Heuristic Random Designs for Exact Identification of Positives Using Single Round Non-adaptive Group Testing and Compressed Sensing

Catherine A. Haddad-Zaknoon
dept. of Computer Science
Technion, Israel Institution of Technology
Haifa, Israel
email: catherine@cs.technion.ac.il

***Abstract*—Among the challenges that the COVID-19 pandemic outbreak revealed is the problem of reducing the number of tests required for identifying the virus carriers. To cope with this issue, a prevalence testing paradigm based on Group Testing and Compressive Sensing approach or *GTCS* was examined. In these settings, a non-adaptive group testing algorithm is designed to rule out sure-negative samples. Then, a compressive sensing algorithm is applied to decode the positives without requiring any further testing. The result is a single-round non-adaptive group testing - compressive sensing algorithm to identify the positive samples. In this paper, we propose a heuristic random method to construct the test design called $\alpha-$*random row design* or $\alpha-$RRD. In the $\alpha-$RRD, a random test matrix is constructed such that each test aggregates at most α samples in one group test or pool. The pooled tests are heuristically selected one by one such that samples that were previously selected in the same test are less likely to be aggregated together in a new test. We examined the performance of the $\alpha-$RRD design within the GTCS paradigm for several values of α. The experiments were conducted on synthetic data and sensitivity to noise was checked. Our results show that, for some values of α, a reduction of up to 10 fold in the tests number can be achieved when $\alpha-$RRD design is applied in the GTCS paradigm.**

***Index Terms*—Group Testing, Pooling Design, Compressive Sensing, COVID19-PCR**

I. Introduction

The problem of *group testing* is the problem of identifying a small amount of *items* or *subjects* known as *defective items* or *positive subjects* within a pile of elements using *group tests* or *pools*.

Denote the number of positive subjects by d and the total number of elements by n. A *group test* or a *pool* is a subset of subjects. A test result is *positive* if it contains at least one positive subject and *negative* otherwise. The objective of group testing algorithms is to find the set of positive subjects, denoted by I, with minimum number of group tests.

In this paper, we will examine *non-adaptive* group testing. In non-adaptive algorithms, tests are independent and must not rely on previous results. Therefore, all the tests can be performed in a single parallel step. The set of tests in any non-adaptive deterministic (resp. randomized) algorithm can be identified with an (resp. random) $m \times n$ *test design matrix* M (also called *pool design*) that its rows are all the assignments a that correspond to the group tests selected by the algorithm.

Group testing approach was first introduced during World War II [3], when Robert Dorfman, in 1943, suggested the method to reduce the expected number of tests needed to weed out all syphilitic soldiers in a specific unit. Among its recent applications, due to the recent pandemic outbreak, group testing approach for accelerating COVID-19 testing was widely applied across the globe. Due to severe shortages in testing kits supply, a number of researches adopted the group testing paradigm for COVID-19 mass testing not only to accelerate the testing process, but also to reduce the number of the tests required to reveal positive virus-carriers [4] [5] [6] [9] [12] [13] [16]. In many labs, COVID-19 detection was performed using *Polymerase Chain Reaction* tests or PCR tests for short. PCR-based machines can perform multiple parallel tests in single run, while each run can be several hours long. Driven by the process of PCR testing, non-adaptive group testing is most fit for these settings. In this context, the items in question are samples taken from potential patients and the *positive subjects* are samples that test positive to the virus.

While many researchers applied Dorfman's attitude with multi-stage PCR runs, some have examined designing single-PCR round tests instead. One of the promising directions is the Group Testing - Compressed Sensing paradigm (GTCS) used in [7] [8] [10] [14]. This method includes the following stages; initially, a test matrix M is designed for a single non-adaptive group testing round. Upon test results delivery, a two-stage decoding process is performed. The decoding process is purely combinatorial and does not involve any further sample testing. Using standard non-adaptive group testing decoding (e.g., Combinatorial Matching Pursuit or COMP algorithm [2]), a substantial amount of samples that tested negative to the virus are ruled out. Obviously, the main benefit of this phase is to reduce the dimension of the compressed sensing problem by cutting down the number of samples that need further decoding. This is crucial due to the

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-971-3

computational complexity of compressed sensing algorithms. In the next stage, compressive sensing techniques are used over the reduced problem (e.g., Orthogonal Matching Pursuit - OMP [15], Fast-OMP [11]), to identify real carriers.

The design of the test matrix M is crucial for both group testing phase and the compressive sensing phase that follows. In [14], the design matrix is constructed using Reed Solomon error correcting codes. The authors has checked their method on a set of $n = 384$ samples in which 5 samples are positive (about 1.3%). For pool size of 48 and using 48 group tests, they could recover all the 5 positive samples. In the work of Jirong, Mudumbai and Xu [10], the authors investigated two types of pooling designs. The first is Bernoulli random matrix where each entry is selected to be 1 or 0 with equal probability. The second design is obtained using expander graphs where each column has a fixed number of non-zero entries. The designs tested in [7] are based on Kirkman triples.

In this paper, we propose a heuristic random method to construct the test design M called $\alpha-$*random row design* or $\alpha-$RRD. In the $\alpha-$RRD, a random test matrix is constructed such that each test in M aggregates at most $\alpha < n$ samples in one group test. This model is useful in applications were tests reliability might be compromised if the pool size is large. We call α the *pool size*. The matrix rows are selected one by one. The main idea of the construction is to choose the non-zero entries of a new row according to two considerations: samples that belong to the same subject participate in similar number of tests on average (fairness); and samples that were previously selected in the same test are less likely to be aggregated together in a new test (sparsity). We perform experiments on noiseless and noisy synthetic data to examine the performance of the design, while applying Orthogonal Matching Pursuit or OMP as the compressive sensing algorithm. Practically, test designs need to be deterministic, meaning, they need to be predefined before the testing process. To use random test design, it is acceptable to make simulations of several random designs and choosing the design that performs best on some set of data. Then, this design is adopted to be used as a deterministic one for the real time tests.

The advantage of the $\alpha-$RRD design is first that it can be applied for any dimensions m and n. In many applications, the pool size is crucial for the accuracy of the testing process, therefore, it is highly recommended to use α as small as possible. In some applications there is an upper bound on the number samples that can be merged in one pool. Therefore, the $\alpha-$RRD design fits those settings when choosing α within the bounds of the pool size.

Our experiments results suggest that using the GTCS framework with $\alpha-$RRD design can reduce the number of tests dramatically. In the experiments, we tested the performance of the framework on designs with total number of tests $m = 96$ and the number of samples can be $n = 400, 600$ or 900. For each value of n, we tested on several pool sizes α that range between $\alpha = 12$ up to 48. The number of positives d ranges from 1 up to 20. For each n, α and d, we calculated the average error in restoring the positives subset over 200 random sets. The results imply that there is an evident correlation between the value of α and the performance of the process; choosing higher values of α can increase the success rate in identifying the positives. Moreover, the tests results show that the GTCS paradigm with the $\alpha - RRD$ matrix, can improve dramatically the total number of tests. In some settings, a $10-$fold improvement can be achieved compared to the single sample per test approach.

The paper is organized as follows. In Section II, we cover some definitions and preliminaries required for defining the problem of group testing and the compressive sensing in mathematical terms. Moreover, in this section, we define the group testing - compressive sensing (GTCS) paradigm. In Section III, we describe in details of the $\alpha-$RRD design and give a detailed algorithm for constructing such design. Section IV outlines experiments results designed to measure the performance of the $\alpha-$RRD design as part of the GTCS paradigm, and in Section V, we give some conclusions and future directions.

II. Definitions and Preliminaries

In this section, we define the mathematically of the problems of group testing and compressive sensing.

A. The group testing − GT problem

Let $X = [n] := \{1, \cdots, n\}$ be a set of n items or subjects, and let $I \subseteq X$ be the set of positive (defective) items such that $|I| = d \ll n$. A *group test* or a *pool* is a subset $Q \subseteq X$ of items. The quantity $\alpha := |Q|$ is called the *pool size*. The result of the test Q with respect to I is defined by $Q(I) := 1$ if $Q \cap I \neq \varnothing$ and $Q(I) := 0$ otherwise. Alternatively, we identify the test $Q \subseteq X$ with an *assignment* $a \in \{0, 1\}^n$ where $a_i = 1$ if and only if $i \in Q$.

The set of tests in any non-adaptive group testing algorithm can be identified with an $m \times n$ *test design matrix* M (pool design), where each row corresponds to an assignment $a \in \{0, 1\}^n$ that defines a group test selected by the algorithm. Upon performing the tests defined by M, each test of the m assignments in M yields the value 1 or 0 according to whether the tests contains at least one positive sample or not. Let $y \in \{0, 1\}^m$ denote the test results obtained by performing the tests of M, and let $x \in \{0, 1\}^n$ be a vector such that $x_i = 1$ if and only if $i \in I$. Formally,

$$y = M \odot x,$$

where the operation $\odot$ is defined as follows; for each $1 \leq i \leq m$,

$$y_i = \bigvee_{j=1}^{n} M_{i,j} \cdot x_j, \tag{1}$$

where the $\vee$ operation is the logic OR. It is easy to see that, the definition from (1) is equivalent to $y_i = 1$ if and only if $M_{(i)} \cap I \neq \varnothing$, where $M_{(i)}$ is the set that corresponds to the test defined by the ith row in M.

B. The compressive sensing − CS problem

Assume that each subject sample can be measured by a real valued number that expresses the *magnitude* or the *load* of the examined symptom (e.g., viral load in COVID-19 case). Let $\hat{x} \in R^n$ be a $n-$dimensional real-valued vector that signifies the symptom load of the subjects; i.e., for each $1 \leq i \leq n$, $\hat{x}_i$ indicates the symptom load of the subject i, where the value of $\hat{x}_i$ is directly proportional to the load. Symptom-free items will have their corresponding load measure equals to 0. We assume that the number of positives $d \ll n$, therefore, the load vector $\hat{x}$ is $d-$sparse; it includes only d non-zero entries. The objective is to restore the indexes of the non-zero entries in $\hat{x}$.

Similar to the definition of the result vector y from the GT settings, the design matrix M, also called the *sensing matrix* in the compressive sensing context, defines the load vector $\hat{y} \in R^n$ where each entry $\hat{y}_i$ correlates with the load of the ith pool in M. That is,

$$\hat{y} = M \cdot \hat{x}, \tag{2}$$

where the $(\cdot)$ operation is the standard matrix multiplication, therefore, for each $1 \leq i \leq m$,

$$\hat{y}_i = \sum_{j=1}^{n} M_{i,j} \cdot \hat{x}_j. \tag{3}$$

In this paper, we are interested in restoring the indexes of the non-zero entries of the vector $\hat{x}$, which is equivalent to restoring the binary vector x from the GT settings.

Formally, to find solutions for (2), we consider the following optimization problem (P_0):

$$\min_{\hat{x}} \quad \|\hat{x}\|_0 \quad \text{s.t.} \qquad \hat{y} = M \cdot \hat{x} \tag{4}$$

where $\|x\|_0$ denotes the zero norm, L_0, which is defined as the number of non-zero entries in x. The problem in (4) is NP-Hard. The main difficulty in solving (P_0) is that the constraint is highly non-smooth due to the L_0 penalty. Therefore, some relaxations are considered for approximating the solution. Even if the problem (P_0) has a unique solution, for slightly perturbed vector $\hat{y}$, the system $M \cdot \hat{x} = \hat{y}$ will no longer have a sparse solution as desired (a solution with at most d non-zero entries). Moreover, the L_0 measure is strict, and a small random noise in $\hat{x}$ causes the solution of (P_0) to be fully dense. To cope with these two problems, the following alternative problem (P_0^ϵ) is considered:

$$\min_{\hat{x}} \quad \|\hat{x}\|_0 \quad \text{s.t.} \qquad \|M \cdot \hat{x} - \hat{y}\|_2^2 \leq \epsilon^2. \tag{5}$$

It is well known that the $\epsilon-$deviation in the constraint in (P_0^ϵ) overcomes the two difficulties. Therefore, compressive-sensing algorithms designed to solve the problem (P_0^ϵ) have inherent robustness for noise. The OMP algorithm finds the best approximation for the solution of (5) using a greedy attitude. Other methods relax the L_0 norm via the L_1 norm. In our experiments, we choose $\epsilon = 10^{-3}$.

C. The group testing - compressive sensing paradigm - GTCS

The GTCS paradigm suggests a non-adaptive group testing generic algorithm for identifying the exact set of positives while using compressive-sensing based decoding techniques. The GTCS paradigm is composed of three basic phases.

1) **Create and perform the actual tests:** Create a test design M and perform the group tests defined by the design. Practically, the outcome of this stage is a vector $\hat{y} \in R^n$ as described in (2) and (3). This stage is followed by two-stage decoding process to exactly identify the test of positives. The vector y is derived from $\hat{y}$ by assigning each entry $y_i = 1$ if $\hat{y}_i > 0$, and $y_i = 0$ otherwise.
2) **Group testing decoding:** using standard group testing decoding methods (e.g., Combinatorial Maching Pursuit or COMP algorithm [2]) on the problem $y = M \odot x$, a subset $X_0 \subseteq X$ of items that are guaranteed by the GT algorithm to be negative samples is identified. This stage is used to reduce the size of the problem to be solved in the next stage. The rational behind this step is to exploit the fact that GT decoding algorithms like COMP has zero false negatives (i.e. all sample that were detected by the algorithm as negative ones are actually negative). Therefore, eliminating the set of sure-negative samples X_0 reduces the computational complexity of the step that follows, while keeping its decoding accuracy intact.
   The reduced compressive-sensing problem is established by applying the following enhancements. Given the set X_0 that includes the sure negatives, we define a new set $X_r := X \setminus X_0$. Let $Y_0 \subseteq [m]$ be the set of tests indexes that yielded the result 0 in the previous stage. The new test design matrix M_r is constructed from M, X_0 and Y_0 by projecting M on the columns that correspond to the samples in $X \setminus X_0$ and the rows that appear in the set $[m] \setminus Y_0$. Therefore, the resulting matrix is an $(m_r \times n_r)$ binary matrix where $m_r = m - |Y_0|$ and $n_r = n - |X_0|$. The reduced test result vector $\hat{y}_r$ is derived from $\hat{y}$ by deleting the entries that correspond to Y_0.
3) **Compressive sensing decoding:** by applying standard compressive sensing algorithms (e.g., OMP) on the reduced problem $\hat{y}_r = M_r \cdot \hat{x}_r$, and using the results from previous stage, the vector $\hat{x}_r$ and therefore, the vectors $\hat{x}$ and x can be restored.

III. Random Row Design - $\alpha - RRD$

In this section, we propose a random design for GT. For this design, we restrict the pool size to be at most $\alpha < n$. The design is constructed row by row. The main idea of the construction is to choose the non-zero entries in the new row according to two principles;

1) **Fairness:** Elements that participated in the minimum number of tests in previous rows, will be more likely to be chosen in the new test.

2) **Sparsity**: Elements that were previously selected in the same test, will be less likely to be assembled together in the new test.

For a vector $a = (a_1, \ldots, a_n) \in \{0,1\}^n$, recall that $\mathcal{H}(a) := \{i : a_i = 1\} \subseteq [n]$. The set $\mathcal{H}(a)$ is also called *the support of* a. The *Hamming weight* of a is denoted by $\omega(a)$ and is equal to $\omega(a) = |\mathcal{H}(a)|$. Let $\mathbf{0}^n$ ($\mathbf{1}^n$, ∞^n) denote the all zero (one, ∞ resp.) vector of length n. Let $A_{(m\times n)}$ be an $m \times n$ matrix over $\{0,1\}$. For all $1 \leq i \leq m$, denote by $A_{(i)} \in \{0,1\}^n$ the ith row of the matrix A, by $A^{(j)}$ the jth column and by $A_{i,j}$ the element in A that corresponds to the ith row and the jth column. The *columns weight vector* of a matrix $A_{m\times n}$ is a vector $w = (w_1, \ldots, w_n) \in R^n$ such that $w_j = \sum_{i=1}^{m} A_{i,j}$. Practically, the weight column vector indicates the number of tests each element participated in.

The procedure $\mathbf{RRD}(n, m, \alpha)$ from Fig. 1 describes the RRD strategy to choose a random design A with m rows and n columns where each row is of Hamming weight at most α. The algorithm starts by randomly choosing the first row in A from the set of binary vectors of length n and Hamming weight α. Assume that the first $\ell-1$ rows are already chosen, and let $A_{\ell-1}$ be the matrix defined by those rows. Let $\hat{w} = (w_1, \ldots, w_n) \in R^n$ be the columns weight vector of $A_{\ell-1}$. Then, the algorithm chooses the first non-zero entry in the ℓth row uniformly randomly from the set of indexes that correspond to the entries of minimal value in $\hat{w}$. This choice complies with the fairness principle.

Let $k < \alpha$ be the number of non-zero entries that algorithm already chose for the ℓth row. The $k+1$ entry is chosen as follows. Let Q_k be the set of indexes of the non-zero entires chosen so far in the current row. Let Z be the set of rows indexes i, such that $\mathcal{H}(A_{(i)}) \cap Q_k \neq \varnothing$, and let $\hat{w}$ be the weight vector of submatrix of A defined by the rows in Z. The algorithm evaluates $\hat{w}$ in steps (8) and (9) in Fig. 1. Then, the algorithm constructs the set of indexes $S \subseteq [n]$ that includes all the indexes j such that w_j is of minimum value among the entries in $\hat{w}$ and sums the corresponding columns. Let X be the set of column indexes with minimum value. Then, among the indexes in X, choose $s \in X$ uniformly at random and assign $A_{\ell,s} = 1$. These are steps (10) $-$ (16) in the algorithm in Fig. 1. This choice complies with the fairness principle.

Fig. 4 describes the procedure $\mathbf{CalcSelectedRows}$. The procedure outputs a set of row numbers $C \subseteq \{1, \ldots, \ell - 1\}$ such that $i \in C$ if and only if $\mathcal{H}(A_{(i)}) \cap Q_k \neq \varnothing$. The procedure $\mathbf{UpdateWeight}$ calculates the weight vector $w = \sum_{j\in C} A_{(j)}$ (See step no. 2 in Fig. 2). To ensure that the entries in Q_k are excluded from the selection of the next non-zero index, the weight vector w is updated to have the value ∞ in the corresponding indexes. (See step 8 in Fig. 2).

The selection of the set S complies with the sparsity principle. The set S is derived from the weight vector $\hat{w}$ by selecting the indexes with minimal weight, where the weight is evaluated over the rows that agree with one or more of the entries selected for the current test. The initialization step in $\mathbf{SumColumns}$ implies that the choice of the next non-zero entry of the current test will be from the indexes in S (See Fig. 3). Those indexes are the ones with minimum agreement with the current test, therefore, choosing the next non-zero entry from them is the best choice to keep the sparsity principle. The selection of the set X in step 14 of the algorithm in Fig. 1 complies with the fairness principle; among the best candidates from the indexes of S, the algorithm chooses s uniformly from those that appeared minimum number of times over all the previous tests.

The time complexity of generating $\alpha-$RRD design is polynomial in the dimensions of the design and can be easily generated for any dimensions m and n.

Procedure: m, n, α
Output: An $m \times n$ design matrix A
1: $A \leftarrow \{0\}_{(m\times n)}$.
2: Choose $a \in \{0,1\}^n$ uniformly at random from all vectors of weight α.
3: $A_{(1)} \leftarrow a$.
4: **for** $\ell = 2$ to m **do**
5: $k \leftarrow 0, Q_0 \leftarrow \{\}$
6: **while** $k < \alpha$ **do**
7: $k \leftarrow k + 1$.
8: $C \leftarrow \mathbf{ClacSelectedRows}(n, Q_{k-1}, A, \ell - 1)$
9: $\hat{w} \leftarrow \mathbf{UpdateWeight}(n, Q_{k-1}, C, A)$
10: $\hat{w}_{\min} \leftarrow \min_{1\leq j\leq n} \hat{w}_j$
11: $S \leftarrow \{p : \hat{w}_p = \hat{w}_{\min}\}$
12: $\hat{z} \leftarrow \mathbf{SumColumns}(n, A, \ell - 1, S)$
13: $\hat{z}_{\min} \leftarrow \min_{1\leq j\leq n} \hat{z}_j$
14: $X \leftarrow \{t : \hat{z}_t = \hat{z}_{\min}\}$
15: Select s uniformly at random from X.
16: $Q_k \leftarrow Q_{k-1} \cup \{s\}$.
17: $A_{\ell,s} \leftarrow 1$
18: **end while**
19: **end for**
20: Return A.

Fig. 1: The procedure $\mathbf{RRD}(m, n, \alpha)$

Procedure: $\mathbf{UpdateWeight}(n, Q, C, A)$
Output: An updated weight vector w
1: **if** $C = \varnothing$ **then**
2: $w \leftarrow 1^n$
3: **else**
4: $w \leftarrow \sum_{j\in C} A_{(}j)$
5: **end if**
6: **for** $i = 1$ to n **do**
7: **if** $i \in Q$ **then**
8: $w_i \leftarrow \infty$
9: **end if**
10: **end for**
11: Return w

Fig. 2: The procedure **UpdateWeight**

Procedure: n, A, ℓ, S.
Output: Sum all columns in A
1: $w \leftarrow \infty^n$
2: **for** each $i \in S$ **do**
3: $\quad w_i \leftarrow \sum_{j=1}^{\ell} A_{j,i}$
4: **end for**
5: Return w

Fig. 3: The procedure **SumColumns**(n, A, ℓ, S)

Procedure: n, A, ℓ, S.
Output: Calculate selected rows
1: $C \leftarrow \varnothing$
2: **for** each $i = 1$ to ℓ **do**
3: $\quad$ **if** $(\mathcal{H}(A_{(i)}) \cap Q \neq \varnothing)$ **then**
4: $\quad\quad C \leftarrow C \cup \{i\}$
5: $\quad$ **end if**
6: **end for**
7: Return C

Fig. 4: The procedure **CalcSelectedRows**(n, Q, A, ℓ)

IV. Experiments and Simulations

In this section, we outline tests results of the performance of the $\alpha-$RRD design when chosen as the test design in the GTCS generic paradigm. For the GT decoding, the COMP algorithm is selected to generate initial sure-negative set, while the OMP is used as the CS algorithm in the final stage.

A. Data generation

We test the performance of the $\alpha-$RRD design on both synthetic noisy and noiseless data, where an $\alpha-$RRD design matrix is generated for the dimensions $m = 96$ and $n = 400, 600$ and 900. Despite the fact that the construction from section III does not impose any limitation on the parameter m, the choice of the value of m in the experiments is derived from the number of tests that can be performed in parallel in most PCR machines used in the industry. We examine several values of α starting from $\alpha = 12$ up to 48. The minimum choice of α is derived from the applicability of the compressive sensing algorithm on the constructed matrix, while the maximum value of the pool size matches the maximum value tested for COVID-19 PCR pool designs [14]. The number of positive subjects d ranges from 1 up to 20. For each choice of n and α, an $m \times n$ $\alpha-$RRD matrix M was randomized according to the algorithm from Fig. 1 for several values of α. For each value d, we randomized 200 vectors $\hat{x} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_n) \in \Re^n$ with d non-zero entries that signify the symptom load in the positive samples amongst the n samples. The vector $\hat{x}$ is chosen where the d non-zero entries are chosen uniformly at random, while the symptom load of each non-zero entry is chosen uniformly over the real range $[1, 2]$.

For each realization $\hat{x}$, the test result $\hat{y}$ is generated according to $\hat{y} = M \cdot \hat{x}$. In the noisy settings, a random noise vector v with energy $\|v\|_2 = 10^{-3}$ was added to $\hat{y}$ to generate a noisy version of $\hat{y}$, denoted by $\tilde{y} = \hat{y} + v$ (here, $\|\cdot\|_2$ denotes the L_2 norm of the vector v). In the noiseless case, we have $\tilde{y} = \hat{y}$. Given the design matrix M and $\tilde{y}$, we use the COMP algorithm followed by OMP to restore the support of $\hat{x}$. Denote by $\tilde{x}$ the result calculated after the OMP phase. Let S be the true support of $\hat{x}$, i.e. $S = \{i | \hat{x}_i > 0\}$, and $\tilde{S}$ be the support of $\tilde{x}$. The *support recovery error* is defined as

$$1 - \frac{|S \cap \tilde{S}|}{\max\{|\tilde{S}|, |S|\}}.$$

B. Tests results

Our experiments results suggest that using the GTCS framework with $\alpha-$RRD design dramatically reduces the number of tests. Fig. 5 shows the average support recovery error over all the 200 trials for each value of d and α for the noiseless case when $n = 400, 600$ and 900, where the number of positive samples is up to $d = 20$. In all these settings, an $\alpha-$RRD design matrix with total number of tests $m = 96$ is selected. For $n = 400$ and positives rate near 2.5% $(d = 10)$, the average error in restoring the correct support is less than 0.005 when α approaches 20. Moreover, the error drops to 0 for positives rate 1.5% $(d = 6)$ for $\alpha = 16$. This is $4-$fold improvement compared to the single test per sample settings. For $n = 900$ and positives rate up to 1% $(d = 9)$, for $\alpha = 48$ the error probability is less than 0.06. When the rate is 0.5% and $\alpha = 48$, the error probability drops to near 0 value. (See Fig. 5.(c)). This is $10-$fold improvement over single-test per sample method.

The results of the noiseless settings are reproduced also for the noisy case. For example, Fig. 6.(a) shows the average support recovery error over all the 200 trials for each value of d and α for the noisy case, when $n = 400$. It can be noticed that, for the same settings of the noiseless case, i.e, $\alpha = 20$ and $d = 10$, the error is bellow 0.005, and reaches 0 for $d = 6$ and $\alpha = 16$. For $n = 900$ and positives rate up to 1%, for $\alpha = 48$, the error probability is less than 0.06 and for positive rate 0.56% $(d = 5)$ the error rate is near 0 (See Fig. 6.(c)).

Moreover, the results imply that there is a correlation between the value of α and the performance of the process; choosing higher values of α can decrease the average error in identifying the positives. For example, in the noisy case, Fig. 6 shows that for $n = 400$ and $d = 10$ $(p = 2.5\%)$, the average error for $\alpha = 12$ is greater than 0.1 while it can be decreased bellow 0.005 for α between 20. For $n = 600$, and $p = 1\%$, when $\alpha = 16$, the error is about 0.069 while it drops to less than 0.001 when $\alpha = 44$. Similarly, for $n = 900$, $p = 0.56\%$ and $\alpha = 48$, the error probability is near zero, while for $\alpha = 20$, the error is higher than 0.1. This paradigm is reproduced in both noisy and noise free case too.

In practice, deciding the best value for α for the problem in-hand can be done while taking in consideration the limitations of the test process (for example, if there is some upper bound on the pool size). Once such limitations on the value of α are known, we can use computer simulation on synthetic data, similar to the ones described in this work, to decide on the best choices of α for each settings.

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-971-3

The results in Fig. 5 and Fig. 6 show that the error rate increases with d. This behavior is as expected from any group testing - compressive sensing algorithm, since those are designed to be used when the solutions x and $\hat{x}$ of the equations (1) and (2) are sparse vectors, meaning the number of the non-zero entries d is very small relative to n, more precisely when $d = O(\sqrt{n})$.

It is worth noticing that, the differences in the noisy case vs. the noise-free case are almost negligible. This behavior can be explained by two major factors. First, group testing decoding algorithms like COMP used in our simulations, are known for their robustness for *false negatives* (a false-negative is a sample classified by the algorithm as negative, but it is actually positive). That is, the set of samples classified by the GT algorithm as sure-negatives and therefore excluded from the decoding of the CS algorithm, does not include false-negatives. Therefore, it is highly unlikely to miss positive samples during the GT initial classification. The second factor is the noise-tolerance of the compressive sensing algorithm. Specifically, the OMP algorithm is known for its high accuracy in the presence of noise, while its drawback is its computational complexity.

V. Conclusion

In this paper, we suggested a new random pooling design $\alpha-$RRD. This design can be used as part of the GTCS paradigm in order to build a single-round non-adaptive group testing protocol to exactly identify positives within a large set of elements. The complexity of generating $\alpha-$RRD design is polynomial in the dimensions of the design and can be easily generated for any dimensions m and n. By its design, the $\alpha-$RRD pooling matrix is designed to restrict the size of the pool α which might be critical for test accuracy. If there is no practical restrictions on the size of α, then, given the parameters m, n and positives rate, the best choice for the parameter α can be concluded using computer simulations. Moreover, since random sensing matrices can perform well with compressive-sensing algorithms, the GTCS paradigm can be further tested with other well-known group testing random designs such as RID, RrSD, RsSD and Transversal design [1]. Similarly, other compressive sensing algorithms can be applied too. Besides being tested on synthetic data, it is worth examining the efficiency of the method and the design on real COVID-19 data or any other disease that follow the same paradigm.

References

[1] N. H. Bshouty, G. Haddad and C. A. Haddad-Zaknoon, "Bounds for the number of tests in non-adaptive randomized algorithms for group testing," SOFSEM 2020: Theory and Practice of Computer Science, Lecture Notes in Computer Science, Springer, vol. 12011, pp. 101–112, 2020.

[2] C. L. Chan, S. Jaggi, V. Saligrama and S. Agnihotri, " Non-adaptive group testing: Explicit bounds and novel algorithms," 2012 IEEE International Symposium on Information Theory Proceedings, pp. 1837-1841, 2012.

[3] R. Dorfman, "The detection of defective members of large populations," The Annals of Mathematical Statistics, vol. 14(4), pp. 436-440, 1943.

[4] A. Eis-Hübinger et al., 'Ad hoc laboratory-based surveillance of SARS-CoV-2 by real-time RT-PCR using minipools of RNA prepared from routine respiratory samples," Journal of clinical virology : the official publication of the Pan American Society for Clinical Virology, vol. 127, 104381, 2020.

[5] J. Cabrera et al., "Pooling for SARS-CoV-2 control in care institutions," BMC infectious diseases, vol. 20(1), 745, 2020.

[6] R. Ben-Ami et al., "Large-scale implementation of pooled rna extraction and rt-pcr for sars-cov-2 detection," Clinical microbiology and infection : the official publication of the European Society of Clinical Microbiology and Infectious Diseases, vol. 26(9), pp.1248–1253, 2020.

[7] S. Ghosh et al., "A Compressed Sensing Approach to Pooled RT-PCR Testing for COVID-19 Detection," IEEE Open Journal of Signal Processing, vol. 2, pp. 248-264, 2021.

[8] S. Ghosh et al., "Tapestry: A single-round smart pooling technique for covid-19 testing," Unpublished.

[9] C. Gollier and O. Gossner, "Group testing against covid-19.," Covid Economics, vol. 1, pp. 32–42, April 2020.

[10] 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,", unpublished.

[11] M. Yaghoobi, D. Wu and M. E. Davies, "Fast non-negative orthogonal matching pursuit," IEEE Signal Processing Letters, vol. 22(9), pp. 1229–1233, 2015.

[12] C. Mentus, M. Romeo, and C. DiPaola, "Analysis and applications of adaptive group testing methods for covid-19.," medRxiv, https://www.medrxiv.org/content/early/2020/04/07/2020.04.05.20050245, 2020.

[13] H. Shani-Narkiss, O. Gilday, N. Yayon, and I. Landau., "Efficient and practical sample pooling for high-throughput pcr diagnosis of covid-19," medRxiv, https://www.medrxiv.org/content/early/2020/04/14/2020.04.06.20052159, 2020.

[14] N. Shental et al., "Efficient high throughput sars-cov-2 testing to detect asymptomatic carriers," Science Advances, vol. 6, pp. eabc5961, 2020.

[15] Y. C. Pati, R. Rezaiifar and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," Proceedings of 27th Asilomar Conference on Signals, Systems and Computers, vol. 1, pp. 40–44, 1991.

[16] I. Yelin et al., "Evaluation of covid-19 rt-qpcr test in multi-sample pools," Infectious Diseases Society of America, vol. 71(16), pp. 2073–2078, 2020.

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-971-3

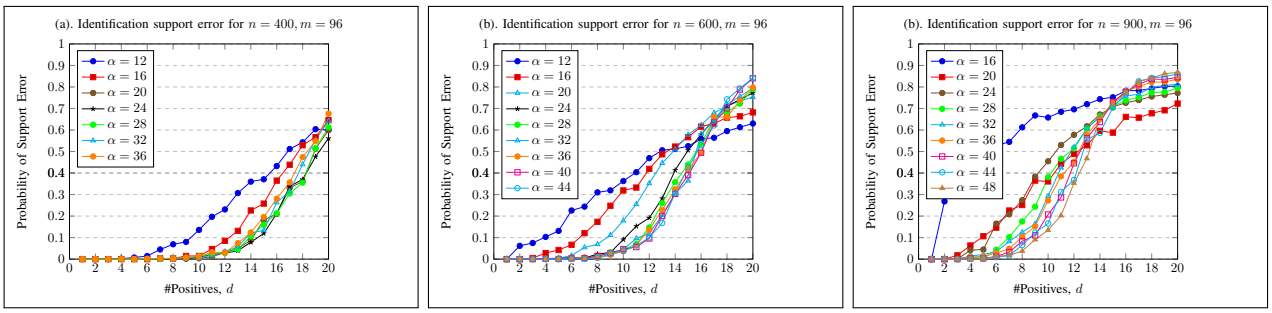

Fig. 5: Probability of support error for d up to 20, $n = 400, 600$, and 900, $m = 96$ and $\alpha \leq 48$ for the noise-free case.

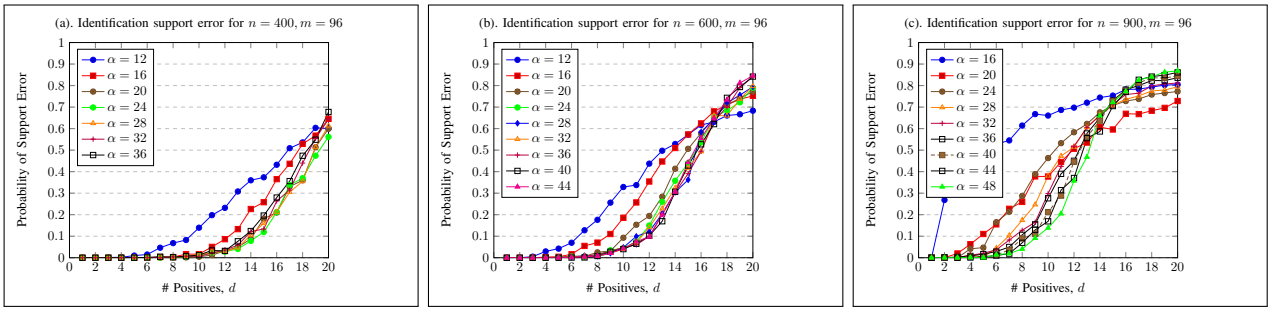

Fig. 6: Probability of support error for d up to 20, $n = 400, 600$, and 900, $m = 96$ and $\alpha \leq 48$ for the noisy case.

Copyright (c) IARIA, 2022. ISBN: 978-1-61208-971-3