# RT-PCR POOLING ANALYSIS

Loay Rashid
Jayant Duneja

# PAPERS:

The following papers have been assigned to us:

- Evaluation of Group Testing for SARS-CoV-2 RNA (Matrix Method)
- Pooling RT-PCR or NGS samples has the potential to cost-effectively generate estimates of COVID-19 prevalence in resource limited environments (Binary Search)
- Efficient and Practical Sample Pooling for HighThroughput PCR Diagnosis of COVID-19 (Shannons Theorem)
- Efficient sample pooling strategies for COVID-19 data gathering (Fisher Information matrix)
- Noisy Pooled PCR for Virus Testing (GAMP)
- On Accelerated Testing for COVID-19 Using Group Testing (Info Theory)
- Group Testing for COVID-19: How to Stop Worrying and Test More

# Evaluation of Group Testing for SARS-CoV-2 RNA (Matrix Method)

The paper presents an adaptive testing strategy to test large populations of low risk individuals.

The paper assumes that the testing is perfectly noise-less, and has not accounted for false positives/negatives, DNA mixing, not enough genetic material, etc. Moreover, the testing method becomes inefficient when COVID19 prevalence rises above 10%. Also, since it is a partially adaptive testing method it takes more time as compared to individual testing/nonadaptive.

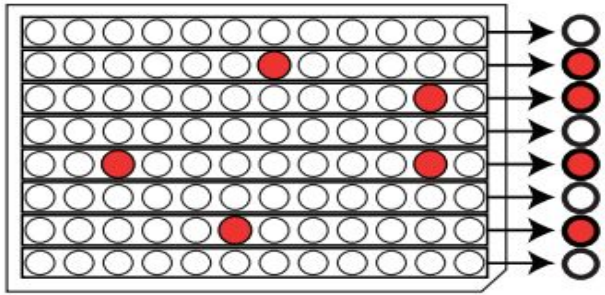
“When most tests are negative, pooling reduces the total number of tests up to four-fold at 2% prevalence and eight-fold at 0.5% prevalence”

On average, it is recommended to use 96-well matrices. However the performance for multiple pooling strategies has been simulated in the graph. Different prevalence rates=different matrix sizes.
This is a very bare bones paper, very little mathematics is needed here.

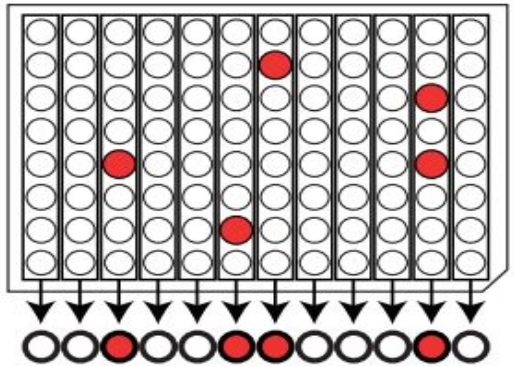
Each well contains a single patient sample

● : samples from positive cases

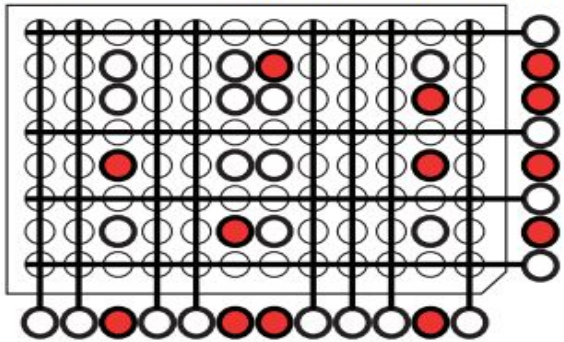
1. Test mixtures of all the samples in each row (8 tests)...



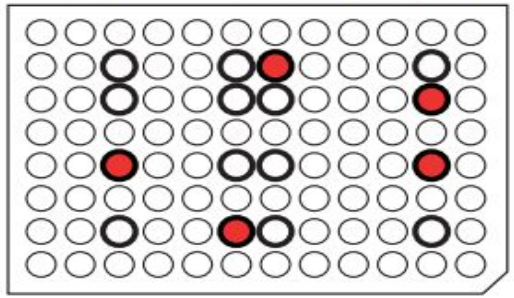
...and mixtures of all the samples in each column (12 tests)



2. Exclude negative rows and columns



3. Test each remaining sample (16 tests)



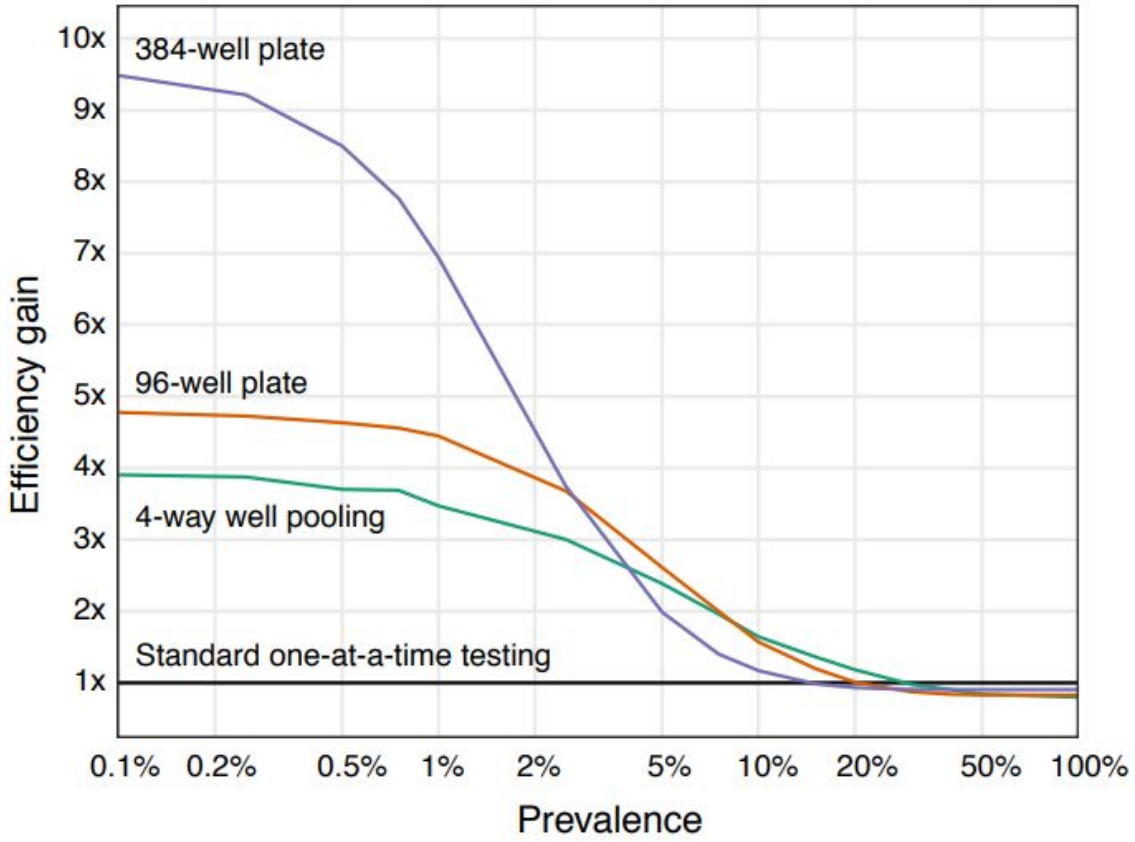



Figure 2: Evaluation of mean improvement of the pooling strategies relative to naive testing under different predicted prevalence. At 2% prevalence and below, row and column pooling on 384-well plates performs best; between 2% and 10%, row and column pooling on 96-well plates performs best; and at greater than 10%

# Pooling RT-PCR or NGS samples has the potential to cost-effectively generate estimates of COVID-19 prevalence in resource limited environments (Binary Search)

The aim of the paper is to classify infection rates in a given population as high or low (NOT INDIVIDUAL INFECTION DETECTION!!)

RT-PCR pooling was effective for malaria and NAAT was effective for malaria, prompting pooling to be applied to COVID 19

The paper presents a very inexact (hand-wavey) process: the threshold values and ideal group sizes are to be chosen without any reasoning acc to the paper (V has been arbitrarily given to be 1). Moreover, the probabilities of false alarm and detection etc are not well defined

In worst case scenario you will not do a lot of tests, you will declare high rate of infection and move on

Graphs have been copied from the information theory paper, and all the information here has been given in more detail and more mathematically in the other papers.

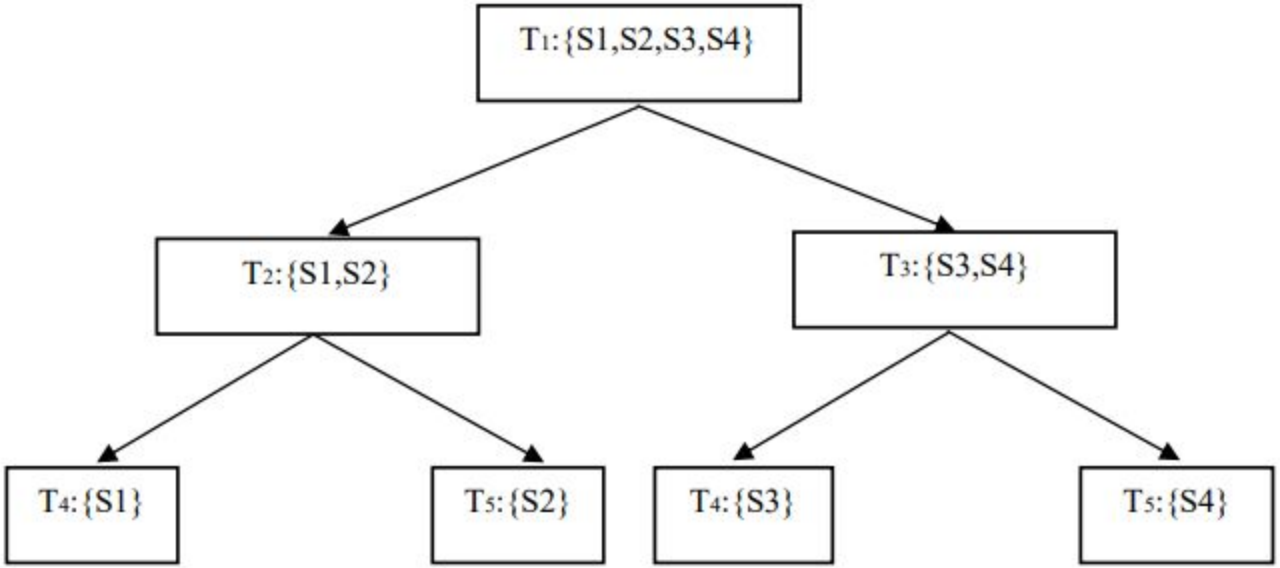



**Figure 1. Flowchart showing the binary splitting procedure.**

**Table 1 shows the number of tests required to detect a given infection rate.**

| Number of infected subpools | Infection rate | Number of tests required (*T*) |
|---|---|---|
| 0 | 1% (low) | 1 |
| 1 | 1% (low) | 5 |
| 2 | 5% (high) | 3 or 5 |
| 3 | 5% (high) | 3 |
| 4 | 5% (high) | 3 |

# Efficient and Practical Sample Pooling for High Throughput PCR Diagnosis of COVID-19 (Shannon's Theorem)

The main problem this paper tries to address is: what is the optimal value of batch size given a prevalence rate and testing strategy in a population. Most other papers assume the value of batch size to be a variable and then proceed, this paper goes in detail to figure out the batch size. Moreover, this paper applies Shannon's theorem to find the minimum possible limits on tests.

Two testing strategies have been considered:

1) Determine optimal pool size and then do binary search on each positive pool.
2) Determine optimal pool size and do one time testing, then do individual tests on each positive pool.

The first case is more efficient for extremely small values of p (prevalence)/asymptotic populations, the second method is more efficient for larger values of p. This will be demonstrated later.

**TESTING METHOD 1:**

The most efficient binary test is that which gives us the most information about the testing subject. To get the maximum information, the information entropy has to be maximum. For information entropy to be maximum, we want the probability of either test result (in this case only 2 possible results) to be equal, ie, half. We wish the probability of getting a positive or negative to be half each. This is a restatement of Shannons Source Coding Theorem.A similar intuition can be used to think about games like "Guess Who" and "Twenty Questions".

Let the expected frequency of infection (prevalence rate) be p. Let b be the batch size.
The probability of getting a negative result in a batch of size b is given by:

$$(1-p)^b,$$

We want this probability to be ½. Equating this expression to ½ and solving for b, we get:

$$b = -\frac{\log(2)}{\log(1-p)}$$

This is our optimal batch size for a given prevalence rate assuming we use method 1 of testing. We round this off to the nearest power of 2.
The number of tests is given by:

$$N_{tests} \approx \frac{N}{b}(1.3\log_2 b + 0.4)$$

In all subsequent rounds of testing, simply divide any positive batches in half and repeat, essentially performing a binary tree search algorithm.

**METHOD 2:**

We calculate the optimal batch size once, and individually test all the positive pools henceforth.

If N is the total population size,and b is the batch size

We will perform N/b tests in the first round of testing.

The probability of a pool testing positive is given by:

$$(1-(1-p)^b).$$

Therefore the total number of tests performed will be:

$$N_{tests} = N\left(\frac{1}{b} + 1 - (1-p)^b\right)$$

This function is then minimised to get the ideal value of b.

The paper says: “The above expression can be optimized numerically to find the optimal b given p.”

## THE LOWER BOUND ON THE NUMBER OF TESTS BY SHANNON'S SOURCE CODING THEOREM

Each test result can be mapped as a bernoulli random variable with probability p. The information entropy of each random variable is given by:

$$H = -p \log_2 p - (1-p) \log_2 (1-p)$$

By the data compression limit, you cannot compress N random variables with entropy H into less than NH bits. N multiplied with the expression above gives us the absolute bound on the number of tests. You cannot reduce the number of tests beyond this limit. (Explain using vector concept)

**UNCERTAINTY CONSIDERATIONS:**

- The total number of infected samples is given by Np with a variance of 1/N.
- Imperfect knowledge of p: in case we do not know the exact prevalence in a given population.

The effect these uncertainties have upon our testing efficiency is demonstrated in graphs below.
Practically, you estimate a value of p and see which experiment gives you the minimum value of Ntests.
^Here we are doing the opposite of what we have done throughout the paper.

**Table 1. One-Time Pooling**

| Range of Ratios | Range of p | Optimal Batch Size | Fraction of Tests Needed |
|---|---|---|---|
| <1:5 | $0.04 < p < 0.2$ | 4 | 0.40 - 0.84 |
| <1:25 | $0.008 < p < 0.04$ | 8 | 0.19 - 0.40 |
| <1:125 | $0.003 < p < 0.008$ | 16 | 0.11 - 0.18 |
| <1:333 | $0.001 < p < 0.003$ | 24 | 0.07 - 0.11 |
| <1:1000 | $0.0005 < p < 0.001$ | 32 | 0.05 - 0.06 |
| <1:2000 | $p < 0.0005$ | 64 | < 0.05 |

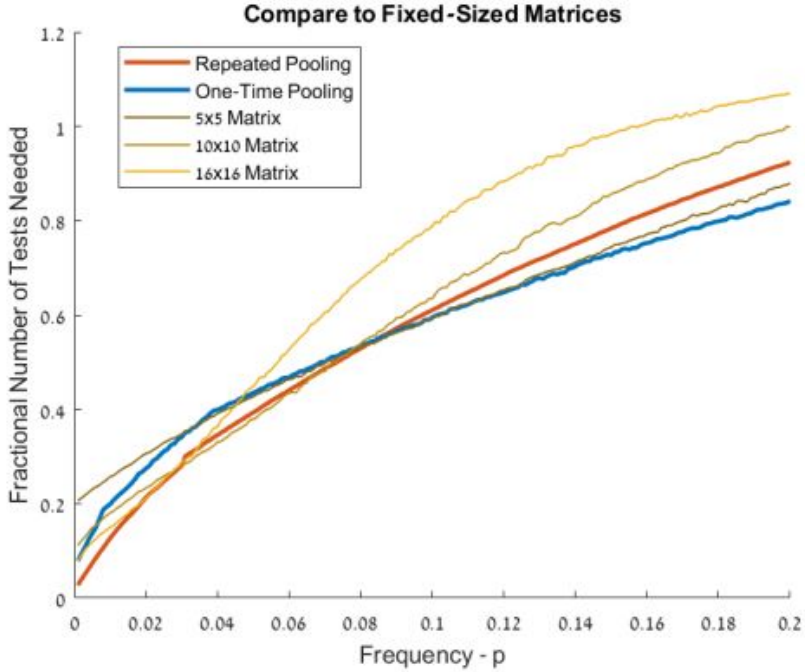
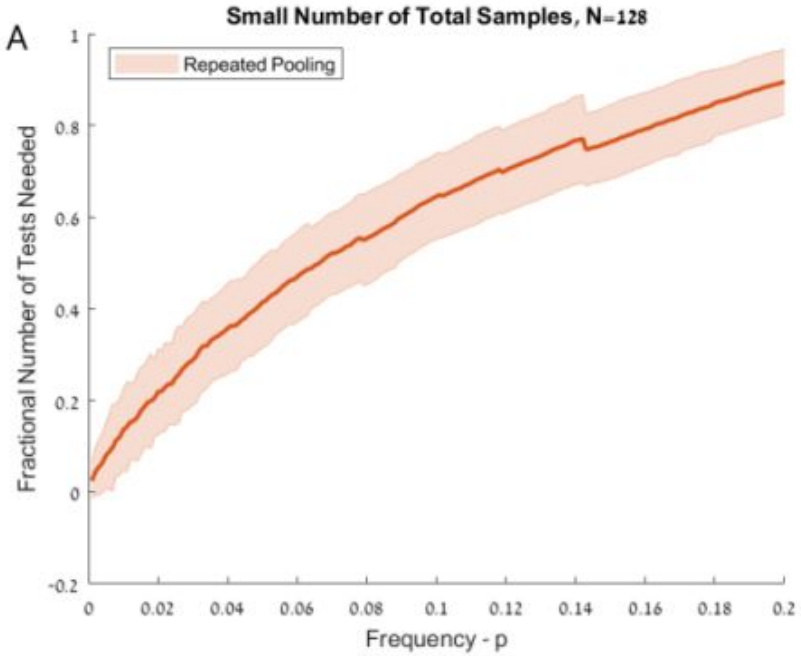



**Figure 3. Comparison to Fixed-Size Matrix Method in Realistic Setting.** We apply both methods presented here as they would be employed in a realistic setting with incomplete knowledge of *p*, and limited capacity for complex protocols. For one-time pooling we use three possible batch sizes: 4, 8 and 16. For repeated pooling we use four batch sizes: 8, 16, 32 and 64, with ranges of *p* assigned according to Tables 1 and 2, respectively. We compare the results to the matrix method, which uses 2D positional information, as described in the main text. No single method dominates all others. Note, however, that repeated pooling and one-time pooling can be easily combined into a single protocol with a single threshold value for *p*.



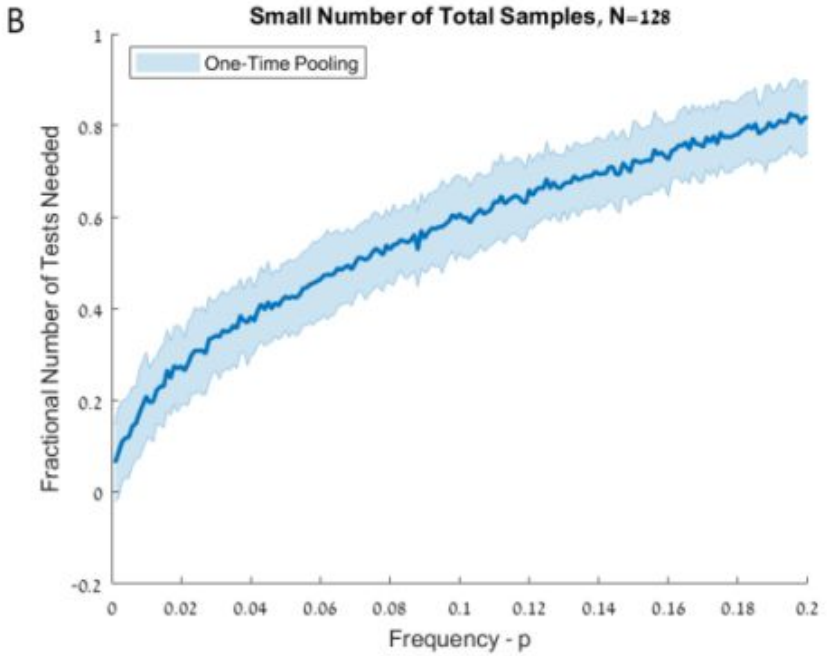
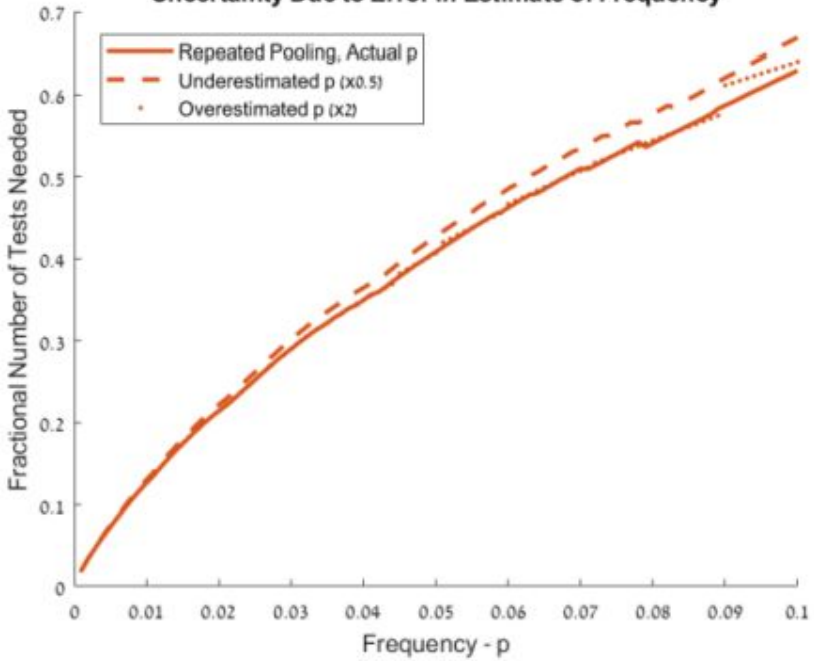
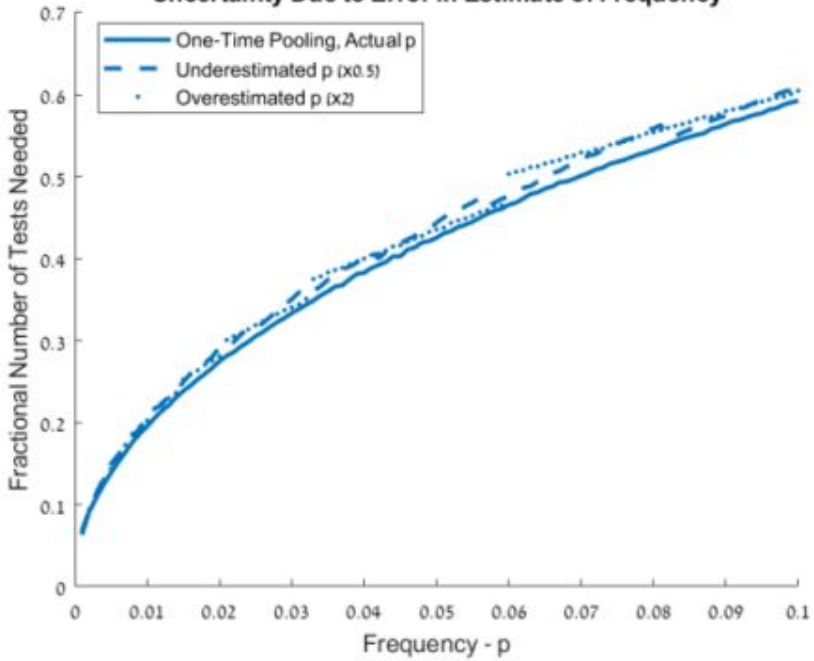




Fig2:
A: The orange region corresponds to the effect of uncertainties for repeated pooling
B:Same as A but for one time pooling. As you can see the ‘splay’ is smaller, ie, one time is more robust

C



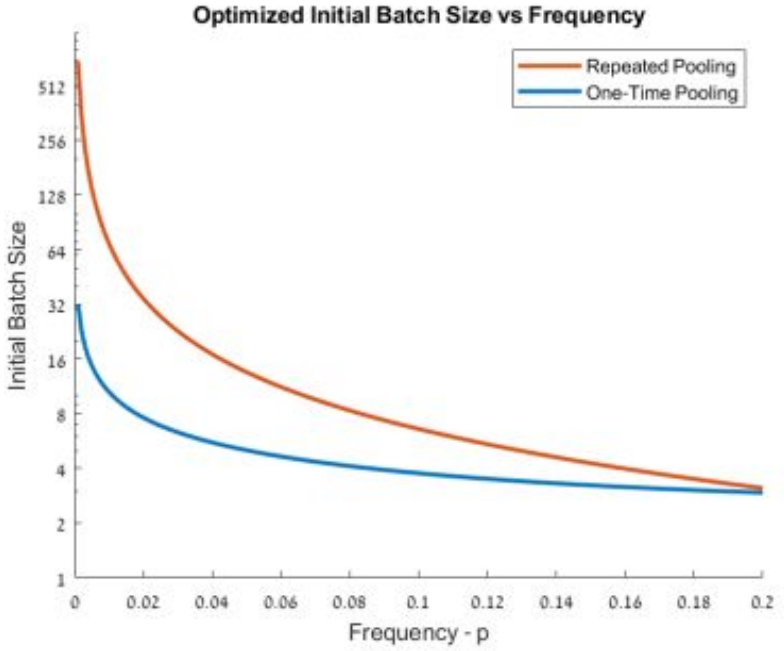

D

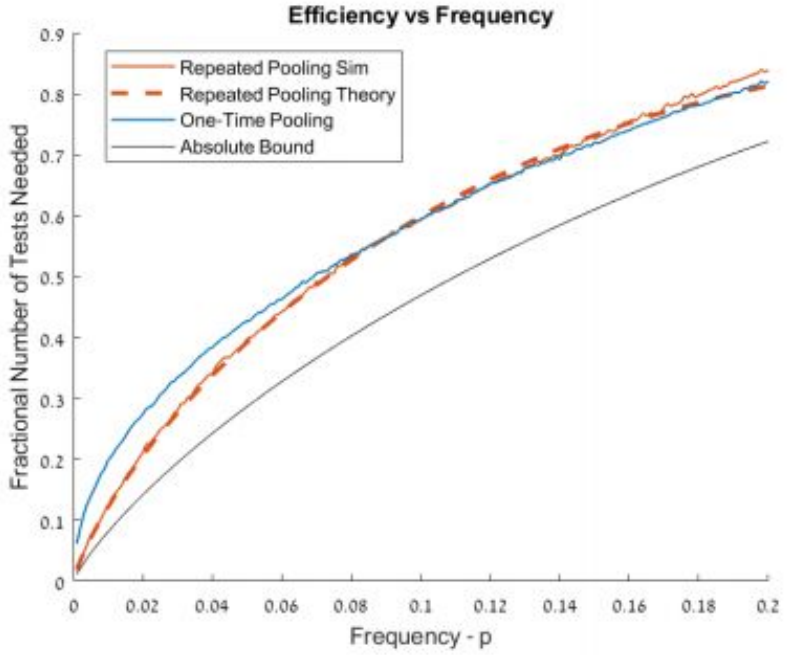



C: Underestimating p means if we considered the value of p less than the true value and continue testing leading to more inefficient testing

Overestimating means the exact opposite

D: Same process with One Time Pooling




The b is smaller for one time pooling (obviously)

As you can see, one time pooling is more efficient at higher prevalence and vice versa for repeated pooling.

# Efficient sample pooling strategies for COVID-19 data gathering (Fisher Information matrix)

This paper also sets out with a similar goal: finding the optimal pool size. This paper gives us an exact expression between the value of prevalence and pool size  Moreover, this paper does not consider any pooling strategy. What the paper does can be represented as follows:

1) Do a test with a larger than optimal batch size.
2) Find the conditional probability of prevalence being p given some number of pools test positive.
3) Maximise this probability to find the expected value of prevalence (p will be a function of n)
4) Use log likelihood to evaluate the fisher information.
5) Maximise fisher information to get optimal batch size.

Let p be the prevalence rate. Then q = 1-p.

Then, the probability of a negative/positive pool result is given by:

$$P_- = q^n \qquad P_+ = 1 - q^n$$

The probability of finding N+ positive results given the prevalence rate is:

$$P(N_+|q) = \binom{N}{N_+}(1-q^n)^{N_+}q^{nN_-}$$

Using bayes rule to flip the probability:

$$P(q|N_+) = P(N_+|q)P(q)/C.$$

C is a constant (normalization factor). P(q) is also taken to be a constant factor and absorbed into C. Differentiating this probability twice to find the maximum over q,

$$\frac{\partial \log P(q|N_+)}{\partial q} = \frac{N_+}{q^n - 1} n q^{n-1} + \frac{N_- n}{q}.$$

$$\frac{\partial^2 \log P(q|N_+)}{\partial q^2} = -\frac{n - 1 + q^n}{(1-q^n)^2} n N_+ q^{n-2} - \frac{N_- n}{q^2}.$$

From these equations we get an estimate for q that minimises the probability.

Fisher matrix is the covariance of the score function (the probability in this case), basically the expected value of the score func multiplied with its transpose.

$$\mathrm{F} = \mathop{\mathbb{E}}_{p(x|\theta)} \left[ \nabla \log p(x|\theta) \, \nabla \log p(x|\theta)^{\mathrm{T}} \right]$$

$$\mathrm{H}_{\log p(x|\theta)} = \mathrm{J} \left( \frac{\nabla p(x|\theta)}{p(x|\theta)} \right) \qquad \mathrm{F} = - \mathbb{E}_{p(x|\theta)} \left[ \mathrm{H}_{\log p(x|\theta)} \right].$$

Maximising fisher information corresponds to minimising the variance.
On taking the differential of the equation for the Fischer Matrix with respect to n( pool size),we get

$$F = \frac{n^2 N q^{n-2}}{1-q^n}. \qquad \frac{\partial F}{\partial n} = \frac{n N q^{n-2}}{(1-q^n)^2}(2 - 2q^n + n \log q)$$

On equating this to 0,we can get the ideal relation between q and n which is given by

$$q^n = 0.203188.$$

On plugging different values of q, we get optimal pool sizes for various values of q.

| $n$ | $q$ |
|---|---|
| 64 | 97.5% |
| 32 | 95% |
| 16 | 90% |
| 8 | 82% |
| 4 | 67% |
| 2 | 45% |

# On Accelerated Testing for COVID-19 Using Group Testing (Info Theory)

The paper presents a highly mathematical and theoretical overview of the GT problem applied to the COVID19 crisis.

The paper deals with group testing strategies for two problems:

1) Testing **individuals**: Identifying individual infected people. (this is similar to the GT presented in class and other papers)
2) Testing **populations**: Identifying whether the infection rate is high or low in a given population. (Similar to another paper we studied, this paper explains it all mathematically!)

**TESTING INDIVIDUALS:**

For testing individuals, the paper assumes a completely **noiseless and ideal testing procedure**, and presents 4 testing strategies employing Adaptive, Nonadaptive and mixed testing. Moreover, the paper does not go into much detail for testing individuals, just the basic strategy has been provided. **No algorithms/pseudo codes have been provided.**

ADAPTIVE TESTING:

The exact same method as that discussed in class. However this is a very slow implementation, and it is not efficient in the worst case. The paper only considers the case when 4 people are pooled and presents a typical binary search format to find the infected individual.

NON ADAPTIVE TESTING:

Again, exact same method as that described in class. The only difference being that they have defined the testing matrix as the transpose of our method. The concept of **k-separability** was also discussed for the testing matrix.

In this case we pool 1,4 2,4 and 3,4
Testing vector is also defined
similarly

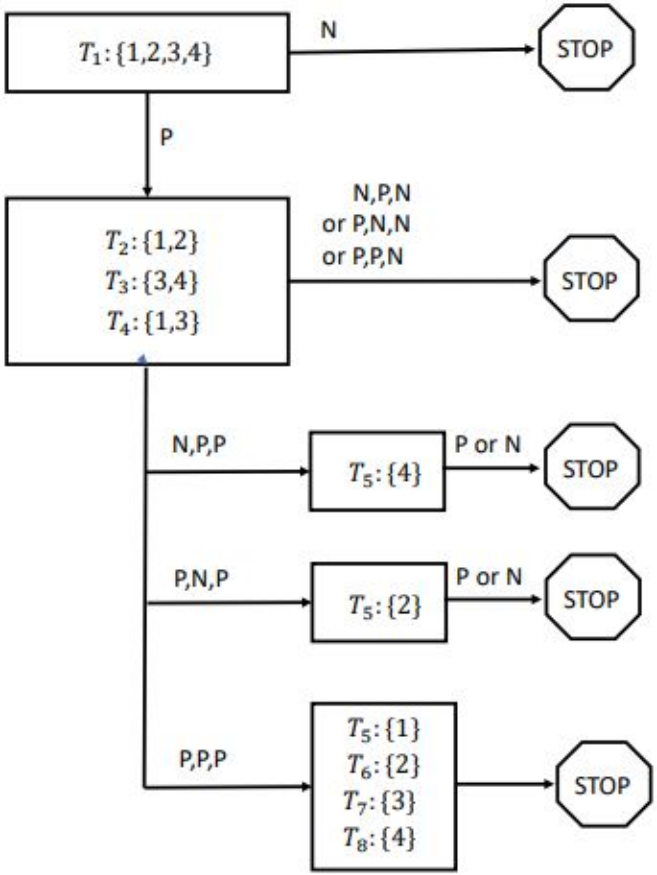
$$M = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix},$$

SPARSITY OBLIVIOUS MULTI-STAGE (SOMS) GROUP TESTING:

It is a **mixture** of adaptive and non adaptive testing methods

No pseudocode was provided, and the algorithm was explained only for a population of size 4.

The number of infected people do not affect the testing scheme (unlike in non adaptive) but the number of tests can become suboptimal in case of large prevalence




Flowchart showing the SOMS group testing strategy for $n = 4$ people.

SPARSITY-OBLIVIOUS FULLY ADAPTIVE (SOFA) GROUP TESTING:

**SOFA needs lesser number of tests on average, however it is also a slower algorithm.**

Let Np be the expected number of infected people. If the number of already identified people is close to Np, then we will perform a single test on the entire group of people. Otherwise, we will perform a binary search.

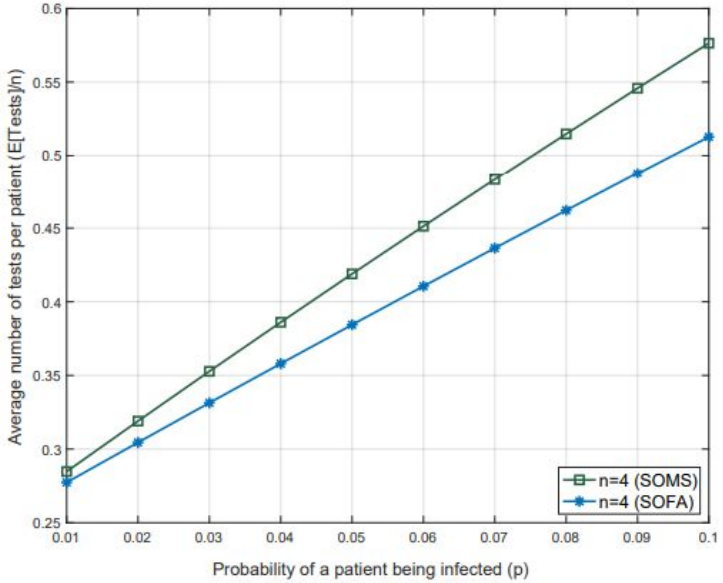
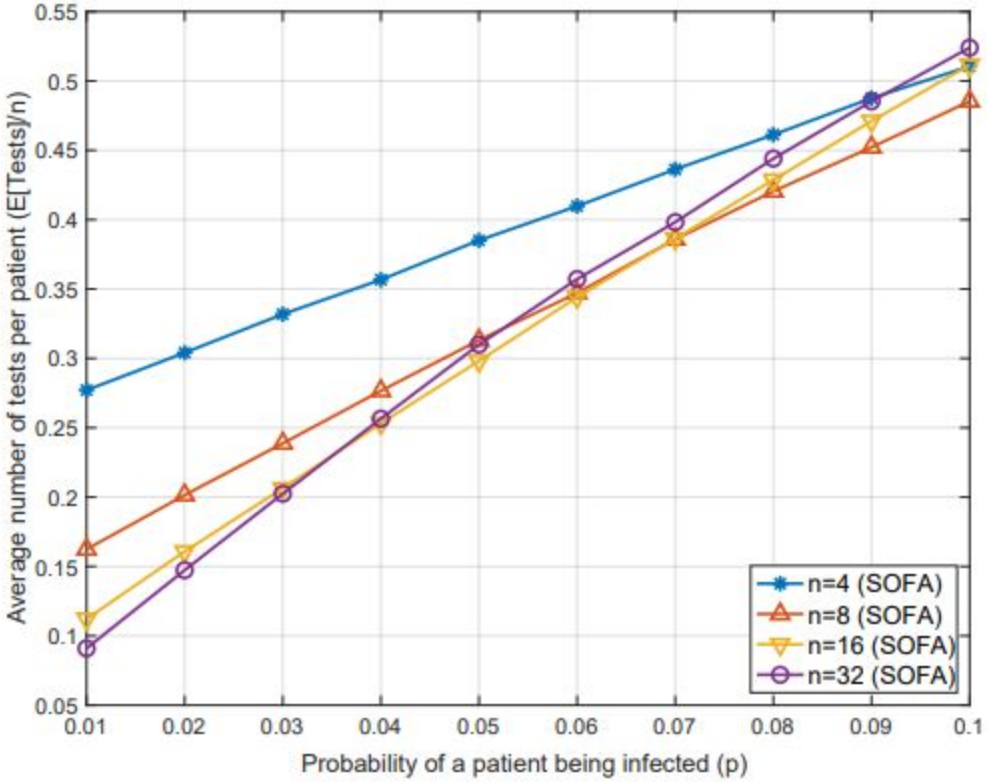



Fig. 3. Average number of tests per person for $n = 4$ and different values of $p$.




ig. 4. Average number of tests per person for different values of $n$ and $p$.

**INFECTION RATE CLASSIFICATION:**

Consider two hypothesis:

$$H_0: \quad \text{Infection rate is } p_0$$

$$H_1: \quad \text{Infection rate is } p_1$$

We take a group of N people divided into L subpools. A subpool is infected if any one person is infected. Define the following random variables:

$$X_k = \begin{cases} 1, & \text{if } S_k \text{ is infected;} \\ 0, & \text{if } S_k \text{ is not infected.} \end{cases}$$

Assuming that the sampling process is random, Xi's can be treated as Bernoulli random variables. The probability of Xi=1 is the probability of any subpool being infected. If p is the prevalence rate,

$$q_i = 1 - (1 - p_i)^{N/L}.$$

Let x be a realization of the random variables X and n(x) be the number of infected subpools

$$\underline{x} = [x_1, x_2, \ldots, x_L] \qquad n(\underline{x}) := \sum_k x_k$$

$$LLR = L(\underline{x}) = log(\frac{P(H_0|\underline{x})}{P(H_1|\underline{x})})$$

$$= log(\frac{\pi_0 P(\underline{x}|H_0)}{\pi_1 P(\underline{x}|H_1)})$$

$$P(\underline{x}|H_0) = q_0^{n(\underline{x})}(1-q_0)^{L-n(\underline{x})}$$

$$P(\underline{x}|H_1) = q_1^{n(\underline{x})}(1-q_1)^{L-n(\underline{x})}$$

$$L(\underline{x}) = log(\frac{\pi_0 q_0^{n(\underline{x})}(1-q_0)^{L-n(\underline{x})}}{\pi_1 q_1^{n(\underline{x})}(1-q_1)^{L-n(\underline{x})}})$$

$$= log(\frac{\pi_0}{\pi_1}) + n(\underline{x})log(\frac{q_0}{q_1}) + [L-n(\underline{x})]log(\frac{1-q_0}{1-q_1})$$

$$= log(\frac{\pi_0}{\pi_1}) + n(\underline{x})[log(\frac{q_0}{q_1}) - log(\frac{1-q_0}{1-q_1})] + Llog(\frac{1-q_0}{1-q_1})$$

Hence, the LLR (and threshold value by extension) depend on $L$, $q_0$ and $q_1$

$$\text{Select } H_0 \text{ if } n(\underline{x}) \leq V = \left\lfloor \frac{\log \frac{\pi_0}{\pi_1} + L \log \frac{1-q_0}{1-q_1}}{-\log \frac{q_0}{q_1} + \log \frac{1-q_0}{1-q_1}} \right\rfloor.$$

We will decide upon our infecton rate depending on whether or not the total number of subpools(n(x)) is greater than the threshold decided (V). We will perform binary splitting to achieve this, and wll stop binary splitting as soon as our threshold value is broken.

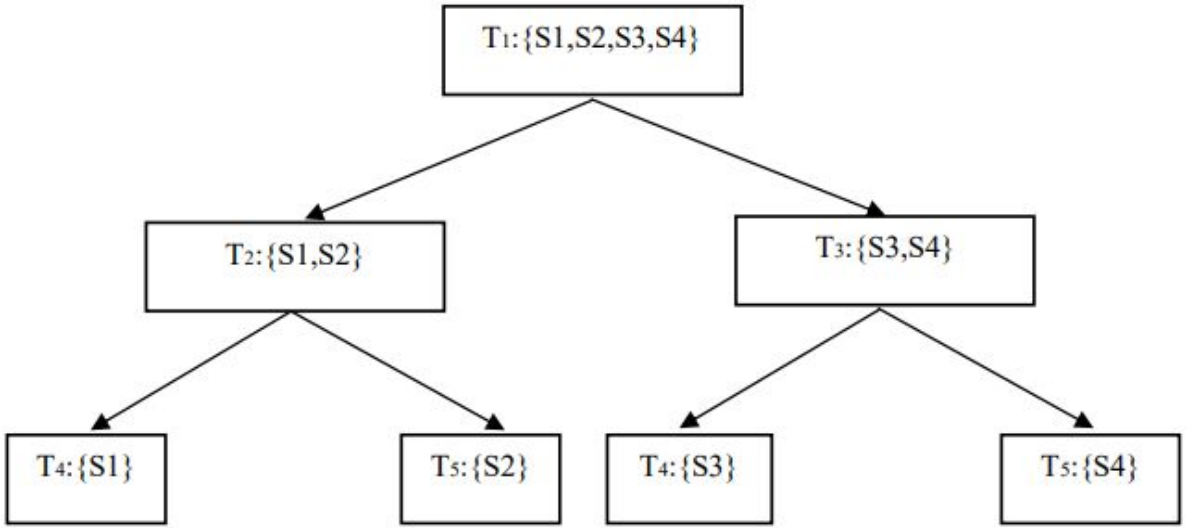



**Figure 1. Flowchart showing the binary splitting procedure.**

PROBABILITY OF FALSE ALARM, DETECTION AND AVERAGE NUMBER OF TESTS:

The probability of false alarm ($P_F$) and probability of correct detection ($P_D$) are given by

$$\begin{aligned} P_F &:= \mathbb{P}(n(\underline{x}) > V | H_0 \text{ is true}) \\ &= \sum_{j=V+1}^{L} \binom{L}{j} q_0^j (1-q_0)^{L-j}, \end{aligned}$$

$$\begin{aligned} P_D &:= \mathbb{P}(n(\underline{x}) > V | H_1 \text{ is true}) \\ &= \sum_{j=V+1}^{L} \binom{L}{j} q_1^j (1-q_1)^{L-j}. \end{aligned}$$

$$\mathbb{E}[\Gamma] = \sum_{\underline{x}} \mathbb{P}(\underline{x}) \Gamma(\underline{x})$$

The paper also suggests a various number of small optimisations that can be made on this process. The threshold can be chosen in order to trade between Pf/Pd (shown later in graphs).

$p_0 = 0.01, p_1 = 0.05, \pi_0 = 0.5, \pi_1 = 0.5.$

$$N = 64$$
$$L = 4$$
$$V = 1.$$

| $\underline{x}$ | $\mathbb{P}(\underline{x}\|H_i)$ | Chosen hypothesis | No. of tests $\Gamma(\underline{x})$ |
|---|---|---|---|
| 0 0 0 0 | $(1-q_i)^4$ | $H_0$ | 1 |
| 0 0 0 1 | $q_i(1-q_i)^3$ | $H_0$ | 5 |
| 0 0 1 0 | $q_i(1-q_i)^3$ | $H_0$ | 5 |
| 0 0 1 1 | $q_i^2(1-q_i)^2$ | $H_1$ | 5 |
| 0 1 0 0 | $q_i(1-q_i)^3$ | $H_0$ | 5 |
| 0 1 0 1 | $q_i^2(1-q_i)^2$ | $H_1$ | 3 |
| 0 1 1 0 | $q_i^2(1-q_i)^2$ | $H_1$ | 3 |
| 0 1 1 1 | $q_i^3(1-q_i)$ | $H_1$ | 3 |
| 1 0 0 0 | $q_i(1-q_i)^3$ | $H_0$ | 5 |
| 1 0 0 1 | $q_i^2(1-q_i)^2$ | $H_1$ | 3 |
| 1 0 1 0 | $q_i^2(1-q_i)^2$ | $H_1$ | 3 |
| 1 0 1 1 | $q_i^3(1-q_i)$ | $H_1$ | 3 |
| 1 1 0 0 | $q_i^2(1-q_i)^2$ | $H_1$ | 5 |
| 1 1 0 1 | $q_i^3(1-q_i)$ | $H_1$ | 3 |
| 1 1 1 0 | $q_i^3(1-q_i)$ | $H_1$ | 3 |
| 1 1 1 1 | $q_i^4$ | $H_1$ | 3 |

$$\mathbb{E}[\Gamma] = \left[(1-q_i)^4\right] \cdot 1 + \left[4q_i^2(1-q_i)^2 + 4q_i^3(1-q_i) + q_i^4\right] \cdot 3 + \left[4q_i(1-q_i)^3 + 2q_i^2(1-q_i)^2\right] \cdot 5.$$

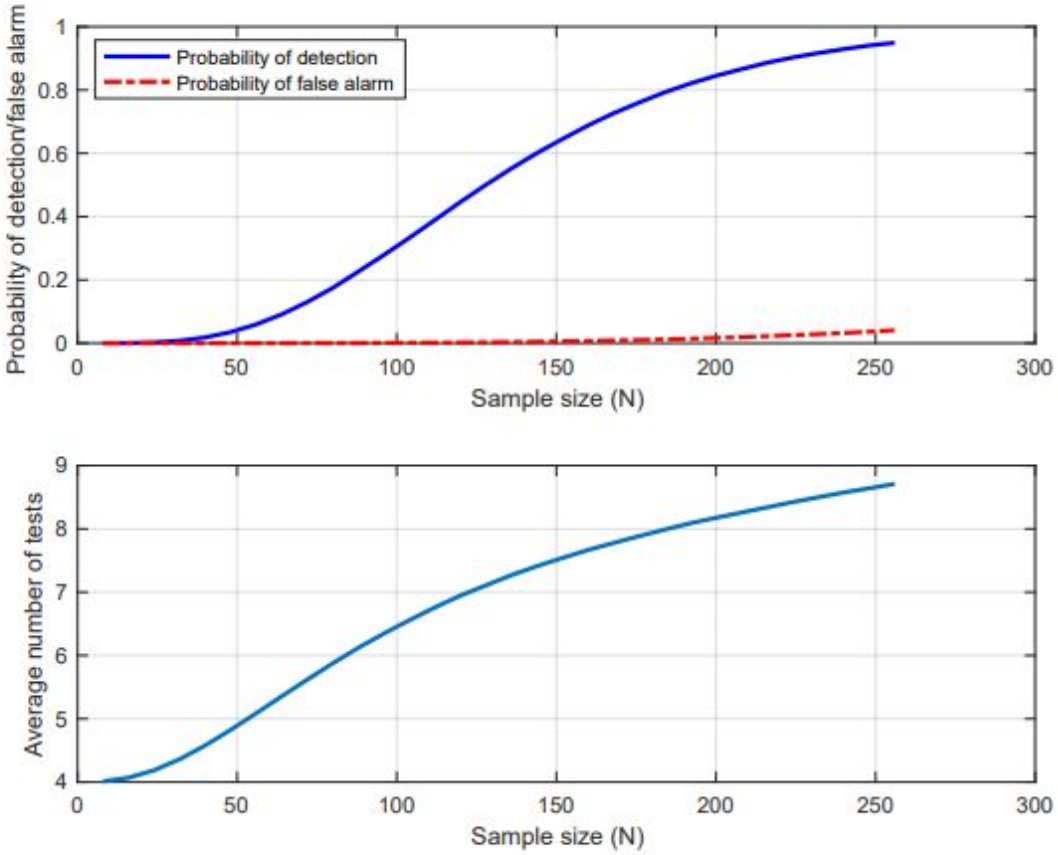
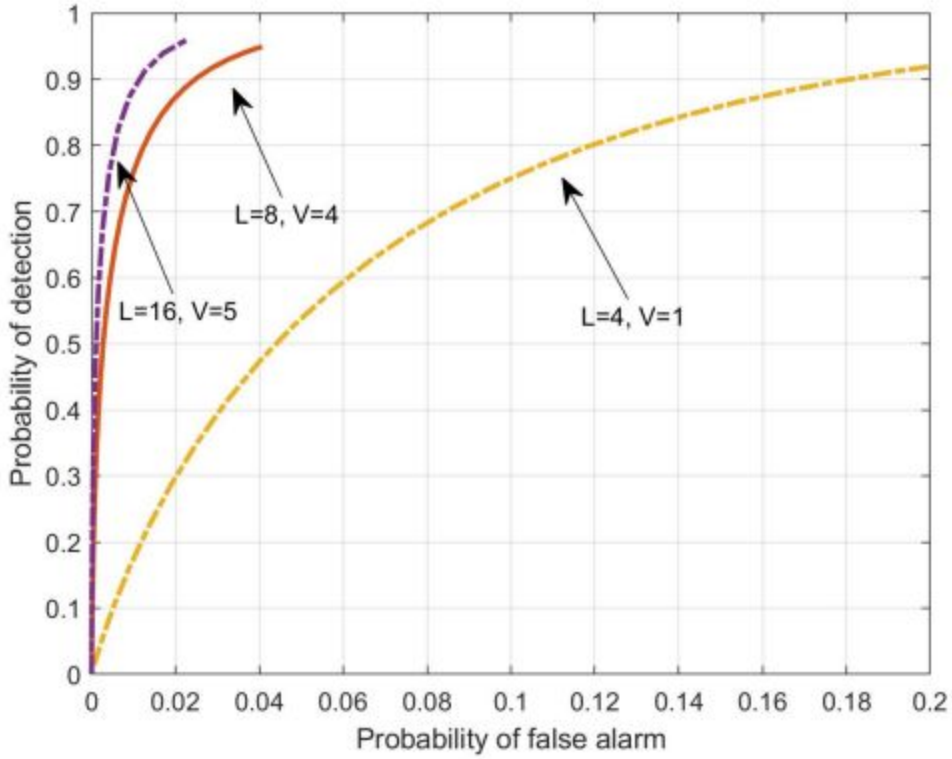



Fig. 7. $P_F, P_D$, and $\mathbb{E}[\Gamma]$ as a function of $N$ when $L = 8$, $V = 4$, $p_0 = 0.01$, and $p_1 = 0.05$.




ig. 8. ROC curve obtained by changing $N$ for different values of $L$ and $V$ when $p_0 = 0.01, p_1 = 0.05$.

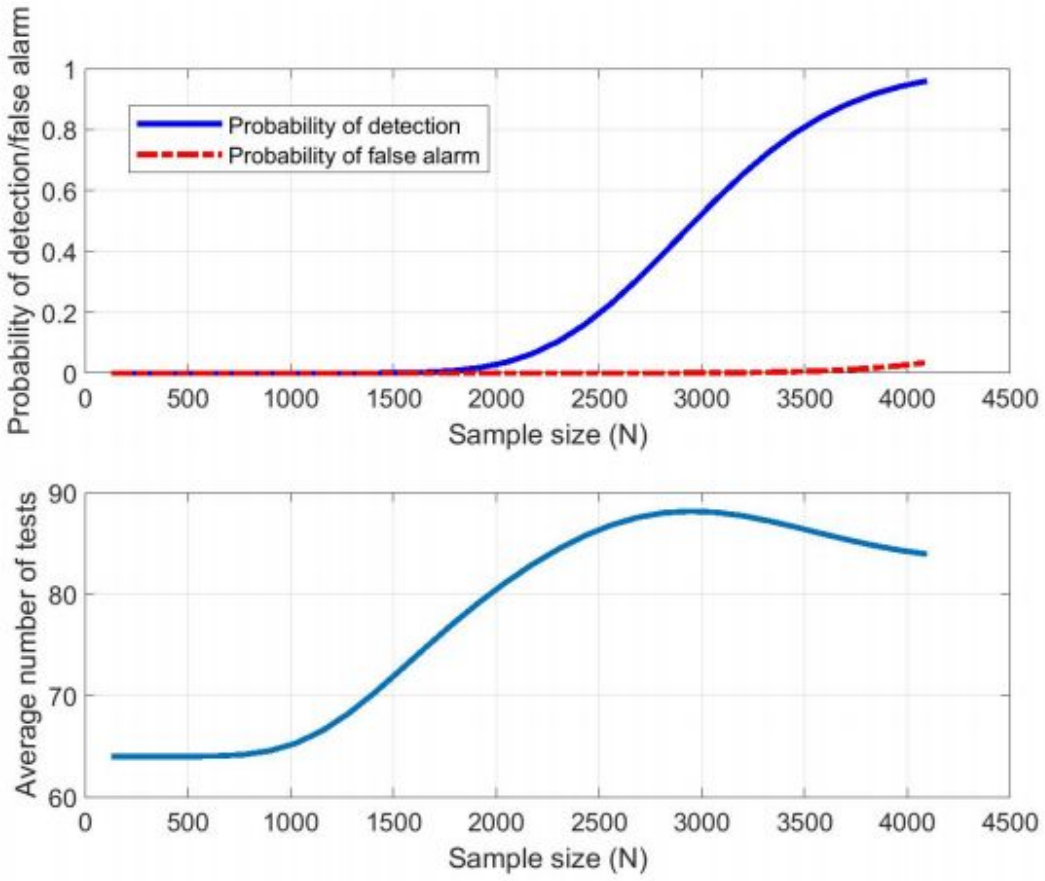



Fig. 9. $P_F, P_D$ and $\mathbb{E}[\Gamma]$ as a function of $N$ when $L = 128$, $V = 26$, $p_0 = 0.005$, and $p_1 = 0.01$.

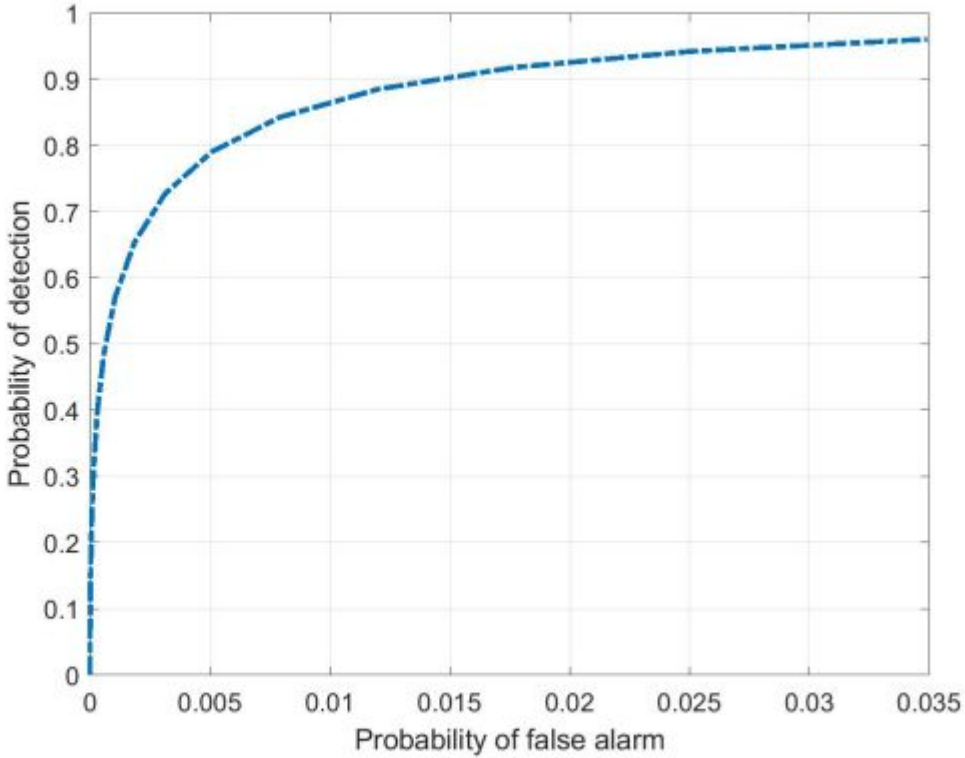



Fig. 10. ROC curve obtained by changing $N$ when $L = 128$, $V = 26$, $p_0 = 0.005$, and $p_1 = 0.01$.

ACCOUNTING FOR NOISE:

The paper takes the effect of uncertainties into account and makes the following changes:

1) The probability of a negative result in an uninfected subpool is 1
2) The probability of a positive result in an infected subpool ‘i’ is ρi.

**We have written a matlab code to simulate the following tables.**

Let us assume that hypothesis $H_0$ is true. Define the following events:

Ai: The event that someone in $ith$ group is infected
Bi: The event that the $ith$ group tests positive

Then, the following probabilities are defined:

$$P(A) = q_0$$

$$P(B|A) = \rho_i$$

$$P(A \cap B) = P(B|A)P(A)$$

$$P(A \cap B) = \rho_i q_i$$

$$P(infected\ group\ tests\ positive) = \rho_i q_i$$

$P_D$, $P_F$, AND $\mathbb{E}[\Gamma]$ FOR DIFFERENT VALUES OF $N$, $L$, AND $V$ WHEN $p_0 = 0.01$ AND $p_1 = 0.05$.

| Pool size $N$ | No. of subpools $L$ | Max. pool size $\frac{2N}{L}$ | Sensitivity $\rho$ | $P_D$ | $P_F$ | $\mathbb{E}[\Gamma]$ | Threshold $V$ |
|---|---|---|---|---|---|---|---|
| 256 | 8 | 64 | 100% | 96.0% | 4.1% | 8.7 | 4 |
| 256 | 8 | 64 | 80% | 88.8% | 7.7% | 6.7 | 3 |
| 256 | 16 | 32 | 100% | 98.8% | 7.6% | 9.3 | 4 |
| 256 | 16 | 32 | 80% | 91.2% | 3.4% | 9.8 | 4 |
| 448 | 14 | 64 | 100% | 99.8% | 6.2% | 10.4 | 6 |
| 448 | 14 | 64 | 80% | 97.3% | 6.6% | 11.1 | 5 |
| 448 | 28 | 32 | 100% | 99.9% | 4.6% | 16.8 | 7 |
| 448 | 28 | 32 | 80% | 99.1% | 4.1% | 16.1 | 6 |

$P_D$, $P_F$, AND $\mathbb{E}[\Gamma]$ FOR DIFFERENT VALUES OF $N$, $L$, AND $V$ WHEN $p_0 = 0.005$ AND $p_1 = 0.01$.

| Pool size $N$ | No. of subpools $L$ | Max. pool size $\frac{2N}{L}$ | Sensitivity $\rho$ | $P_D$ | $P_F$ | $\mathbb{E}[\Gamma]$ | Threshold $V$ |
|---|---|---|---|---|---|---|---|
| 4096 | 128 | 64 | 100% | 97.5% | 5.6% | 83.0 | 25 |
| 4096 | 128 | 64 | 80% | 92.5% | 4.7% | 81.0 | 21 |
| 4096 | 256 | 32 | 100% | 98.3% | 6.1% | 146.0 | 26 |
| 4096 | 256 | 32 | 80% | 94.2% | 4.6% | 143.2 | 22 |
| 4736 | 148 | 64 | 100% | 98.3% | 4.4% | 95.8 | 29 |
| 4736 | 148 | 64 | 80% | 94.9% | 4.3% | 92.7 | 24 |
| 4736 | 296 | 32 | 100% | 98.9% | 5.2% | 168.9 | 30 |
| 4736 | 296 | 32 | 80% | 96.4% | 4.6% | 165.6 | 25 |

# Noisy Pooled PCR for Virus Testing (GAMP)

The paper presents a non adaptive testing scheme (using a testing matrix) and calls it a “linear inverse problem”. This paper mainly focuses on the fact that false positives and negatives can be treated as channel noise. It reduces this noise from the input and output channels using the generalized approximate message passing (GAMP), a compressed sensing algorithm.

$$w = Ax \in \mathbb{N}^M,$$

Here, A is the MxN testing matrix, x is the Nx1 vetor containing information about whether or not a particular individual is infected, and w contains the Mx1 test results. Here, we have done normal matrix multiplication so the ith element of w contains the number of positive cases in that particular test.
The paper models the noise as follows:

1) When the test is performed, some noise is added to the pooled samples which turns the vector x into the vector q.
2) The vector w is then calculated as w=Aq.
3) Noise is then added to te vector w and it turns into the vector y.
4) The algorithm aims to get w from y, then get q from w, then get x from q.

The probability of any one person being infected is modeled as an IID bernoulli random variable (X):

$$\Pr(X_n = 1) = \rho \quad \text{and} \quad \Pr(X_n = 0) = 1 - \rho,$$

Further, we model the output noise using the probability P(Y|W). If there was no noise this probability would always be 1 when Y=W and 0 everywhere else. Y and W are both vectors.

Let the probability of false negative by one patient, ie, P(Y=0|W=1), be given by p1.
The probability of false positive was a little vague in this case: $\Pr(Y_m = 0|W_m = w_m) = (1 - p_2)(p_1)^{w_m}$.

This is defined as our channel.

The measurement rate R is defined as: $\lim_{N\to\infty} \frac{M(N)}{N} = R$

The noise in the input is defined as: $q = x + v,$ where v is mean AWGN

The denoising function at the input can be defined as:

$$\widehat{x}_n = g_{in}(\Delta_{vn}, q) = E[X_n | Q_n = X_n + \mathcal{N}(0, \Delta_{vn})].$$

This function reduces the MSE (mean squared error) as quickly as possible.

$$h_m = g_{out}(k_m, y_m, \Theta_m) = \frac{E[W_m | K_m, Y_m, \Theta_m] - k_m}{\Theta}$$

This hm is a correction term for w and is used to estimate the value of x.

With each iteration, they improve the estimates of hm and x.

$$q = \widehat{x} + \Delta_v A^T h$$

Eventually after a set number of iterations (tmax). The algorithm ends.

**Algorithm 1** GAMP

**Inputs.** Maximum iterations $t_{max}$, percentage of sick patients $\rho$, false negative probability $p_1$, false positive probability $p_2$, measurements $y$, and matrix $A$.

**Initialize.** $t, k, h_m, \Theta_m, \widehat{x}_n, s_n, \forall m, n$.

**Comment.** $t$ is iteration number, $k$ is mean of our estimate for $Ax$, $h_m$ is correction term for $w_m$, $\Theta_m$ is variance of $h$, $\widehat{x}_n$ is our estimate for $x_n$, $s_n$ is variance in our estimate $\widehat{x}_n$.

**while** $t < t_{max}$ **do**
  **// clean up output channel**
  $\Theta = (A)^2 s$ // variance of $h$
  $k = A\widehat{x} - \Theta h$ // mean of $w$ per previous iteration
  **for** $m = 1$ to $M$ **do**
    $h_m = g_{out}(k_m, y_m, \theta_m)$
    **Comment:** $\frac{1}{\Theta}(E[W_m|K_m, Y_m, \Theta_m] - k_m)$.
    $r_m = -\frac{\partial}{\partial k_m} g_{out}(\cdot)$
  $\Delta_v = \left\{\frac{1}{N}(A^T)^2 r\right\}^{-1}$ // scalar channel noise variance
  $q = \widehat{x} + \Delta_v A^T h$ // pseudo data
  **// clean up input channel**
  **for** $n = 1$ to $N$ **do**
    $\widehat{x}_n = g_{in}(\Delta_{vn}, q_n) = E[x_n|q_n]$ // mean estimate
    $s_n = E[x_n^2|q_n] - E^2[x_n|q_n]$ // variance estimate
  $t = t + 1$

**Output.** Estimate $\widehat{x}$, pseudo data $q$, and scalar channel noise variance $\Delta_v$.

The ROC (receiver operating characteristic) is a plot that shows the diagnostic ability of a bnary classifier. The Area Under the Curve of the ROC demonstrates the diagnostic ability. The higher the AUCROC, the better classifier we have.

The ROC curves were also plotted in the last paper but were only plotted in two dimensions.

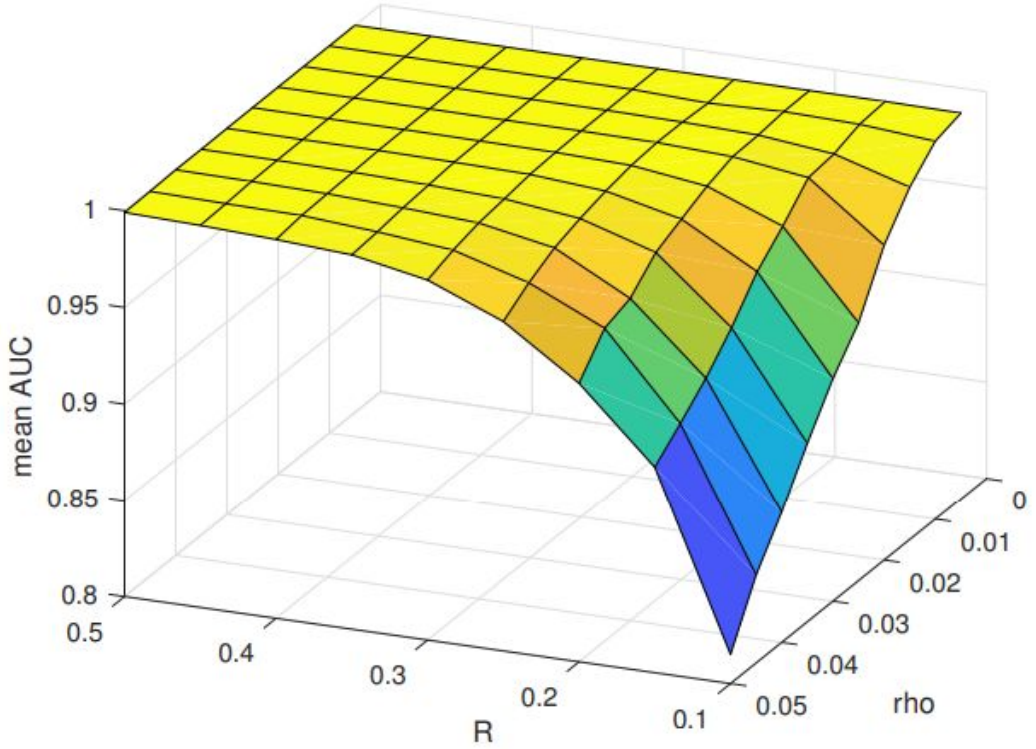



Fig. 3. Estimation accuracy in AUC (vertical axis) as a function of the percentage of sick patients, $\rho$, and measurement rate, $R = M/N$. ($N = 5000$; $n_{pos} = 0.5$; $p_1 = 0.02$; $p_2 = 0.001$.)

The top figure shows how noise reduces as the number of iteration go on

The bottom figure shows how close our estimate is to the real value.

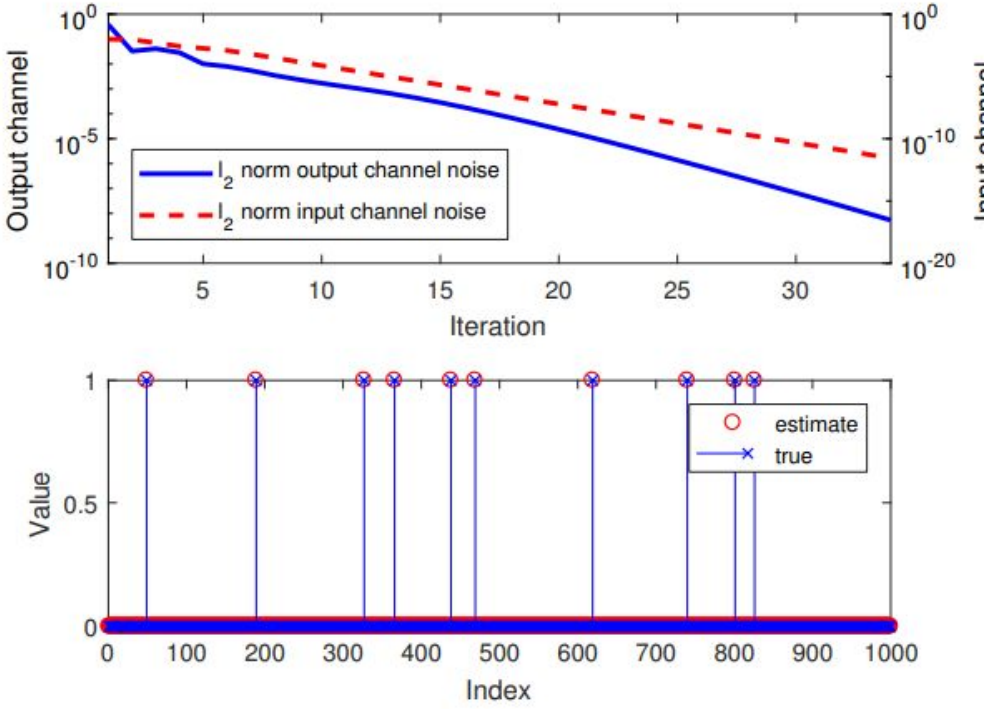



Fig. 2. Top: $\ell_2$ norms of the input (dashed red line; associated with right vertical axis) and output (solid blue; left) channel noise as functions of the GAMP iteration. Bottom: The first 1000 entries of the unknown patient illness status vector $x$, and their estimates. ($N = 5000$ patients; $\rho = 0.01$ percentage of sick patients; measurement rate $R = M/N = 0.3$; $n_{pos} = 0.5$ average sick patients per measurement; false negative probability $p_1 = 0.02$; false positive $p_2 = 0.001$.)

CHALLENGES:

1) p1, the false negative rate will not be the same for every patient, since it depends upon the amount of genetic material (virus particles) on the swab taken from the patient.
2) Similarly the false positive probability will vary.
3) In the testing matrix, since all the rows and columns are binary, they all provide the same SNR value. This is not realistic and we can change A to support non binary values in order to have more control over our experiment.

In general, refinement in matrix design will help us make a more accurate testing scheme.

# Group Testing for COVID-19: How to Stop Worrying and Test More

This paper presents 3 group testing algorithms (along with their detailed pseudocode).
Also, the paper discusses how various schemes such as replication of tests and repeated sampling of swab affects our test results. The detail given in this paper is unmatched, most other papers have assumed AWGN, or considered how the results would change if one of the parameters (such as prevalence rate) was mistaken.

**Definitions:**

False negative rate: The probability of getting a negative outcome when the virus is actually present in the sample.

$$False\ negative\ rate: \gamma = \Pr(X_t = 0 | X = 1)$$

False Positive Rate: The probability of getting a positive outcome when the virus is not present in the Sample

$$False\ positive\ rate: \beta = \Pr(X_t = 1 | X = 0)$$

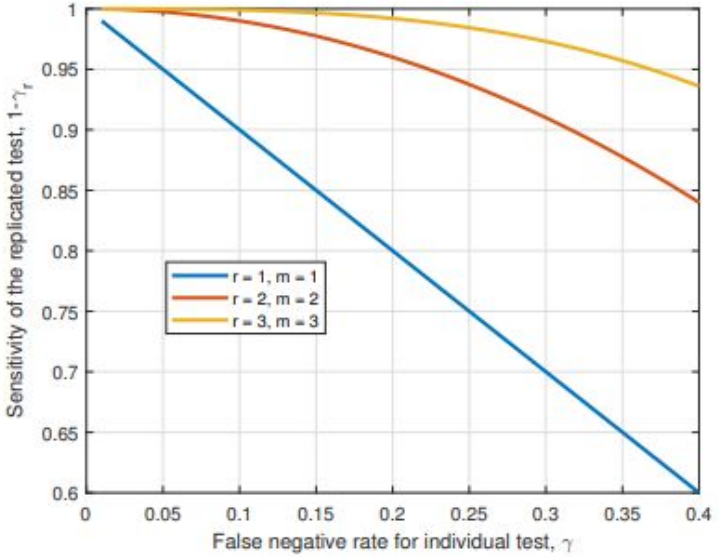
Prior Probabilty:Probability that a sample actually has the virus

$$- Prior: \alpha = \Pr(X = 1)$$

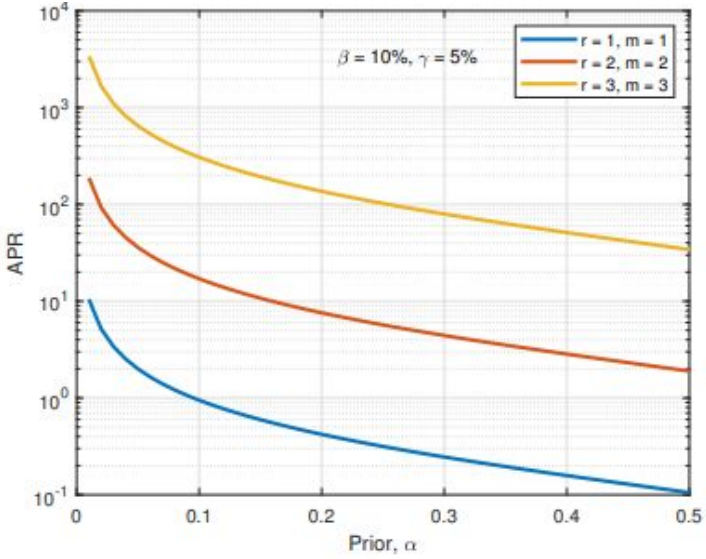
Sensitivity: $1-\gamma = \Pr(X_t = 1|X = 1)$

Replication: Tests on samples are repeated to confirm the outcome of a test,and hence they increase the accuracy of the tests. These graphs analyse the improvement in accuracy as the number of tests increases:

$$APR = \frac{\Pr(X = 0|X_1, \cdots, X_r)}{\Pr(X = 1|X_1, \cdots, X_r)} = \frac{\Pr(X_1, \cdots, X_r|X = 0)}{\Pr(X_1, \cdots, X_r|X = 1)} \frac{1-\alpha}{\alpha} = \frac{(1-\alpha)\beta^{r-m}(1-\beta)^m}{\alpha\gamma^m(1-\gamma)^{r-m}},$$




(a) Variation of sensitivity for different replicates.




(b) Variation of APR for different replicates.

Figure 2: Effect of replication on the accuracy of the tests for $r$ replicates and $m$ number of negative outcomes

Let the false negative probability be given by $\gamma$
Upon r replications, the probability of false negative will be given by $\gamma_r$
By definition, the probability of a false negative after $r$ replications means that the majority of the replications tested negative while the person is infected.
Let $X_t$ be the Bernoulli random variable that takes values 0 or 1 depending on $t^{th}$ test is positive or not.

$$\begin{cases} \mathrm{X}_t = 0 & \mathrm{t}^{th} \text{ test is negative} \\ \mathrm{X}_t = 1 & \mathrm{t}^{th} \text{ test is positive} \end{cases}$$

Then the probability of a false negative after r replications is given by:

$$\begin{aligned} \gamma_r &= Pr\left(\sum_{t=1}^{r} X_t < \left\lceil \frac{r}{2} \right\rceil \middle| X = 1\right) \\ &= \sum_{t=\lfloor \frac{r}{2} \rfloor + 1}^{r} \binom{r}{t} \gamma^t (1-\gamma)^{r-t} \end{aligned}$$

Similarly the replicated false positive probability is given by:

$$\beta_r = \sum_{t=\lceil \frac{r}{2} \rceil}^{r} \binom{r}{t} \beta^t (1-\beta)^{r-t}$$

GROUP TESTING SCHEMES:

1) Combinatorial Group Testing: These require an upper bound on D, the number of infected samples in the group.
2) Probabilistic Group Testing: This requires an upper bound on the prior probability and detects all the samples with probability Pd. In some cases number of tests(T) might become greater than the number of people, usually a trade-off exists between T and Pd.

It is known that as the number of defective items approaches the total number of items, exact combinatorial solutions require significantly more tests than probabilistic solutions — even probabilistic solutions permitting only an asymptotically small probability of error

Group Testing is only efficient when the ratio of D/N is:

$$\frac{D}{N} < \frac{3-\sqrt{5}}{2}.$$

When the ratio becomes larger than this the group testing schemes become sub-optimal and it is better to perform individual tests on all the samples.

EFFECTS OF DILUTION:

Sampling dilution: When a sample is divided for replication, the viral load (particles) get distributed. If L is the minimum viral load required for accurate detection, and Vl is the total viral load from one swab, a single swab cannot be divided into more than **Vl/L** samples.

Moreover, the false negative rate **depends upon the viral load** present in the sample. If we want a false negative rate of $\gamma^*$

Then the corresponding viral load will be $\gamma^{-1}(\gamma*)$

So the maximum number of samples with the desired false negative rate is given by $\frac{V_l}{\gamma^{-1}(\gamma^*)}$

Pooling Dilution: When a portion of N samples,out of which D are infected, the virus samples from the N-D uninfected samples **dilute and reduce the concentration of the virus particles**. The senstivity after pooling dilution is given by:

$$1 - \gamma_N = \Phi\left(1.6449 \frac{\log\left(\frac{\chi D V_p}{N V_{50}}\right)}{\log\left(\frac{V_{95}}{V_{50}}\right)}\right) \qquad N = \frac{V_l}{rTV_{50}} 10^{-\frac{\Phi^{-1}(1-\gamma^*)}{1.6449}\log\frac{V_{95}}{V_{50}}},$$

χ is the number of RNA copies per viral particle (χ = 1 in the case of corona virus), D is the number of infected samples, Vp is the viral load in each infected sample, V50 and V95 are the viral loads corresponding to a sensitivity of 0.5 and 0.95, respectively

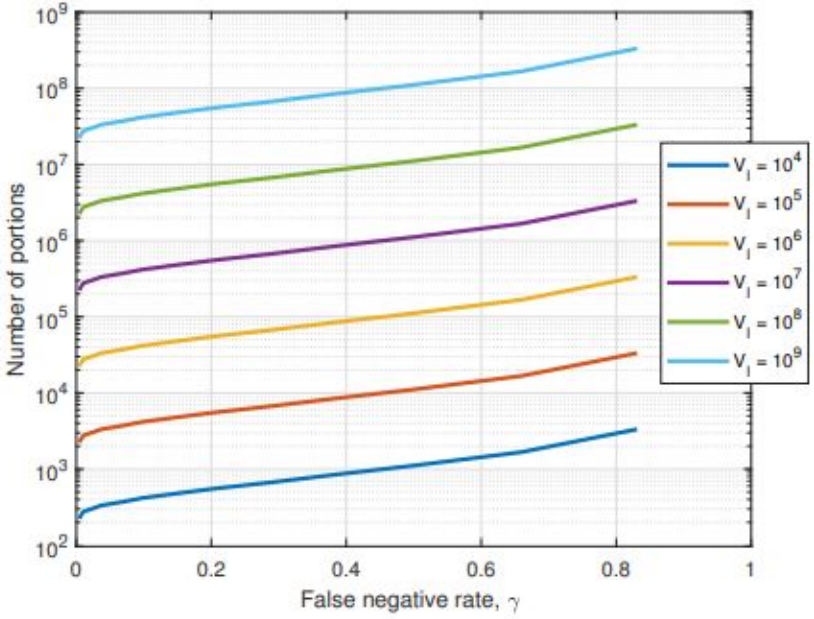



Figure 4: False negative rates achieved by different portion size at different sample viral loads ($V_l$).

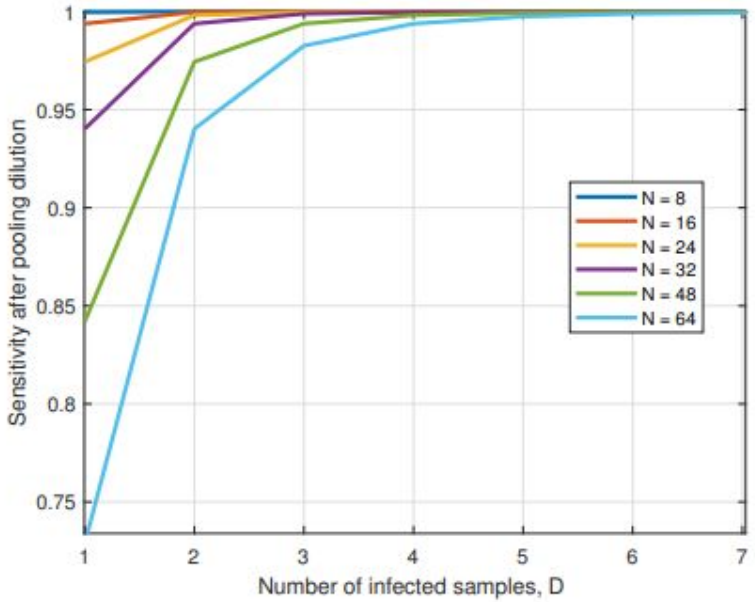



Figure 5: Effect of pooling dilution on sensitivity.

**ALGORITHMS:**

**GENERALIZED BINARY SPLITTING:**

Most commonly used CGT algorithm. Following is the BSP algorithm

- **Step 1** Split the $N$ samples into two halves, say groups $G_1$ and $G_2$.
- **Step 2** Test $G_1$.
- **Step 3** When the test is positive: (i) continue performing all future tests with the samples from only $G_1$, (ii) set $N = |G_1|$, and (iii) if the number of samples in $G_1$ is 1, then one infected sample has been identified and the algorithm is terminated.
- **Step 4** When the test is negative: (i) continue performing all future tests with the samples from only $G_2$, (ii) set $N = |G_2|$, and (iii) if the number of samples in $G_2$ is 1, then one infected sample has been identified and the algorithm is terminated.
  Note that, since $D \geq 1$, if $G_1$ tests negative, then $G_2$ must test positive.
- **Step 5** With the updated $N$ and samples pool, goto **Step 1**.

This algorithm is applied D times to form the GBS algorithm. The above algo only works if D=1.

**Algorithm 2** Generalized Binary Splitting Test

**Input**: $N, D$ and samples pool $\mathcal{P} = \{1, 2, \cdots, N\}$.
**while** $N \geq 2D - 1$ and $D > 0$ **do**
    Choose a group $\mathcal{G}$ of size $2^{\lfloor \log_2 \frac{N-D+1}{D} \rfloor}$
    Test group $\mathcal{G} \subseteq \mathcal{P}$
    **if** test outcome is positive **then**
        Identify an infected sample in $\mathcal{G}$ with BSP (Since the group tested positive, it must contain at least one infected sample)
        Update $N = N - 1 - g$ (where $g$ is the number of uninfected items diagnosed from BSP, remove these from the pool)
        Update $D = D - 1$ (remove the identified infected sample from the pool)
    **else**
        Update $N = N - |\mathcal{G}|$ ($|\mathcal{G}|$ is the number of uninfected items in $\mathcal{G}$, remove these from the pool)
    **end if**
**end while**
**if** $D > 0$ and $N > 0$ **then**
    Test the $N$ samples individually
**end if**

Practical Considerations:

- Since GBS is purely **adaptive**, it is sequential leading to it having extremely long execution times
- Suited when D/N < 0.2
- **Inherent replication** may happen in GBS, but it also may not test some samples at all. Need to replicate carefully
- If D is underestimated, then the algorithm will fail **(become suboptimal**

**This graph is shown in more detail later on**

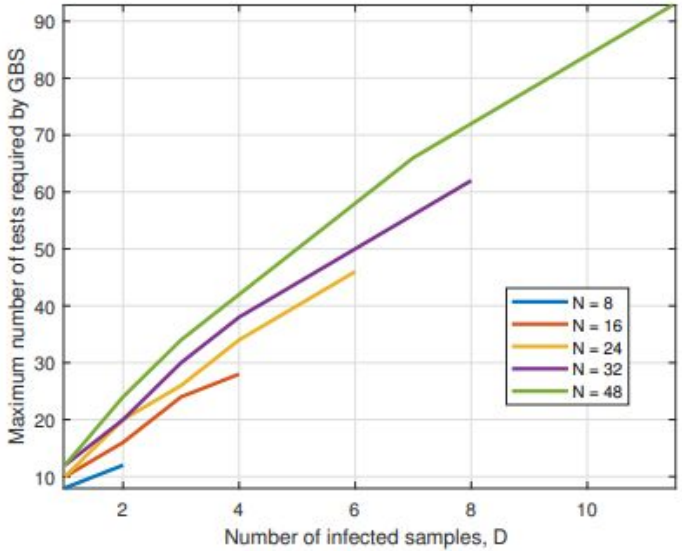



Figure 6: Worst case number of tests required by GBS with double replication for different values of $N$ and $D$.

**MULTI-STAGE TESTING (MST)**

Maintain **three bins**: Uninfected bin, Infected bin and Queued bin.

At the i-th stage, calculate the **batch** size as

$$k_i = \delta^{1-\frac{i}{s}}$$

And the **group** size as:

$$g_i = \lceil N_{i-1}/k_i \rfloor$$

The **number of stages** is given by s where

$$s = \underset{x \in \left\{ \lfloor \ln \frac{N}{D} \rfloor, \lceil \ln \frac{N}{D} \rceil \right\}}{\arg\min} \; xD\left(\delta\right)^{\frac{1}{x}}, \qquad \text{where} \quad \delta \triangleq \frac{N}{D}.$$

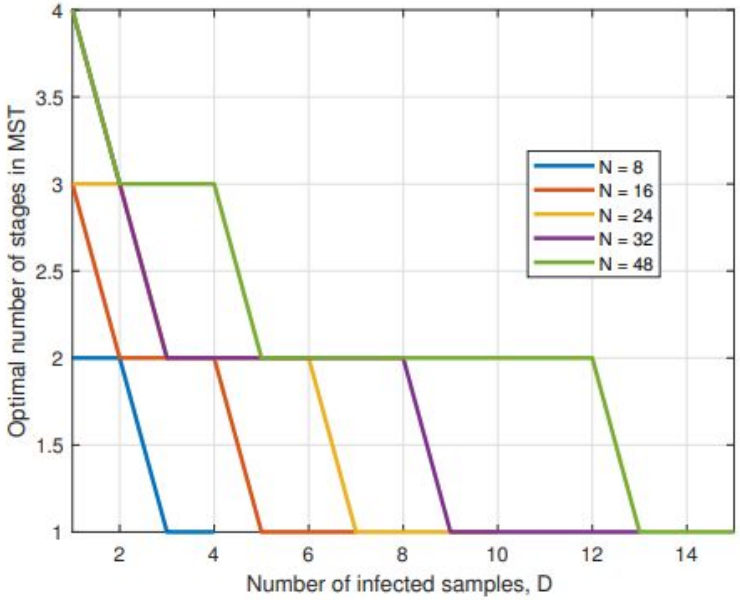
The **total number of tests** is given by $T = \sum_{i=1}^{s} g_i$

**Algorithm 1** Multi-stage Testing

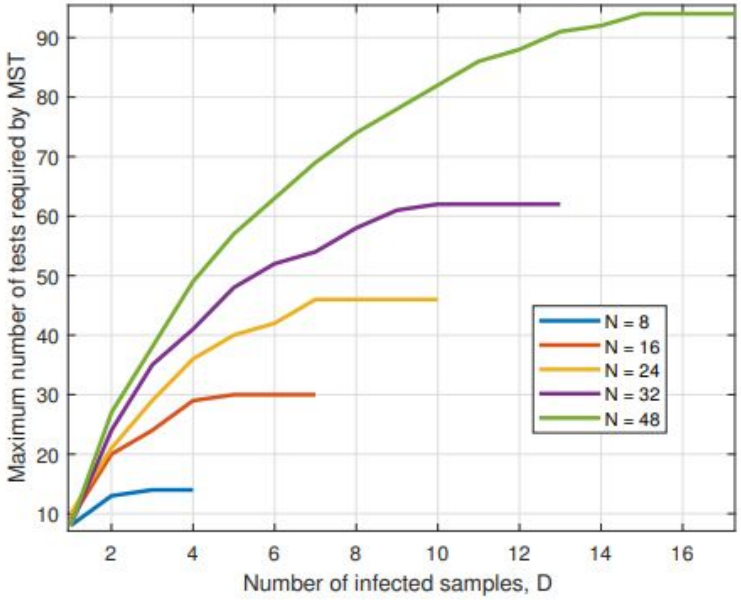
**Input**: $N, D, s$ and samples pool $\mathcal{P}_0 = \{1, 2, \cdots, N\}$.
$N_0 = N$
**for** $i = 1$ to $s$ **do**
    $k_i = \delta^{1-\frac{i}{s}}$
    $g_i = \lceil N_{i-1}/k_i \rfloor$
    Divide $\mathcal{P}_{i-1}$ into $g_i$ disjoint groups - $\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_{g_i}$ such that $\mathcal{G}_1 \cup \mathcal{G}_2 \cup \cdots \cup \mathcal{G}_{g_i} = \mathcal{P}_{i-1}$
    Test groups $\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_{g_i}$
    Discard groups that tested negative
    $\mathcal{P}_i = \{$samples from groups that tested positive$\}$, $\mathcal{P}_i \subset \mathcal{P}_{i-1}$
    $N_i = |\mathcal{P}_i|$
**end for**
Return $\mathcal{P}_s$ as the set of infected samples

PRACTICAL CONSIDERATIONS:

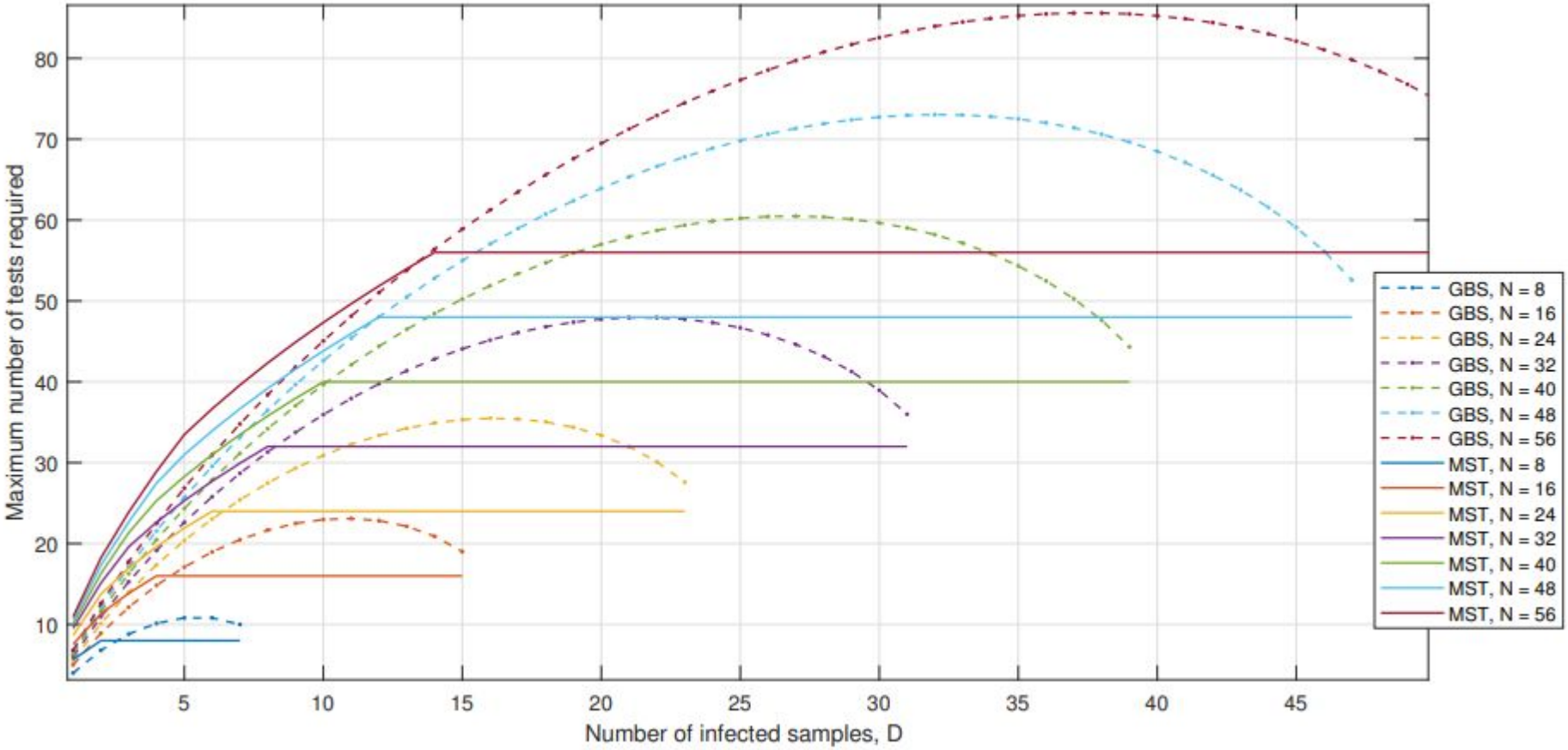
- Since it is a mix of adaptive and non adaptive schemes, it is **fairly fast**.
- The maximum number of times a sample will bes tested is s. So a swab will get divided into atmost **sr samples** where r is the number of replications
- **Max tests** in MST is always N (reference to the delta formula)
- The groups that test positive are tested at least s times, **reducing false positive rate**.

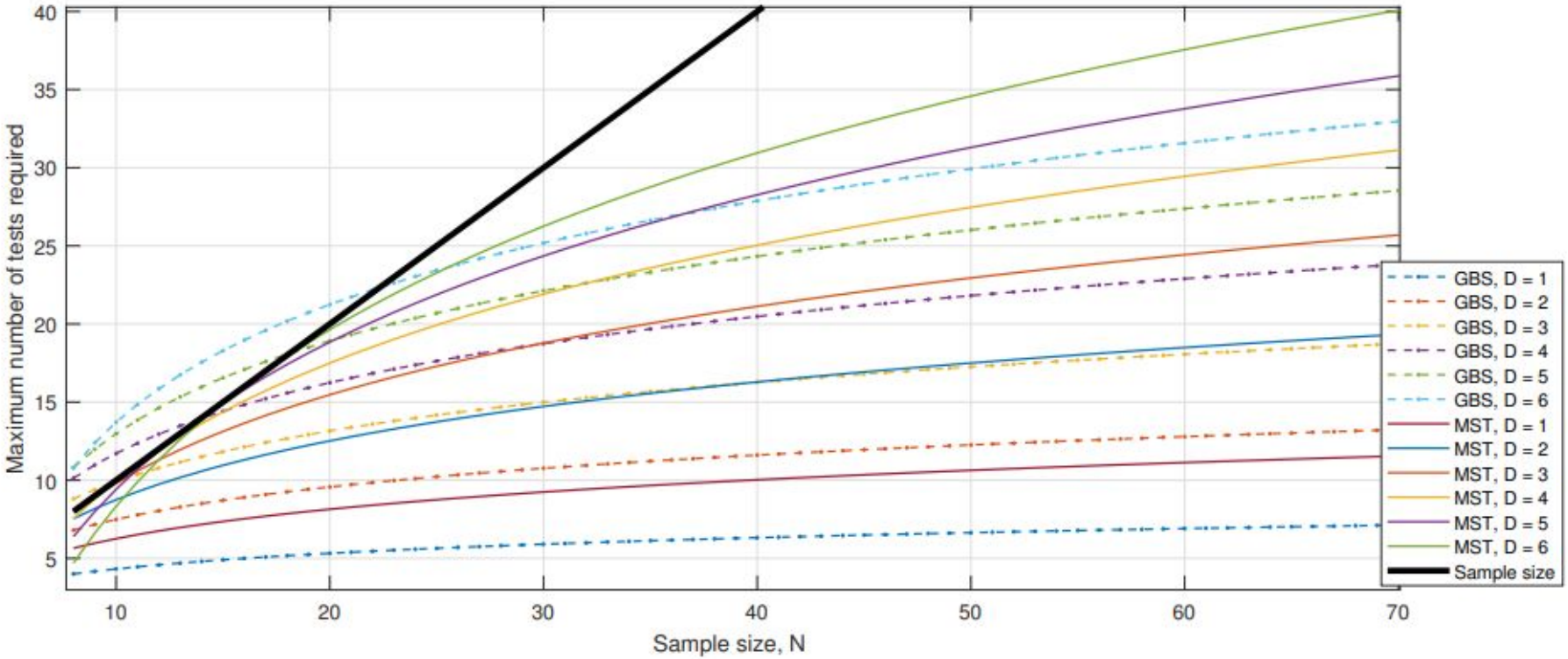



(a) Optimal number of stages in MST.




(b) Worst case number of tests required by MST with double replication.


Maximum number of tests required
Number of infected samples, D
10
20
30
40
50
60
70
80
5
10
15
20
25
30
35
40
45
GBS, N = 8
GBS, N = 16
GBS, N = 24
GBS, N = 32
GBS, N = 40
GBS, N = 48
GBS, N = 56
MST, N = 8
MST, N = 16
MST, N = 24
MST, N = 32
MST, N = 40
MST, N = 48
MST, N = 56




As the prevalence goes on decreasing, **GBS becomes more efficient than MST.**

**NESTED TESTING:**

In CGT algorithms, the value of D plays a pivotal role. In case D is underestimated, the total number of tests can easily become suboptimal.

The NT algorithm takes N and α as the input and provides an adaptive test plan that minimizes the average number of total tests required to diagnose N samples. **The actual number of infected samples in a pool or its estimate is not required for NT, only the prior probability of the presence of an infected sample is needed.**

In this algorithm, the total cases are divided into 3 bins: Undiagnosed bin (UB), Diagnosed bin (DB), and Potentially infected bin (PIB).

If you know α, you know the P(D=i) as a binomial distribution. In the worst case scenario, when the **actual number of infected is the least probability**, the algorithm can become suboptimal. But low prob of this. The average number of tests is given by

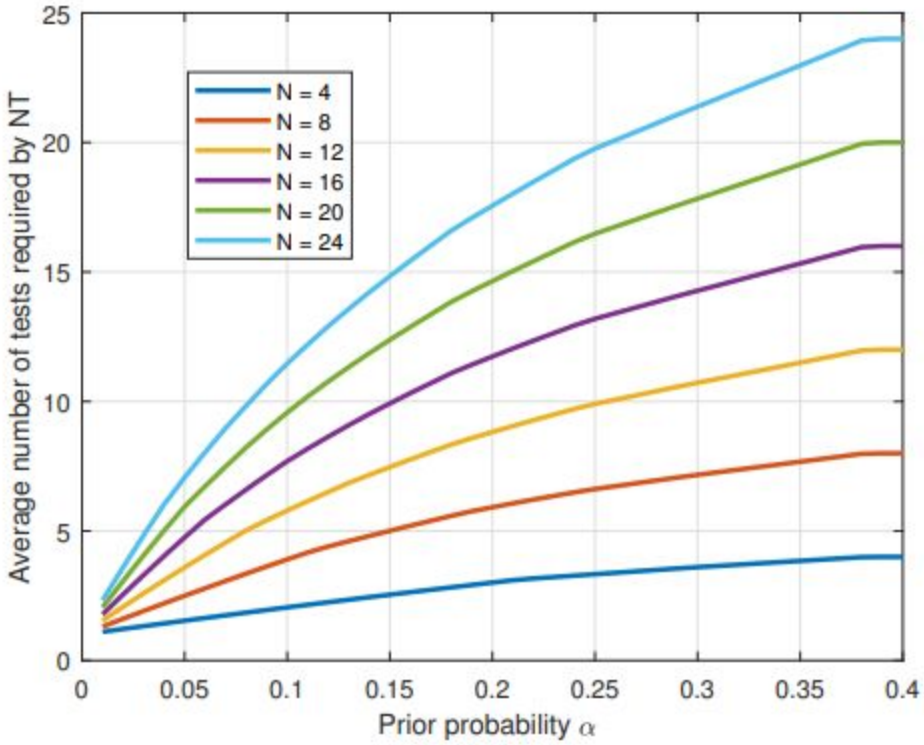



Figure 9: Average number of tests required by nested testing for different prior probabilities.

**Algorithm 3** Nested Testing

**Input**: $N, \alpha$ and samples pool $\mathcal{U} = \{1, 2, \cdots, N\}$.
UB $= \mathcal{U}$; PIB $= \mathcal{P}$ and DB $= \mathcal{D}$ are empty
**while** $\mathcal{U}$ is not empty **do**
    Compute $h = \tilde{G}(0, |\mathcal{U}|)$ using (8)
    Test a group $\mathcal{G} \subseteq \mathcal{U}$ of size $h$
    Update $\mathcal{U} = \mathcal{U} - \mathcal{G}$
    **if** Test is negative **then**
        Update $\mathcal{D} = \mathcal{D} + \mathcal{G}$
    **else**
        Update $\mathcal{P} = \mathcal{G}$
        **if** h == 1 **then**
            Update $\mathcal{D} = \mathcal{D} + \mathcal{G}$ and make $\mathcal{P}$ empty
        **else**
            **while** $\mathcal{P}$ is not empty **do**
                Compute $g = \tilde{G}(|\mathcal{P}|, |\mathcal{P} \cup \mathcal{U}|)$ using (8)
                Test a group $\mathcal{G} \subseteq \mathcal{P}$ of size $g$

**if** Test is positive **then**
Update $\mathcal{U} = \mathcal{U} + \mathcal{P} - \mathcal{G}$ and $\mathcal{P} = \mathcal{G}$
**else**
Update $\mathcal{D} = \mathcal{D} + \mathcal{G}$ and $\mathcal{P} = \mathcal{P} - \mathcal{G}$
**end if**
**if** $|\mathcal{P}| == 1$ **then**
Update $\mathcal{D} = \mathcal{D} + \mathcal{P}$ and make $\mathcal{P}$ empty
**end if**
**end while**
**end if**
**end if**
**end while**

$$\tilde{G}(0,n) = \arg\min_{x=1,\cdots,n} \hat{G}(0,n,x) \qquad \text{and} \qquad \tilde{G}(m,n) = \arg\min_{x=1,\cdots,m-1} \hat{G}(m,n,x),$$

$$G(0,n) = \min_{x=1,\cdots,n} \hat{G}(0,n,x) \qquad \text{and} \qquad G(m,n,x) = \min_{x=1,\cdots,m-1} \hat{G}(m,n,x),$$

$$\begin{aligned}
\hat{G}(0,n,x) &= 1 + (1-\alpha)^x G(0,n-x) + (1-(1-\alpha)^x) G(x,n), \\
\hat{G}(m,n,x) &= 1 + \frac{(1-\alpha)^x - (1-\alpha)^m}{1-(1-\alpha)^m} G(m-x,n-x) + \frac{(1-(1-\alpha)^x)}{1-(1-\alpha)^m} G(x,n), \\
G(1,m) &= G(0,m-1), \qquad G(0,1) = 1, \qquad G(0,0) = 0.
\end{aligned}$$