f}\|_2^2 + \lambda \|\boldsymbol{f}\|_1 \text{ s. t. } \boldsymbol{f} \succeq \boldsymbol{0}, \tag{17}$$

   where $\succeq$ refers to an element-wise $\geq$ inequality and $\boldsymbol{0}$ refers to a vector of zeros with the same number of elements as in $\boldsymbol{f}$. The minimization procedure proceeds by standard algorithms such as iterative shrinkage and thresholding (ISTA) or its fast variants [68]. The regularization parameter λ ideally depends on the noise level but can be selected using cross-validation.

2. BPDN: This is a constrained version of the earlier NNLASSO problem, and acquires the form of minimizing the following cost function:

$$J_2(\boldsymbol{f}) \triangleq \|\boldsymbol{f}\|_1 \text{ s. t. } \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{f}\|_2^2 \leq \varepsilon \text{ and } \boldsymbol{f} \succeq \boldsymbol{0}, \tag{18}$$

where ε depends on the noise level.

3. NNLAD and NNLS: These methods work based on minimizing only the data fidelity term with a non-negativity constraint. This non-negativity constraint is enough to guarantee uniqueness and stability of the solution with high probability, in conjunction with a random binary matrix $\mathbf{\Phi}$, as analyzed in [14, 61]. The cost function for NNLAD is given as follows:

$$J_3(\boldsymbol{f}) \triangleq \|\boldsymbol{y} - \mathbf{\Phi}\boldsymbol{f}\|_1 \text{ s. t. } \boldsymbol{f} \succeq \mathbf{0}. \tag{19}$$

The cost function for NNLS is

$$J_4(\boldsymbol{f}) \triangleq \|\boldsymbol{y} - \mathbf{\Phi}\boldsymbol{f}\|_2^2 \text{ s. t. } \boldsymbol{f} \succeq \mathbf{0}. \tag{20}$$

4. NNOMP: This is a variant of the popular orthogonal matching pursuit (OMP) algorithm. OMP is a very well known greedy approximation technique [62], and the variant imposes a non-negativity constraint on the solution. The original algorithm requires the solution of multiple least-squares solution of increasing sizes in every iteration. In the variant, we would require least-squares solutions with non-negativity constraints which adds to the overall computational cost. Fast variants have been developed in [63].

5. Sparse Bayesian Learning (SBL): This [60, 69] is a non-convex optimization algorithm based on Expectation-Maximization (EM). In recent empirical studies on compressed sensing, SBL has demonstrated the best reconstruction performance in comparison to many other algorithms [70]. SBL considers a Gaussian likelihood function for $\boldsymbol{y}$ given $\mathbf{\Phi}\boldsymbol{f}$ and a Gaussian prior on every element of $\boldsymbol{f}$ with different unknown variances for each element, in the following manner:

$$p(\boldsymbol{y}|\mathbf{\Phi}\boldsymbol{f}) = \frac{\exp(-\|\boldsymbol{y} - \mathbf{\Phi}\boldsymbol{f}\|_2^2/(2\sigma^2))}{(2\pi\sigma^2)^{m/2}} \tag{21}$$

$$p(f_i;\gamma_i) = \frac{\exp(-f_i^2/(2\gamma_i))}{\sqrt{2\pi\gamma_i}}; \gamma_i \geq 0. \tag{22}$$

In this formulation, note that both $\boldsymbol{x}$ and $\{\gamma_i\}_{i=1}^n$ are unknown. Many theoretical results regarding the properties of the EM procedure in SBL are analyzed in [60]. In particular, it is proved that all local minima are sparse vectors. Unlike the previous algorithms, no non-negativity constraint on the elements of $\boldsymbol{f}$ is explicitly imposed.

Most of the aforementioned use a squared loss for the fidelity function which is essentially related to additive Gaussian noise. However, we know that for CS used for RT-PCR, the noise is multiplicative as seen in Sec. 2.2 and (16). However, following [71, Sec. 7], if we perform a first order Taylor series approximation on both sides of (16), we obtain the following:

$$y_i \approx \mathbf{\Phi}^{\boldsymbol{i}}\boldsymbol{f} + \mathbf{\Phi}^{\boldsymbol{i}}\boldsymbol{f}[\log(1+q_a)]e_i. \tag{23}$$

This approximation is reasonably accurate because the error term e_i (see Sec. 2.2) is a Gaussian random variable with a small variance. Given this approximated noise model, one would have preferred a fidelity term of the form $\sum_{i=1}^m (y_i - \mathbf{\Phi}^{\boldsymbol{i}}\boldsymbol{f})^2/(\mathbf{\Phi}^{\boldsymbol{i}}\boldsymbol{f})^2$. However this function is non-convex and the squared loss $\sum_{i=1}^m (y_i - \mathbf{\Phi}^{\boldsymbol{i}}\boldsymbol{f})^2$ has been widely used instead.

As explained here, CS-based algorithms infer quantitative viral load information. Although a non-infected person has a viral load of zero, the algorithms may report very small non-zero viral loads for such samples. This is due to small algorithmic biases, which can produce unduly low specificity rates (high false positives). To prevent this, the viral loads must be put through some threshold τ such that any estimated viral load value (call it $\hat{x}_i$ is to be regarded as zero if $\hat{x}_i < \tau$, and non-zero otherwise. Such strategies have been followed in the literature [17]. The optimal choice of the threshold τ needs to be decided a priori by 'learning' on past data comparing pooling to sequential testing so as to optimize one of the performance measures outlined in Sec. 4.3. The optimal threshold τ may potentially vary based on prevalence rates.

4.3 Assessment of Algorithm Performance and Experimental Protocols

The various algorithms in the literature are typically compared with respect to the following criteria, where the $\hat{\boldsymbol{f}}$ refers to an estimate of $\boldsymbol{f}$:

1. RMSE $:= \|\boldsymbol{f} - \hat{\boldsymbol{f}}\|_2 / \|\boldsymbol{f}\|_2$
2. Number of false positives (FP) $:= |\{i : f_i = 0, \hat{f}_i > 0\}|$
3. Number of false negatives (FN) $:= |\{i : f_i > 0, \hat{f}_i = 0\}|$
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.

The experiments to assess the performance of the various CS algorithms can be classified broadly as experiments on (1) synthetic and (2) real data.

For the prediction of $\boldsymbol{f}$ from $\boldsymbol{y}, \boldsymbol{\Phi}$ from synthetically generated data [14,17,38,58], the vector $\boldsymbol{f}$ is chosen to be sufficiently sparse and the non-zero entries are drawn uniformly at random from the interval $[0, 32768]$. Alternatively, the distribution of cycle threshold (CT) values [20, Fig. 3] could be used to generate CT data and convert those into viral loads, although such an approach has not been followed in the literature to our best knowledge. The pools have been synthetically simulated using the noise model from (16) using different types of $\boldsymbol{\Phi}$ matrices, as surveyed later in Sec. 4.4.

For the experiments on real data, the work in [17] used artificially injected viral RNA to create positive samples. These data were obtained from the Wyss Institute at Harvard, USA and the National Centre for Biological Sciences, India. In [17], pooling matrices of sizes 16×40 and 24×60 were used and the number of infected samples was varied between 0 and 4 (prevalence rate up to 10%). Experimental results on pooled inference showed mostly 0 and much less commonly 1 or 2 false negatives, and between 0 and 3 false positives with the distribution of false positives skewed heavily towards 0 or 1. The work in [57] used left-over samples that had been previously tested for COVID19. Their experiments considered pooling matrices of size 48×384 with the number of positive samples ranging between 2 to 5 (prevalence rate less than 1%). Their approach yielded no false negatives and an occasional false positive.

4.4 Choice of Pooling Matrices

Pooling matrices for compressive recovery in pooled testing must allow for successful recovery of $\boldsymbol{f}$ from $\boldsymbol{y}$. Moreover, unlike typical CS, these matrices must also obey certain additional physical constraints (unique to pooling tasks) which have been explained elaborately in [17,71]. We briefly explain these constraints here below. These constraint have been mentioned earlier in Sec. 2.3, but we explain them here again in the context of specifying properties of compressed sensing matrices.

1. The pooling matrix must be binary, for ease of pooling. Thus, $\Phi_{ij} = 0$ means that the j^{th} sample did not contribute to the i^{th} pool, and $\Phi_{ij} = 1$ means that a (small) fixed volume from the j^{th} sample contributed to the i^{th} pool. Note that the contributing volumes are equal across pools and across samples. It is understood that the total sample of a single individual will contribute to multiple pools. Hence while creating a pool, only small (fixed-volume) portions from the contributing samples are collected and mixed together.

2. They must be sparse, again for ease of pooling. Sparsity of $\boldsymbol{\Phi}$ is required to ensure two important properties: $(Z1)$ that not too many samples contribute to a given pool, and $(Z2)$ that a single sample does not contribute to too many different pools. Property $(Z1)$ is important because the sample volume added to an RT-PCR reaction in the RT-PCR machine is fixed. Due to this, an increase in pool size means that each sample would contribute a smaller amount of that volume to any given pool. This can adversely affect the power of RT-PCR to differentiate positive samples from negative ones. Property $(Z2)$ is important because the contribution of one sample to a very large number of pools could lead to sample depletion.

Several different types of combinatorial design strategies have been used in the literature to design $\mathbf{\Phi}$ matrices so as to obey the aforementioned constraints. The work in [17] uses Kirkman triple matrices where each sample contributes to exactly 3 pools and each pool is created from a fixed number of samples. These $m \times n$ Kirkman matrices also have additional flexibility in terms of size, namely that n can be any integer multiple of $C(m, 3)$. Kirkman matrices are also adjacency matrices that correspond to certain types of 'expander graphs' which have interesting properties in compressed sensing, including obeying a variant of the RIP as discussed in detail in [17]. As they satisfy the RIP, it follows that they allow for good compressive recovery. The work in [57, 66] uses matrices designed from Reed-Solomon codes seeking inspiration from the rich classical literature on error correcting codes [72], allowing inherent robustness to experimental noise and variations in viral concentration.

In addition, there exists a large body of literature on the combinatorial design of binary sensing matrices [73–75]. However these designs are applicable to only fixed sizes, which may not be relevant to the sizes required for RT-PCR pooling. For example, the proposed designs in [75] have sizes $r^2 \times r^{l+1}$ where r is a prime power, and l is an integer such that $1 < l < r$. This choice of matrix size may not be immediately relevant to RT-PCR and has unnatural restrictions. To address this issue, the recent work in [71] considers sparse balanced binary matrices of arbitrary size. These matrices have the property that all columns of the matrix contain an equal number of ones, and that all rows of the matrix also contain an equal number of ones. These can be designed for a much wider variety of matrix sizes.

4.5 Choice of Number of Pools

The choice of m, the number of pools, is an important aspect of the design of $\mathbf{\Phi}$. Let us denote the number of non-zero entries in the underlying viral loads vector $\boldsymbol{f}$ by s. Intuitively speaking, the smaller the value of s, the smaller will be the required number of pools m. As per compressed sensing theory, m is $\Omega(s \log n)$ [53,76]. However s is usually unknown, and hence configuring an acceptable number of pools is a difficult task. There are two ways to get some rough estimate of s: (1) use knowledge of prevalence rates from a particular city or area where the pooling is being performed, and (2) determine s on the fly directly from the pooled tests. The latter has been accomplished in [17] using binary pooling matrices $\mathbf{\Phi}$ using properties of the binomial distribution, based on a technique proposed in [77, 78]. These techniques, of course, provide only estimates which could be error-prone, but they do give a good idea of the number of pools required. In fact, if it turns out that s is too high, pooling may not longer be a viable option due to loss of accuracy, and sequential testing is then preferable [17]. In the present literature [17,38,57,58], the number of pools is chosen somewhat conservatively assuming low values of s, and without necessarily always computing s as a first step.

5 Use of Side Information in Pooled Inference

In this section, we cover a very different aspect of pooled inference – namely, the use of extra information, or side information. We first describe the different types of side information, followed by its use in CS algorithms and then its use for improving traditional group testing methods such as Dorfman pooling.

The sparsity of $\boldsymbol{f}$ is a useful and valid assumption given the generally low prevalence rates for COVID-19 [56]. However, there is additional side information (SI) that is available in such applications, which can improve the performance of pooled inference. One form of SI is knowledge of the prevalence rate. This was exploited by means of a probabilistic decoder based on message passing in [79]. Other forms of SI can include knowledge of an individual's symptoms, medical history and travel history. Over and above this, knowledge of family memberships is useful SI, because COVID-19 is likely to spread from one family member to another. Here the notion of a family includes not just the conventional notion, but also includes groups of people in frequent close contact, such as health care professionals working in the same hospital wing, students sharing the same dorm, security officers working at the same checkpoint, etc. Furthermore, contact tracing has been widely employed as a means to control the pandemic [80, 81]. Such information, includes the duration of contact between pairs of individuals and measures of physical proximity (distance). Such information can be collected using one or more of the following methods: Bluetooth in mobile phones [82], the global positioning system [83], manual inquiries by social workers [84, 85], and financial transaction data [81, 83]. There are privacy issues associated with the collection of contact tracing or family information, however it should be

noted that these types of side information have one advantage – namely that they do not require collection of medical history information or medical data.

In [71,86], two classes of estimators are explored to explicitly account for family information and contact tracing information to improve the performance of pooled inference. The first class of techniques involves extensions of the LASSO such as the group LASSO [87] or the overlapping group LASSO [88]. The group LASSO uses the fact that the number of infected *families* is small (as opposed to the number of individuals, as in normal sparsity). This is extended to handle overlapping 'families' as determined from contact tracing information. This class of techniques uses quantitative information and the multiplicative noise model in RT-PCR. The second class of techniques in [71,86] involves extensions of Bayesian algorithms such as generalized approximate matching pursuit (GAMP) to account for family structure and contact tracing information. This model is explicitly for binary health status vectors (as opposed to viral load vectors) and accounts for binary noise in the pooled results. This latter set of techniques explicitly accounts for person-to-person transmission probabilities. Both classes of techniques demonstrate superior empirical performance (in terms of number of false positives and false negatives) over their counterparts which account for no family or contact tracing SI and use only the sparsity of the vector of viral loads or health statuses. Independent work in [89, 90] also demonstrated the benefits of community-based SI on the performance of pooling. The emphasis of the work in [89, 90] is to use the community information to design effective encoders, for which the authors provide theoretical analysis. They employ a loopy belief propagation (LBP) decoder which accounts for community information and show that it outperforms the basic LBP variant which does not account for such information. Their decoders work in tandem with binary health status vectors.

There exist many other pieces of work which account for SI, mostly in the form of prevalence rates or prior infection probabilities, but they do not do so in a non-adaptive CS framework but rather in the framework of traditional group testing algorithms such as Dorfman pooling. Such individual prior infection probabilities can be derived based on factors such as medical or travel history, age, gender, etc. In particular in works such as [91–93], there is emphasis on segregating people into different groups based on their individual prior infection probabilities, and performing pooling of samples belonging to individuals from a given group only. This is in tune with recent publications from the WHO [94] which note reduction in the number of tests if pooling is performed on samples of people with similar infection probability values. There also exist works such as [95–98] which account for correlations between samples of different individuals (represented as nodes of a graph). Such correlations can be obtained from various aspects of a social graph or via contact tracing information. However the works in [95–98] propose adaptive algorithms, quite unlike the non-adaptive CS based algorithms considered in [71, 86, 89, 90].

6 Comparative Discussion and Summary

In this chapter, we surveyed the applications of AI techniques influenced by GT and CS algorithms to pooled testing of COVID-19 RT-PCR samples. Here, we present a comparative discussion of the relative advantages and disadvantages of the two approaches:

1. Traditional approaches such as Dorfman's method are two-stage procedures. In particular, the output of the first stage acts as an input to the second stage. In both stages, RT-PCR testing is required, and the two stages cannot be executed in parallel. Such approaches are termed 'adaptive'. On the other hand, non-adaptive GT and CS based approaches test all pools in *parallel* and the individual health status values are made available. These are only single-stage approaches and also referred to as being 'one-shot'. Note that even if these non-adaptive approaches employ different algorithms in a sequential manner (*e.g.*, [17, 38]), there is only one *testing* stage. As RT-PCR is a time-consuming process requiring about 4 hours, a single-stage procedure is more efficient in terms of time.

2. Adaptive approaches typically have shown lower false-negative and false-positive rates as compared to non-adaptive approaches [17, 64], even though the non-adaptive approaches do perform quite well. However the work in [17] which focuses on non-adaptive CS-based algorithms typically reports a requirement of fewer tests than with Dorfman's method, besides the fact that the former algorithms do not require more than one stage of testing unlike Dorfman's method which requires two stages. A similar set of simulations were run here for different number of infected samples (in simulation) and

comparative results for Dorfman pooling and the COMP-SBL algorithm from Sec. 4.2.1 are presented in Fig. 6 and 7.

3. Both classes of methods require prior knowledge of the number of non-zeros in $\boldsymbol{f}$ to configure the optimal number of tests. Also see Sec. 4.5. In the existing literature, however, this knowledge has not been explicitly used for configuring the number of required tests.

4. With some exceptions, most adaptive techniques work in a regime where $\boldsymbol{f}$ is binary. The selection of non-adaptive approaches should typically depend on what type of output is required. When one seeks to recover the viral loads for diagnosis (as in Example 3), non-adaptive approaches that use CS naturally treat $\boldsymbol{f}$ as non-negative real-valued [17, 38, 57, 58]. When one is interested in merely identifying the presence of infection, GT methods are natural candidates being designed for regimes where $\boldsymbol{f}$ is binary, though one can also adopt binary variations of CS techniques, see, e.g., [79, 89, 90].

Pooling for RT-PCR has been employed on the ground in several countries and university campuses. An incomplete list of this is presented in [99]. In most of these places, each pool typically contained contributions from about a dozen or two dozen samples. The listing at [99] includes many provinces/states in countries such as USA, UK Switzerland, India, Ecuador, Israel, China, Belgium, Brazil and many others. The listing mentions several large pooling based studies or experiments, involving thousands and occasionally even millions of samples. Most of these experiments employed GT techniques, most commonly Dorfman pooling (Algorithm 1), although a few employed CS based methods [57, 100] as explained in Sec. 4. The usage of pooling was shown to yield notable savings and considerable increase in testing throughput [101]. We believe that this widespread usage of pooling amply illustrates its practical relevance during this pandemic. This also illustrates the importance of artificial intelligence and algorithmic tools in combating the spread of COVID-19.

The AI algorithms covered in this survey chapter typically do not require training in the conventional sense that many AI techniques use. However the algorithms can benefit from prior information which can be 'learned' by observing relevant COVID-19 data. This includes inference of prevalence rates as explained in Sec. 4.5, or the choice of optimal thresholds in compressed sensing algorithms (see Sec. 4.2.1) to decide whether certain viral loads are to be regarded as zero or non-zero, or side information as explained in Sec. 5. One could further incorporate ideas from AI or ML techniques that have mined useful knowledge from available data regarding the spread of COVID-19, including information from the epidemiology literature [102]. The incorporation of such information to improve the performance of pooling is a good avenue for future research.

References

[1] Salathé M, Althaus CL, Neher R, Stringhini S, Hodcroft E, Fellay J, et al. COVID-19 epidemic in Switzerland: on the importance of testing, contact tracing and isolation. Swiss medical weekly. 2020;150(11-12):w20225.

[2] Emanuel EJ, Persad G, Upshur R, Thome B, Parker M, Glickman A, et al.. Fair allocation of scarce medical resources in the time of Covid-19. Mass Medical Soc; 2020.

[3] Lucia C, Federico PB, Alejandra GC. An ultrasensitive, rapid, and portable coronavirus SARS-CoV-2 sequence detection method based on CRISPR-Cas12. bioRxiv. 2020.

[4] Ben-Assa N, Naddaf R, Gefen T, Capucha T, Hajjo H, Mandelbaum N, et al. SARS-CoV-2 On-the-Spot Virus Detection Directly From Patients. medRxiv. 2020.

[5] Nolan T, Hands RE, Bustin SA. Quantification of mRNA using real-time RT-PCR. Nature protocols. 2006;1(3):1559-82.

[6] Yelin I, Aharony N, Shaer-Tamar E, Argoetti A, Messer E, Berenbaum D, et al. Evaluation of COVID-19 RT-qPCR test in multi-sample pools. MedRxiv. 2020.

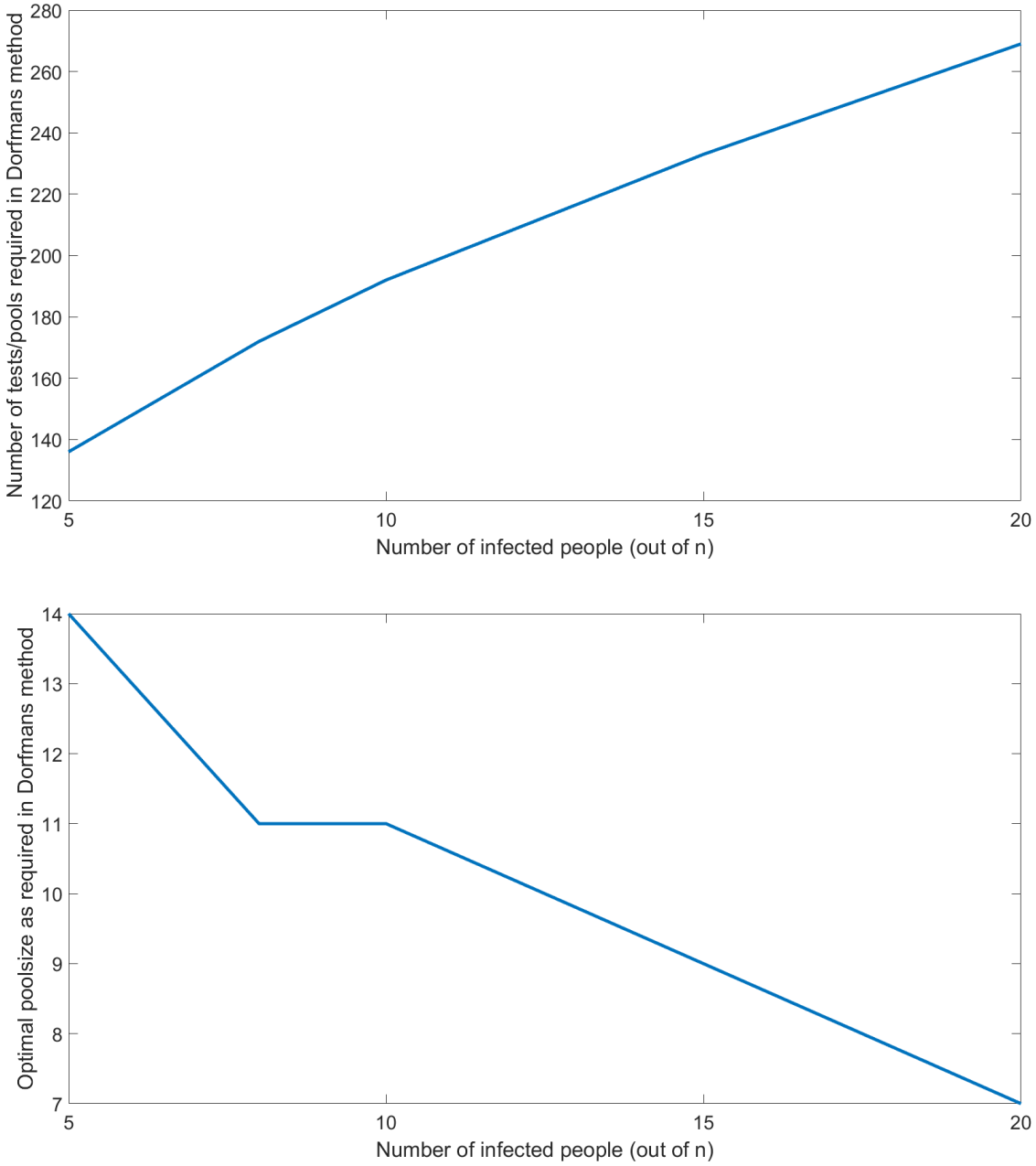

Figure 6: Average number of tests, i.e. number of pools (top sub-figure) and optimal poolsize (bottom sub-figure), required for Dorfman pooling with $n = 961$ samples, with s denoting the number of infected samples. Dorfman pooling yields no false positives or false negatives. Compare these results with the sensitivity and specificity results with the CS algorithm Comp-Sbl in Fig. 7. These results are similar to Tables 3 and 6 of [17].

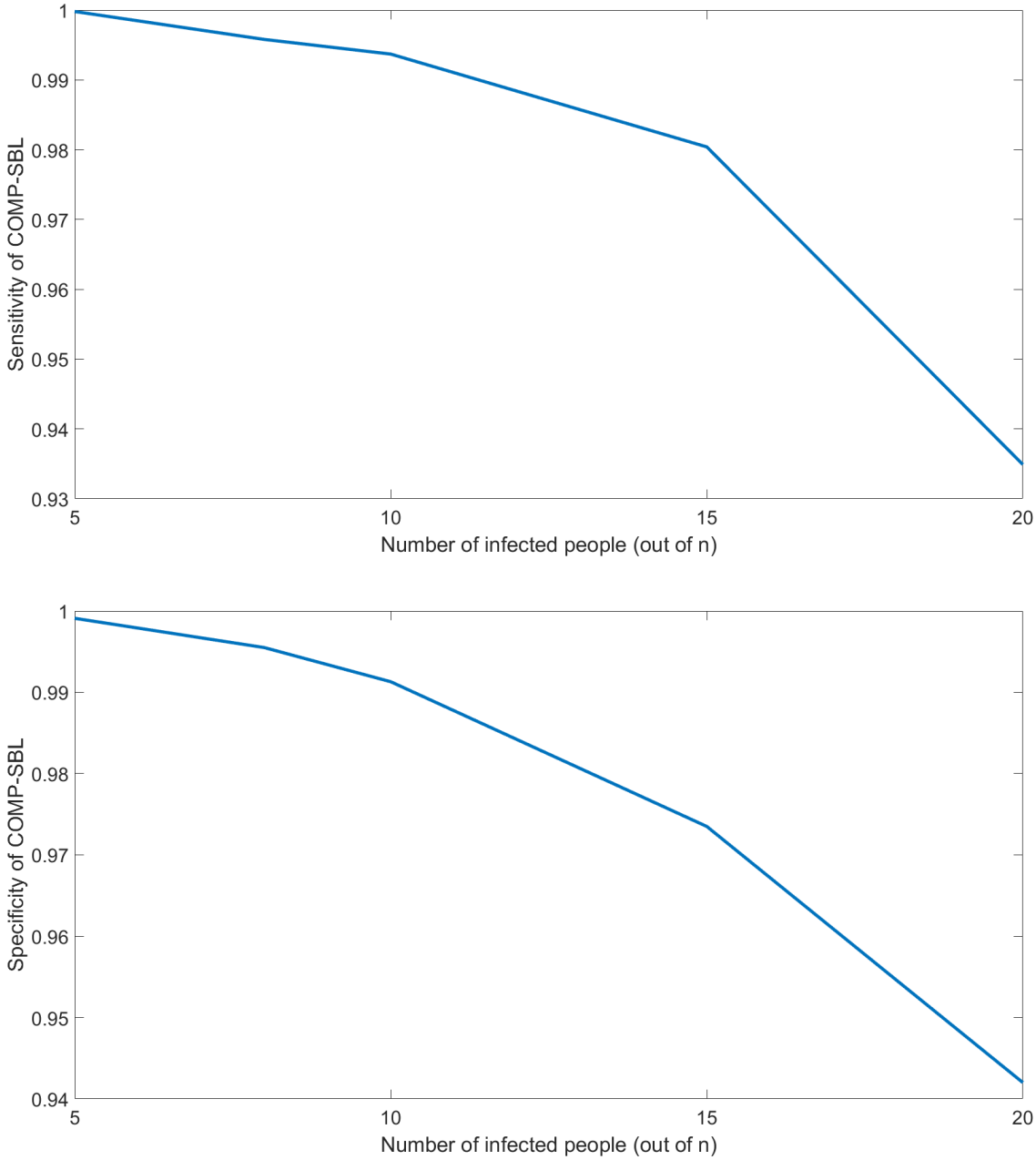

Figure 7: Average sensitivity (top sub-figure) and specificity (bottom sub-figure) results for the CS algorithm Comp-Sbl with $m = 93$ pools and $n = 961$ samples, where the number of infected samples is denoted as s. Compare these results with the number of tests required for Dorfman pooling in Fig. 6. The variance values for all entries are very small. As can be seen, the number of pools required is smaller than that for Dorfman pooling. These results are similar to Tables 3 and 6 of [17].

[7] Hanel R, Thurner S. Boosting test-efficiency by pooled testing strategies for SARS-CoV-2. arXiv preprint arXiv:200309944. 2020.

[8] Gilbert AC, Iwen MA, Strauss MJ. Group testing and sparse signal recovery. In: 2008 42nd Asilomar Conference on Signals, Systems and Computers. IEEE; 2008. p. 1059-63.

[9] Eldar YC, Kutyniok G. Compressed sensing: theory and applications. Cambridge university press; 2012.

[10] Eldar YC. Sampling theory: Beyond bandlimited systems. Cambridge University Press; 2015.

[11] Shental N, Levy S, Skorniakov S, Wuvshet V, Shemer-Avni Y, Porgador A, et al. Efficient high throughput SARS-CoV-2 testing to detect asymptomatic carriers. Science advances. 2020;6(37):eabc5961.

[12] Ghosh S, Agarwal R, Rehan MA, Pathak S, Agarwal P, Gupta Y, et al. A Compressed Sensing Approach to Pooled RT-PCR Testing for COVID-19 Detection. IEEE Open Journal of Signal Processing. early access, 2021.

[13] Yi J, Mudumbai R, Xu W. Low-Cost and High-Throughput Testing of COVID-19 Viruses and Antibodies via Compressed Sensing: System Concepts and Computational Experiments. arXiv preprint arXiv:200405759. 2020.

[14] Petersen HB, Bah B, Jung P. Practical High-Throughput, Non-Adaptive and Noise-Robust SARS-CoV-2 Testing. arXiv preprint arXiv:200709171. 2020.

[15] Yi J, Cho M, Wu X, Xu W, Mudumbai R. Error Correction Codes for COVID-19 Virus and Antibody Testing: Using Pooled Testing to Increase Test Reliability. arXiv preprint arXiv:200714919. 2020.

[16] Zhu J, Rivera K, Baron D. Noisy Pooled PCR for Virus Testing. arXiv preprint arXiv:200402689. 2020.

[17] Ghosh S, Agarwal R, Rehan MA, Pathak S, Agarwal P, Gupta Y, et al. A Compressed Sensing Approach to Pooled RT-PCR Testing for COVID-19 Detection. IEEE Open J Signal Process. 2021.

[18] Jawerth N. How is the COVID-19 Virus Detected using Real Time RT-PCR?;. `https://www.iaea.org/newscenter/news/how-is-the-covid-19-virus-detected-using-real-time-rt-pcr`.

[19] Efficiency of Real-Time PCR;. [Online; accessed 5-April-2021]. `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`.

[20] Buchan B, et al. Distribution of SARS-CoV-2 PCR Cycle Threshold Values Provide Practical Insight Into Overall and Target-Specific Sensitivity Among Symptomatic Patients. Am J Clin Pathol. 2020.

[21] Gevertz J, Dunn S, Roth C. Mathematical model of real-time PCR kinetics. Biotechnology and Bioengineering. 2005;92(3):346-55.

[22] Wyllie ALea. Saliva or Nasopharyngeal Swab Specimens for Detection of SARS-CoV-2. New England Journal of Medicine. 2020;383(13):1283-6.

[23] Allen JWL, Verkerke H, Owens J, Saeedi B, Boyer D, Shin S, et al. Serum pooling for rapid expansion of anti-SARS-CoV-2 antibody testing capacity. Transfusion Clinique et Biologique. 2021;28(1):51-4.

[24] Hughes-Oliver JM, Swallow WH. A two-stage adaptive group-testing procedure for estimating small proportions. Journal of the American Statistical Association. 1994;89(427):982-93.

[25] Dorfman R. The detection of defective members of large populations. The Annals of Mathematical Statistics. 1943;14(4):436-40.

[26] Du D, Hwang FK, Hwang F. Combinatorial group testing and its applications. vol. 12. World Scientific; 2000.

[27] Macula AJ. Probabilistic nonadaptive group testing in the presence of errors and DNA library screening. Annals of Combinatorics. 1999;3(1):61-9.

[28] Varanasi MK. Group detection for synchronous Gaussian code-division multiple-access channels. IEEE Transactions on Information Theory. 1995;41(4):1083-96.

[29] Cheraghchi M, Karbasi A, Mohajer S, Saligrama V. Graph-constrained group testing. IEEE Transactions on Information Theory. 2012;58(1):248-62.

[30] Wu S, Wei S, Wang Y, Vaidyanathan R, Yuan J. Partition information and its transmission over boolean multi-access channels. IEEE Transactions on Information Theory. 2014;61(2):1010-27.

[31] Bajwa WU, Haupt JD, Sayeed AM, Nowak RD. Joint source–channel communication for distributed estimation in sensor networks. IEEE Transactions on Information Theory. 2007;53(10):3629-53.

[32] Clifford R, Efremenko K, Porat E, Rothschild A. k-mismatch with don't cares. In: European Symposium on Algorithms. Springer; 2007. p. 151-62.

[33] Tsai WT, Chen Y, Cao Z, Bai X, Huang H, Paul R. Testing web services using progressive group testing. In: Advanced Workshop on Content Computing. Springer; 2004. p. 314-22.

[34] Cormode G, Muthukrishnan S. What's new: Finding significant differences in network data streams. IEEE/ACM Transactions on Networking. 2005;13(6):1219-32.

[35] Xuan Y, Shin I, Thai MT, Znati T. Detecting application denial-of-service attacks: A group-testing-based approach. IEEE Transactions on Parallel and Distributed Systems. 2009;21(8):1203-16.

[36] Goodrich MT, Atallah MJ, Tamassia R. Indexing information for data forensics. In: International Conference on Applied Cryptography and Network Security. Springer; 2005. p. 206-21.

[37] Chan CL, Jaggi S, Saligrama V, Agnihotri S. Non-adaptive group testing: Explicit bounds and novel algorithms. IEEE Transactions on Information Theory. 2014;60(5):3019-35.

[38] Cohen A, Shlezinger N, Solomon A, Eldar YC, Médard M. Multi-level group testing with application to one-shot pooled COVID-19 tests. In: Proc. IEEE ICASSP; 2021. .

[39] Ben-Knaan EF, Shlezinger N, , Eldar YC. Recovery of Noisy Pooled Tests via Learned Factor Graphs with Application to COVID-19 Testing. In: Proc. IEEE ICASSP; 2022. .

[40] Ben-Ami R, Klochendler A, Seidel M, Sido T, Gurel-Gurevich O, Yassour M, et al. Large-scale implementation of pooled RNA extraction and RT-PCR for SARS-CoV-2 detection. Clinical Microbiology and Infection. 2020.

[41] Aldridge M, Johnson O, Scarlett J. Group testing: an information theory perspective. Foundations and Trends® in Communications and Information Theory. 2019;15(3-4):196-392.

[42] Sobel M, Groll PA. Group testing to eliminate efficiently all defectives in a binomial sample. Bell System Technical Journal. 1959;38(5):1179-252.

[43] Hu M, Hwang F, Wang JK. A boundary problem for group testing. SIAM Journal on Algebraic Discrete Methods. 1981;2(2):81-7.

[44] Atia GK, Saligrama V. Boolean compressed sensing and noisy group testing. IEEE Transactions on Information Theory. 2012;58(3):1880-901.

[45] Kautz W, Singleton R. Nonrandom binary superimposed codes. IEEE Transactions on Information Theory. 1964;10(4):363-77.

[46] Chan CL, Che PH, Jaggi S, Saligrama V. Non-adaptive probabilistic group testing with noisy measurements: Near-optimal bounds with efficient algorithms. In: 2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton). IEEE; 2011. p. 1832-9.

[47] Aldridge M, Baldassini L, Johnson O. Group testing algorithms: Bounds and simulations. IEEE Transactions on Information Theory. 2014;60(6):3671-87.

[48] Weiss Y, Freeman WT. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. IEEE Transactions on Information Theory. 2001;47(2):736-44.

[49] Kschischang FR, Frey BJ, Loeliger HA. Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory. 2001;47(2):498-519.

[50] MacKay DJ. Information theory, inference and learning algorithms. Cambridge university press; 2003.

[51] Shlezinger N, Farsad N, Eldar YC, Goldsmith AJ. Inference from Stationary Time Sequences via Learned Factor Graphs. IEEE Transactions on Signal Processing. 2022.

[52] Davenport M, Duarte M, Eldar Y, Kutyniok G. Introduction to compressed sensing. In: Eldar Y, Kutyniok G, editors. Compressed Sensing: Theory and Applications. Cambridge University Press; 2012. p. 1–64.

[53] Candes E. The restricted isometry property and its implications for compressive sensing. Comptes Rendus Mathematiques. 2008.

[54] Hastie T, Tibshirani R, Wainwright M. Statistical Learning with Sparsity: The LASSO and Generalizations. CRC Press; 2015.

[55] Baraniuk R, Davenport M, DeVore R, Wakin M. A Simple Proof of the Restricted Isometry Property for Random Matrices. Constr Approx. 2008;28:253–263.

[56] Benatia D, Godefroy R, Lewis J. Estimating COVID-19 Prevalence in the United States: A Sample Selection Model Approach;. `https://www.medrxiv.org/content/10.1101/2020.04.20.20072942v1`.

[57] Shental N, et al. Efficient high throughput SARS-CoV-2 testing to detect asymptomatic carriers. Sci Adv. 2020 Sep;6(37).

[58] Yi J, Mudumbai R, Xu W. Low-Cost and High-Throughput Testing of COVID-19 Viruses and Antibodies via Compressed Sensing: System Concepts and Computational Experiments; 2020. `https://arxiv.org/abs/2004.05759`.

[59] et al SG. Tapestry: A Single-Round Smart Pooling Technique for COVID-19 Testing; 2020. `https://www.medrxiv.org/content/10.1101/2020.04.23.20077727v1`.

[60] Wipf D, Rao BD. Sparse Bayesian Learning for Basis Selection. IEEE Trans Signal Processing. 2004;52(8).

[61] Kueng R, Jung P. Robust Nonnegative Sparse Recovery and the Nullspace Property of 0/1 Measurements. IEEE Transactions on Information Theory. 2018;64(2):689-703.

[62] Pati Y, Rezaiifar R, Krishnaprasad P. Orthogonal Matching Pursuit: recursive function approximation with application to wavelet decomposition. In: Asilomar Conf. On Signals, Systems and Computing; 1993. p. 40-4.

[63] Yaghoobi M, Wu D, Davies M. Fast Non-Negative Orthogonal Matching Pursuit. IEEE Signal Processing Letters. 2015;22(9):1229-33.

[64] Heidarzadeh A, Narayanan K. Two-Stage Adaptive Pooling with RT-qPCR for COVID-19 Screening. In: ICASSP; 2021. .

[65] Atia GK, Saligrama V. Boolean compressed sensing and noisy group testing. IEEE Trans Inf Theory. 2012;58(3):1880–1901.

[66] Nida H, Blum S, Zielinski D, Srivastava DA, Elbaum R, Xin Z, et al. Highly efficient de novo mutant identification in a Sorghum bicolor TILLING population using the ComSeq approach. Plant J. 2016;86:349–359.

[67] Shental N, Amir A, Zuk O. Identification of rare alleles and their carriers using compressed se(que)nsing. Nucleic Acids Res. 2010;38(179).

[68] Beck A, Teboulle M. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. SIAM J Imaging Sci;2(1):183-202.

[69] Tipping M. Sparse Bayesian Learning and the Relevance Vector Machine. Journal of Machine Learning Research. 2001.

[70] Crespo Marques E, Maciel N, Naviner L, Cai H, Yang J. A Review of Sparse Recovery Algorithms. IEEE Access. 2019;7:1300-22.

[71] Goenka R, Cao SJ, Wong CW, Rajwade A, Baron D. Contact Tracing Information Improves the Performance of Group Testing Algorithms; 2021. https://arxiv.org/abs/2106.02699.

[72] Reed IS, Solomon G. Polynomial codes over certain finite fields. J Soc Ind Appl Math. 1960.

[73] Naidu RR, Jampana P, Sastry CS. Deterministic Compressed Sensing Matrices: Construction via Euler Squares and Applications. IEEE Transactions on Signal Processing. 2016;64(14):3566-75.

[74] Naidu RR, Murthy CR. Construction of Binary Sensing Matrices Using Extremal Set Theory. IEEE Signal Processing Letters. 2017;24(2):211-5.

[75] DeVore RA. Deterministic constructions of compressed sensing matrices. J Complexity. 2007;23(4):918-25.

[76] Candes E, Wakin M. An introduction to compressive sampling. IEEE Signal Processing Magazine. 2008.

[77] Bioglio V, Bianchi T, Magli E. On the fly estimation of the sparsity degree in compressed sensing using sparse sensing matrices. In: ICASSP; 2015. p. 3801–3805.

[78] Ravazzi C, Fosson S, Bianchi T, Magli E. Sparsity estimation from compressive projections via sparse random matrices. EURASIP J Adv Signal Process. 2018;56.

[79] Zhu J, Rivera K, Baron D. Noisy Pooled PCR for Virus Testing; 2020. https://arxiv.org/abs/2004.02689.

[80] Center for Disease Control and Prevention. Contact Tracing for COVID-19;. https://www.cdc.gov/coronavirus/2019-ncov/php/contact-tracing/contact-tracing-plan/contact-tracing.html.

[81] Case Investigation and Contact Tracing: Part of a Multipronged Approach to Fight the COVID-19 Pandemic; 2021. https://www.cdc.gov/coronavirus/2019-ncov/php/principles-contact-tracing.html.

[82] Hekmati A, Ramachandran G, Krishnamachari B. CONTAIN: Privacy-oriented Contact Tracing Protocols for Epidemics;. https://arxiv.org/abs/2004.05251.

[83] Kleinman R, Merkel C. Digital contact tracing for COVID-19. Canadian Medical Association Journal. 2020.

[84] Hohman M, McMaster F, Woodruff SI. Contact Tracing for COVID-19: The Use of Motivational Interviewing and the Role of Social Work. Clinical Social Work Journal. 2021.

[85] Ross AM, Zerden L, Ruth B, Zelnick J, Cederbaum J. Contact Tracing: An Opportunity for Social Work to Lead. Social Work in Public Health. 2020;35(7):533-45.

[86] Goenka R, Cao SJ, Wong CW, Rajwade A, Baron D. Contact Tracing Enhances the Efficiency of COVID-19 Group Testing. In: ICASSP; 2021. p. 8168-72.

[87] Yuan M, Lin Y. Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society, Series B. 2007;68(1):49–67.

[88] Jacob L, Obozinski G, Vert JP. Group LASSO with overlap and graph LASSO. In: Int. Conf. Mach. Learning; 2009. .

[89] Nikolopoulos P, Srinivasavaradhan SR, Guo T, Fragouli C, Diggavi S. Group testing for connected communities. In: AISTATS; 2021. .

[90] Nikolopoulos P, Srinivasavaradhan SR, Guo T, Fragouli C, Diggavi S. Group testing for overlapping communities. In: IEEE Int. Conf. Commun.; 2021. .

[91] Attia M, Chang W, Tandon R. Heterogeneity Aware Two-Stage Group Testing. IEEE Transactions on Signal Processing. 2021;69.

[92] Bilder CR, Tebbs JM, Chen P. Informative Retesting. J Am Stat Assoc. 2010;105(491):942-55.

[93] McMahan CS, Tebbs JM, Bilder CR. Informative Dorfman screening. Biometrics. 2012;68(1):287-96.

[94] Deckert A, Barnighausen T, Kyei NN. Simulation of pooled-sample analysis strategies for COVID-19 mass testing. Bulletin of the World Health Organization. 2020;98(9).

[95] Arasli B, Ulukus S. Group Testing with a Graph Infection Spread Model; 2021. https://arxiv.org/abs/2101.05792.

[96] Ahn S, Chen WN, Ozgur A. Adaptive Group Testing on Networks with Community Structure; 2021. https://arxiv.org/abs/2101.02405.

[97] Lendle SD, Hudgens MG, Qaqish BF. Group Testing for Case Identification with Correlated Responses. Biometrics. 2012;68:532–540.

[98] Lin YJ, Yu CH, Liu TH, Chang CS, Chen WT. Positively Correlated Samples Save Pooled Testing Costs; 2020. https://arxiv.org/abs/2011.09794.

[99] List of countries implementing pool testing strategy against COVID-19;. https://en.wikipedia.org/wiki/List_of_countries_implementing_pool_testing_strategy_against_COVID-19.

[100] IIT-Bombay professor comes up with new tool for Covid testing. It cuts time and cost; 2021. https://theprint.in/health/iit-bombay-professor-comes-up-with-new-tool-for-covid-testing-it-cuts-time-and-cost/684709/.

[101] Deka S, Kalita D. Effectiveness of sample pooling strategies for SARS-CoV-2 mass screening by RT-PCR: a scoping review. Journal of laboratory physicians. 2020;12(03):212-8.

[102] Gatta VL, Moscato V, Postiglione M, Sperli G. An Epidemiological Neural Network Exploiting Dynamic Graph Structured Data Applied to the COVID-19 Outbreak. IEEE Transactions on Big Data. 2021;7(1):45-55.