Optimization of Pool Testing

Emily Parnell, Maria Parnell

Advisor: Gianluca Guadagni

December 2020

1 Abstract

This study had two objectives: first, to study pool testing combinatorically and determine what percent of a population could be sick before it no longer was effective to utilize pool testing, and second, to develop an algorithm to optimize pool testing for a population with an ambiguous level of infection. In the first part, we iterated through every possible infected percentage and determined which pool size, if pool testing was to be used at all, would save the most tests. We also considered a new model of pool testing where the pools themselves might be split into subpools in the case that they initially test positive, rather than testing each person in the pool individually, as is customary. For these we looked at the effectiveness when splitting into set divisions - like halves, thirds, or fourths - at every level of pooling to better understand the cost advantages of pooling pools. We also considered the time delay associated with this combinatorics model. The combinatoric model helped demonstrate the power and drawbacks of pool testing, but in a realistic situation where one is testing a population, the percent of people infected would not be known, and an estimate that is even slightly off could drastically affect the course of action that should be used. Additionally, the time delay for participants' test results can become substantial with multi-level pooling. When creating an actual planning tool that can be used to administer pool testing, we capped the testing at three rounds (pool, subpool, and individuals), assumed the user would have a rough estimate of the percent infected in their population (p), used this estimate to determine an optimal pool size (n) and subpool size (nprime), and demonstrated that the method was effective for a range of p values even if the user's estimate of p was not accurate.

2 Introduction

As universities, secondary schools, and even larger geographic regions across the globe try to balance the return of normal life and mitigating the spread of the COVID-19 virus, regular testing within any group of people becomes a valuable tool to both track infection rates and prevent further spread. Lack of available COVID-19 tests, however, has proved to be an obstructing factor for these endeavors. Many universities have turned to pool testing, which allows them to combine samples from various people and test the entire group with one test, to overcome this limitation. In this paper, we seek to mathematically explore the true benefits of pool testing, which depends on the infection rate of a population and pool sizes, as well as develop new methods for optimizing the pool testing procedure so the greatest possible number of people can be reached in a time effective way with the fewest possible number of tests.

3 Minimizing Testing Cost

Previous research has found that pool testing is cost effective as long as the percent of the population which is infected remains low enough (Cherif et al., 2020). The probability of a pool returning negative is found by multiplying the chance of a single person testing negative (q) for each person in the pool (n being the number of people in the pool): q^n. If a pool tests negative, only the single test used on the pool is needed, but if a pool tests positive, all of the people in the pool are tested individually, and $n + 1$ tests are used.

$$p = \text{probability person is infected}$$

$$q = 1 - p = \text{probability person not infected}$$

$$n = \#\ \text{people in pool test}$$

$$q^n = \text{probability pool test negative}$$

Thus the expected number of tests needed to check everyone in a single pool would be:

$$1 \cdot q^n + (n+1)(1-q^n) = n(1-q^n) + 1$$

as opposed to n tests needed to check each person individually without pooling.

Applied to an arbitrary population size of 1000, we first found which pool size, n, would minimize the cost, or number of tests needed to test the whole population, given a varying estimate of p, the percent of the population currently infected. We then determined the corresponding cost for each optimal n. A Python script was written to solve this numerically. Nested loops were used to determine the optimal pool size, n, to minimize tests for each p from 0.001 to 0.999 (incrementing by 0.001), considering n values of $2 - 1000$ along with the case of no pool testing ($n = 1$). The total cost for each n was found by calculating the cost of each pool (equation above) times the number of pools that would be needed to cover the population, $\frac{1000}{n}$.

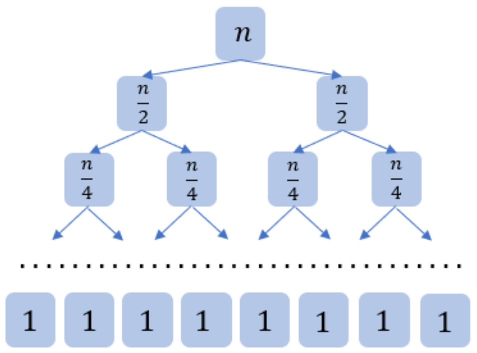
$$\text{Total cost (\# of tests)} = \frac{1000}{n}(n(1-q^n) + 1)$$

We then investigated whether we could improve the efficiency of the pool testing by splitting the pools themselves into subpools, so that when a pool comes back positive, the subpools could be sampled instead of immediately testing each person in the pool individually.

To add the cost contribution of each level of pooling to the total cost, we found a new p, or average probability that each person might be sick, at each new level of pooling since if a pool came back positive this would be higher than the probability for the original, larger population. The expected probability of an individual being infected in a pool that tested positive would be:

$$p_1 = \frac{p}{1 - q^n}$$

To model this, values of p and n were iterated through to find the best n and associated cost again. However this time, the cost was calculated differently to reflect the potentially nested pools. We first considered the case where each positive pool would be split into two subpools, where the positive subpools themselves would be split into two smaller subpools, and continuing this pattern until each pool consisted of 1 or less people (people were considered a continuous quantity in this calculation to accommodate this recursive structure). This was repeated with a splitting value of 3, 4, and 5. Figure 1 explains how cost is calculated when recursively pooling pools, and Figure 2 shows the savings of pool testing alone compared with pooling pools.



Average Cost to Test Single Pool

Cost at level	*	**Probability level need be tested**
1	*	1
+2	*	$1 - q_0^n$
+4	*	$(1 - q_0^n)(1 - q_1^{\frac{n}{2}})$
+.....		
$+(s^d = n)$	*	$(1 - q_0^n)\left(1 - q_1^{\frac{n}{2}}\right)\ldots\left(1 - q_{(d-1)}^{\left(\frac{n}{2^{(d-1)}}=2\right)}\right)$

$d = number\ of\ divisions \quad s = number\ of\ groups\ split\ into$

Figure 1: Recursive Visualization: The number of tests that must be conducted at each level of testing

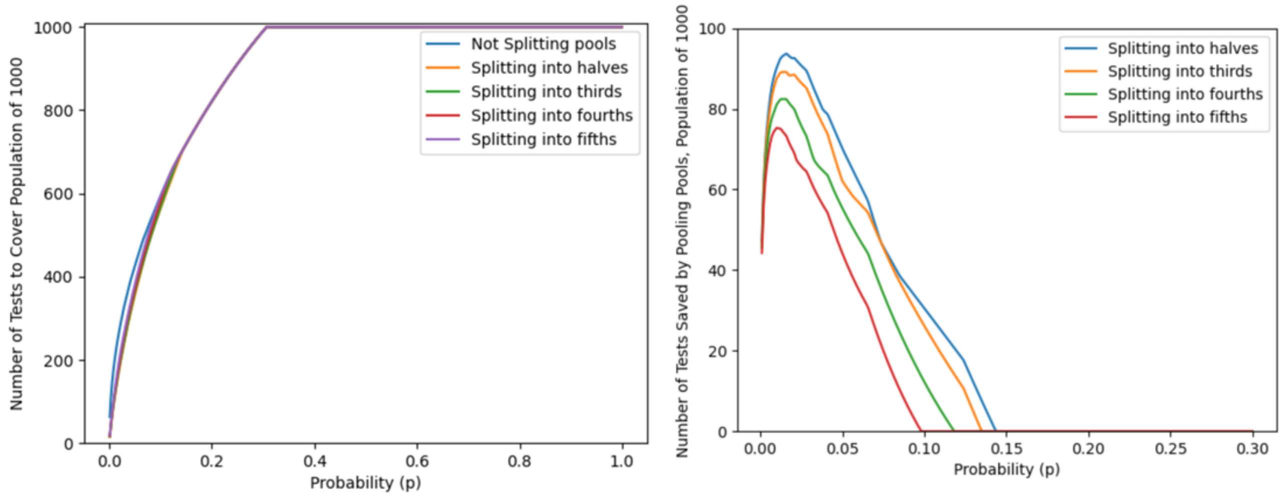


Figure 2: Advantage of Pool Testing and Pooling Pools

From these graphs we can conclude that pool testing holds a cost advantage over no pooling up until around 30% of the population is infected, and that subpooling holds a cost advantage over single level pooling until around 15% of the population is infected.

4 Considering the Time Cost

Another concern to address is that nested pool testing may cause a substantial delay in the amount of time it would take people to receive their test results, slowing the rate at which tests are administered; we referred to this as coverage speed. We considered the case where there were a finite number of tests available each day, and an organization must maximize the number of people they can test and the speed at which they can accomplish this. The total number of tests available each day (k) could be split into two categories, retests (r) and new tests (w) on any given day, or other arbitrary unit of time (i).

$$k = r_i + w_i$$

We are interested in w_i, the number of new tests we "reach" on a given day, which is affected by how many retests must be used on positive testing groups from the previous day:

$$w_i = k - r_i = k - w_{i-1}n(1 - q^n)$$

In the infinite limit as time i grows, w converges to:

$$w = \frac{k}{n(1 - q^n) + 1}$$

But as mentioned, with regular single-level pooling, some of these people receive their tests after two cycles. We can find the average wait time for a person as the percentage of people in a negative pool times one cycle plus the percentage of people in a positive pool times two cycles, or:

$$\text{test time} = q^n + 2(1 - q^n) \text{ cycles}$$

which will always be more than the time without pool testing of 1 cycle, given $q < 1$. Putting together the average number of people reached in a round of pool tests and the average time a person has to wait with pool testing, we can find the average rate that pool tests are administered as:

$$\text{coverage speed} = \frac{\frac{kn}{n(1-q^n)+1}}{q^n + 2(1-q^n)} \text{ people / cycle}$$

A similar Python program was run on single level pool testing, but instead of minimizing cost, we maximized coverage speed (# of tests each cycle/average test wait time). This parameter was only considered for the case of single-level pool testing. This can be seen in Figure 3.

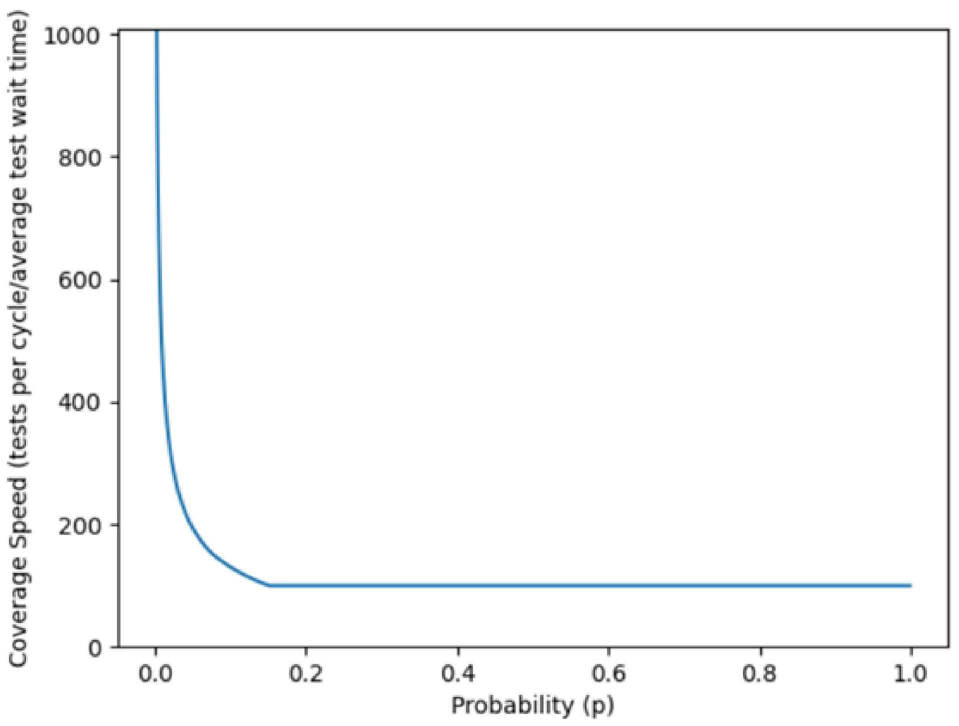


Figure 3: Coverage speed as a function of percent infected (p). This shows that for lower infection rates with single-level pooling, the entire population can be covered faster than testing individually.

We decided to determine how the average test wait time would be affected by pooling pools beyond one level. To intuitively think about this, consider a person's "recursion depth" (visually represented as vertical height in Figure 1) to represent the number of times their sample has been tested (including retests). The expected value of wait time for any random person, where one unit of time represents the length of time of one round of testing, can be calculated by summing the probability of a person advancing to each deeper recursion level until there are not deeper expected recursion depths (each person has either been in a negative or has been tested individually). As the recursion depth gets deeper, the probability of a person reaching that level gets smaller. The expected wait time can be represented as

$$\text{expected test wait time} =$$
$$q_0^n + 2(1-q_0^n)(q_1^{n/s}) + 3(1-q_1^{n/s})(q_2^{n/s^2}) + \ldots + (d-1)(1-q_{d-3}^{n/s^{d-2}})(q_{d-2}^{n/s^{d-1}}) + d(1-q_{d-2}^{n/s^{d-2}})$$

where d represents the recursion depth. It can also be noted that each term in this equation represents the people who do not advance past that recursion depth, not every individual who reaches and passes beyond that depth. The last term does not include a qualifier to limit the term to only include those who test negative at this level, because each person will be tested anyway at the deepest recursion level.

The expected test wait time given in testing cycles for pool testing a single level is given in Figure 4, while the wait time when pooling pools by splitting each pool and subpool in half is seen in Figure 5. Note how both of these flatline around 30% where pool testing is no longer effective.

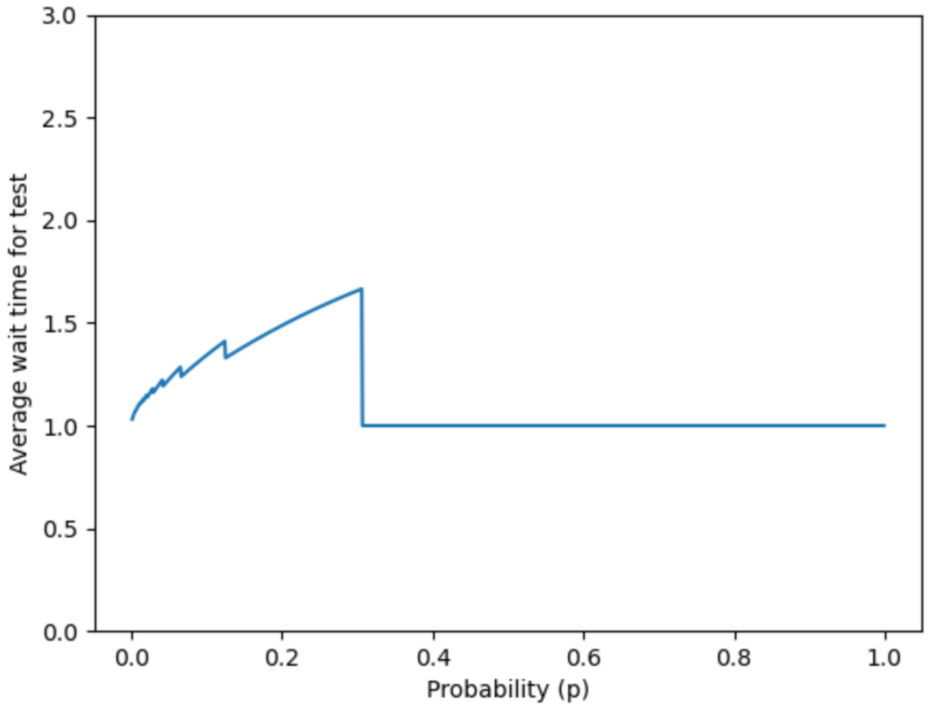


Figure 4: Expected wait time for a test using one level of pool testing.

While single level pooling has only a minimal delay, the time delay when pools are split in halves to form multiple levels of subpools is substantial, especially for the low levels of infection which are most likely to be experienced by a population in reality.

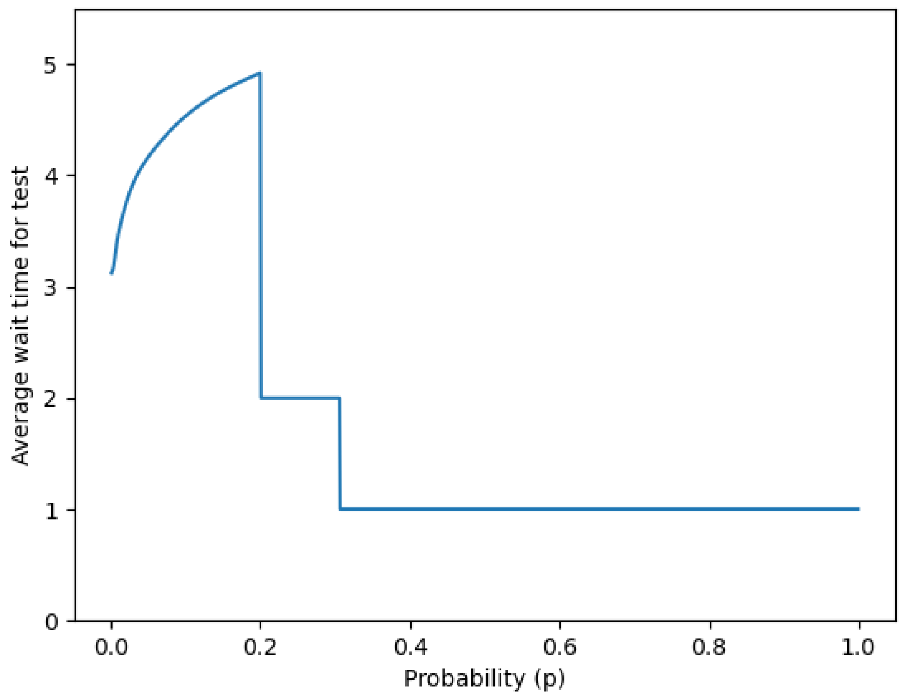


Figure 5: Expected wait time for a test result when splitting pools and subpools in halves to optimize the test savings.

5 Exploring A Proposed Testing Method

Given what we've learned through our analysis of pool size and nested pooling, and its effectiveness in relation to infection rates and coverage speed, we found it fitting to condense our results into an optimized, practical COVID-19 testing approach to be employed by universities, companies, or any other testing effort by a large organization. Though we have analyzed the cost of testing time more extensively above, it became apparent to us that, in reality, having to wait more than three testing periods for the results of any one test is extremely undesirable. We began our design under this limitation and hoped to show that in general, one level of subpooling would be enough to reap most of the cost benefit without extending the testing time too much.

To accommodate this decision, we first decided to split the population to be tested into regular pools of size n. This is similar to the common pool testing approach, although improved, as we can select semi-optimized pool sizes based on a provisional estimate of positivity rate. Beyond this, we must only conduct two more rounds of testing on the positive pools to remain without our 3-period limit. With this in mind, it only makes sense to split this pool into reasonable small pools, as opposed to splitting it in, say, half, as was previously analyzed, because if a pool has tested positive in the first place, then at least one of its halves of size n/2 would be guaranteed to come back as positive, requiring that all members of that large subpool be tested individually in the third round. Instead, we decided to divide the pool, of any size n, into small groups and observe the result. For this part of our analysis, we compared the expected testing cost of splitting each pool into subpools of size 1 (control) through 9 people to learn more about which subpool size is optimal.

To test the approach of sub-testing a positive pool, we focused on the second-level pooling by writing a script to numerically iterate through all realistic pool sizes (n) ranging from [2,50], and all ratios of infected to total people within the positive pool from 1 person infected to half the people infected, $[(1/n), ((n/2)/n)]$, since our earlier work found that pool testing would never be effective if more than ∼30% of the population was infected, and split the pool into subpools of size 2 through 9. For each of these positive first-level pools, we calculated how many tests were expected to fully test that pool. This was done stochastically, by randomly creating an array of size n filled with the correct amount of positive and negative people, and then checking every group within the array to find which "subpools" would come back positive. Averaging the expected tests used for all the p values (infection rates) for each pool size n was used to get the test ratio for the pool, # tests/n.

A test ratio greater than 1 would be indicate that subpooling was less effective than individual testing, while a test ratio less than 1 would indicate cost savings. The test ratios for various subpool sizes were plotted against each other as seen in Figure 6. Recall that these ratios are relative to the control case of testing all members of the pool individually (subpool size of 1, which generates a testing ratio of 1). While the most optimal subpool size varies with the positivity rate, the optimal subpool size appears to never get higher than 6.

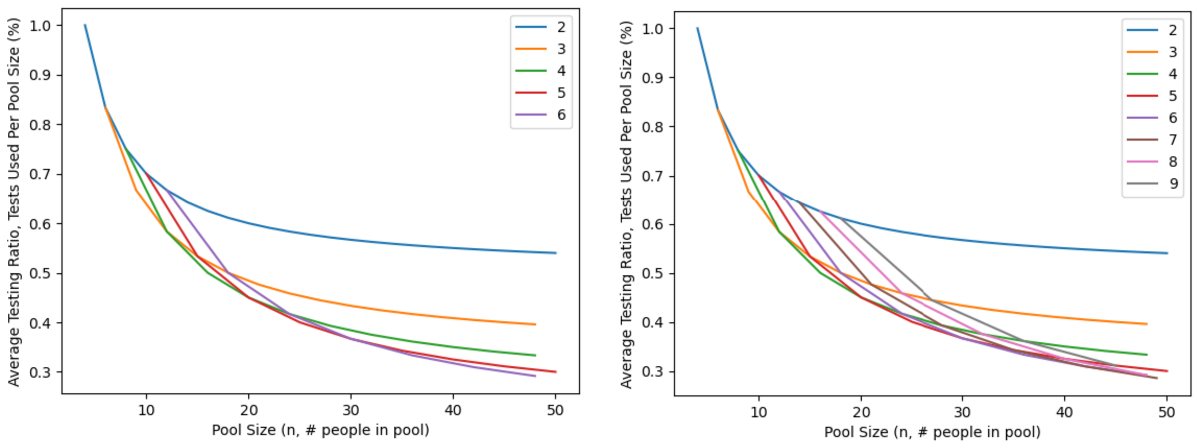


Figure 6: These graphs show the average testing ratio (averaged over a range of percent infected) for various first-level pool sizes, n. It can be observed in the left graph that a subpool size of 6 is marginally more effective than 5 for some pool sizes. The graph on the right shows how increasing to subpool sizes above 6 provides little to no benefit.

6 Optimized Testing Approach

This section will provide a condensed description of the optimized testing approach proposed in this paper, as supported by the mathematical evidence explored above.

The algorithm was considered for testing a population of 1000 people, iterating through the possibilities of 1 through 250 people being sick (corresponding with 0.1% to 25% infected). We considered all n sizes (the first level of pooling) from 2 through 50 and stochastically determined on average how many pools would be infected after the first round. This was used to determine on average how many infected people there might be in each infected pool, by dividing the number of infected people by the number of infected groups. Then we considered all possible values of nprime (second level of pooling), from 2 through n/2. We stochastically determined how many of the subpools would come back positive on average. From here the testing cost for the particular pair of n and nprime on the infected level of the population was calculated as cost = # of pools + # infected pool * # of subpools + # infected subpools * nprime (size of subpool). If the cost for this combination was better than all previous nprime and n combinations for this infected level, and it was better than the base cases of one level of pooling or no pool testing, then the n, nprime, and cost were saved.

Note that it was assumed that any remainder when a pool size or subpool size did not go into the population evenly would form its own smaller pool (eg a remainder of 4 would become an extra pool of size 4). This was accounted for when finding the total number of pools or subpools in calculating cost.

Figures 7 and 8 show the optimal n and nprime selected with this approach. Figure 9 compares the cost of our method as compared with a traditional pool testing method with a fixed, single layer of pool testing of pool size 10. Figure 10 shows the number of tests saved by using our method as opposed to this traditional method.

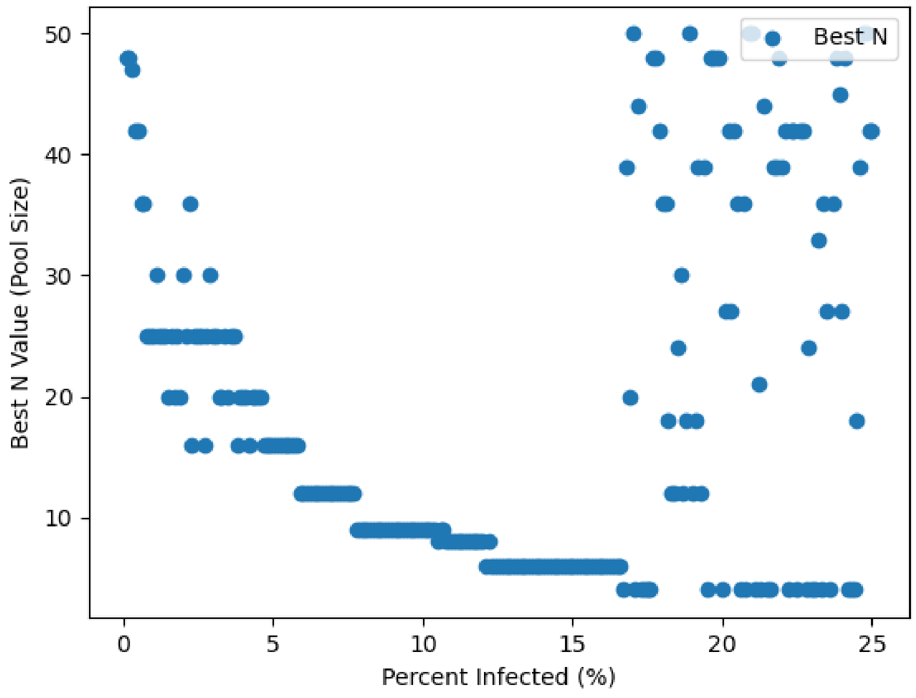


Figure 7: Shows a display of selected optimal first-level pool sizes. Note that at high percentages of an infected population, the model begins to break down. However, such a high prevalence of infection within the population is very unlikely to occur in reality.

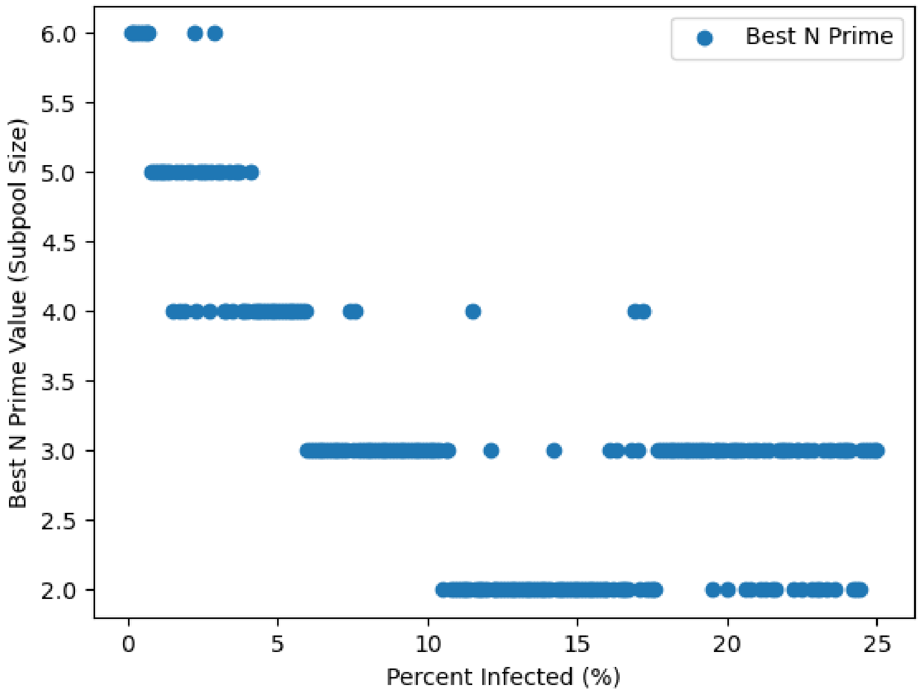


Figure 8: Shows a display of selected optimal subpool sizes. Note that while it varies, it tends to gravitate to around 2 - 5.

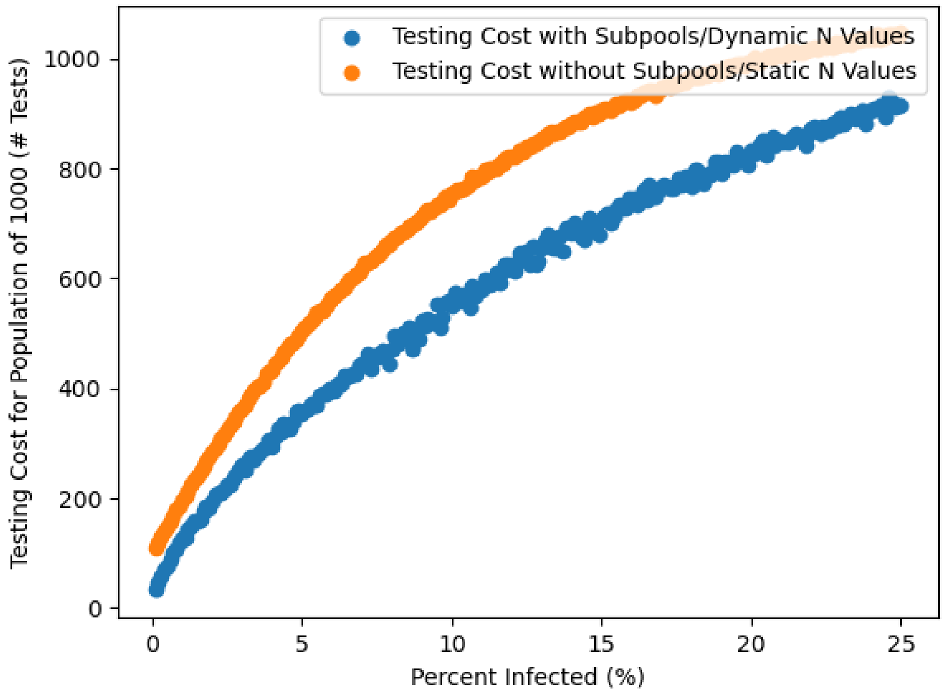


Figure 9: Comparison of testing costs between optimized and traditional pooling methods

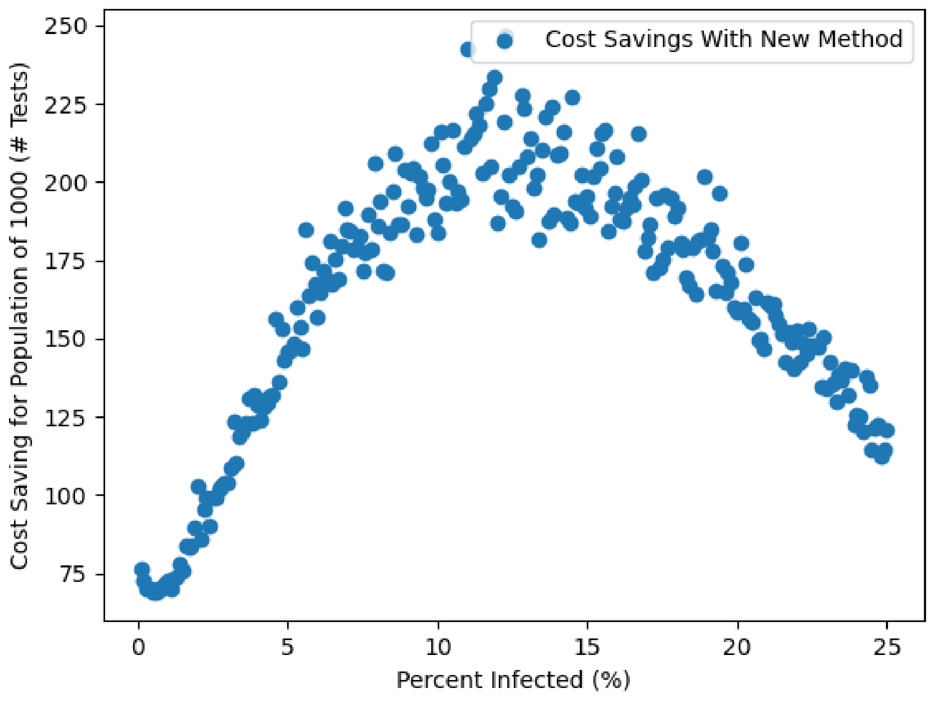


Figure 10: Number of tests saved by using optimized approach at each level of percent infected

Clearly, selecting an optimal pool size, n, as well as an optimal subpool size, nprime, significantly reduces the number of tests needed to cover an entire population.

7 Simulation Testing of Optimized Method

Though it has been shown that the proposed optimized testing approach can greatly reduce testing costs, especially for populations where the percent of people infected is under 15%, the previous models present situations where the estimated p value of the population is perfectly accurate. The purpose of this section is to demonstrate the efficacy of the optimized model even in situations where the estimated p value differs from the actual p value. It supports the claim that the model works flexibly and consistently to reduce testing costs, at the sacrifice of one extra testing period. Additionally, this effort provides values to the model by demonstrating cost savings on a random population which was not used in the prediction of optimal n and nprime values.

Furthermore, the associated code provides a function which may be utilized by those wishing to decrease testing costs. More specifically, the function 'find_best_pool_size(population, p_estimate)' takes two arguments, representing the number of people who are to be tested in the given batch and the estimated percent of the population infected, respectively, and returns the optimal values for n (first-level pool size) and nprime (subpool size). This provides users with all of the necessary tools to conduct our proposed testing approach.

In order to show that our approach can come up with accurate predictions even in random situations, we separated out the process of predicting the optimal n and nprime values and the actual application of these predictions on a random, concrete population. One "simulation" takes in a population size, a p_estimate (the infection rate assumed by the testers), and a p_actual (a guideline for the infection rate of the actual simulated population). First, the simulation predicts the best n and nprime values using 'find_best_pool_size(population, p_estimate)' in the same way any user of our model could. Separately, a concrete population is created which adheres to the infection rate of p_actual, where the infected individuals are randomly distributed across the population.

Then, given this concrete, random population, as well as the "optimal" n and nprime values, the actual cost to test each individual for this specific population is calculated using double-level pool testing. To compare, the cost to test every individual in the same population using traditional, single-level pool testing with a static pool size of 10 is also found.

First, to show the efficiency of the model with varying levels of the population infected, we first assumed a completely accurate p_estimate. Figures 11 and 12 show the consistent reduction of testing costs over infection rates of 0.1% - 1% and 1% - 15% using the optimized approach.

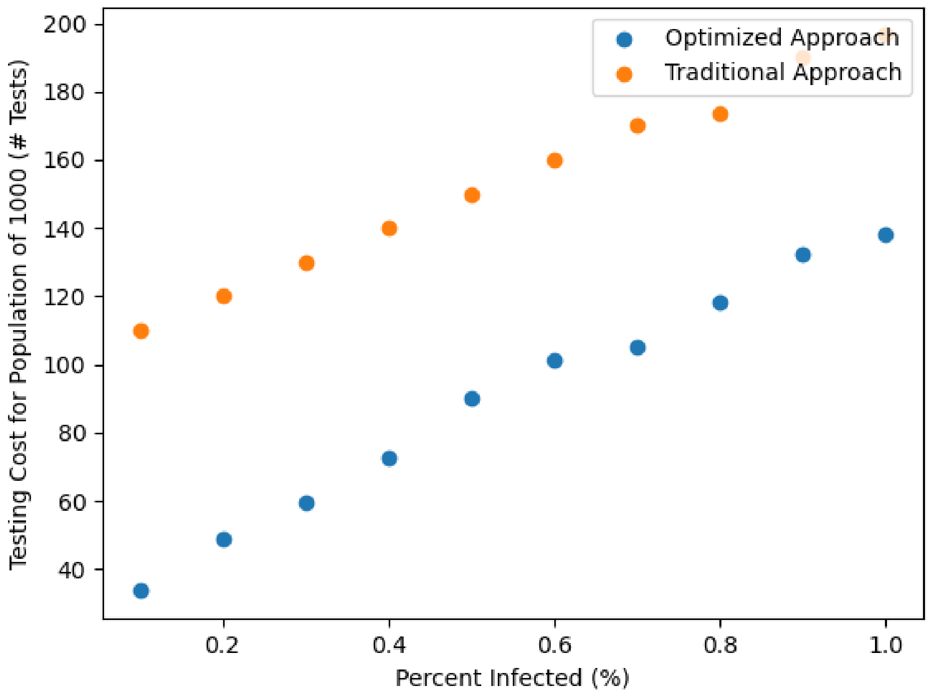


Figure 11: Compares the testing costs of the optimized and traditional pooling approaches for a population of 1000. P_actual and p_estimate are set equal to indicate an accurate prediction, and they are varied from 0.1% - 1%.

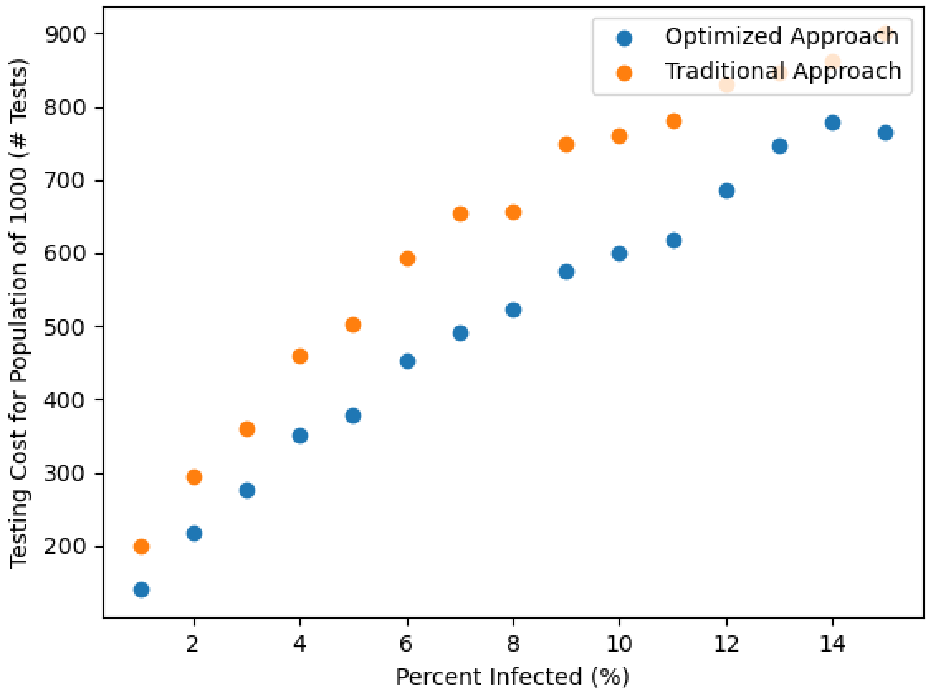


Figure 12: Compares the testing costs of the optimized and traditional pooling approaches for a population of 1000. P_actual and p_estimate are set equal to indicate an accurate prediction, and they are varied from 1% - 15%.

Then, to show the flexibility of the approach, we took into account the fact that users may not have an accurate value for p_estimate. We graphed situations where the p_actual was fixed and the p_estimate varied greatly.

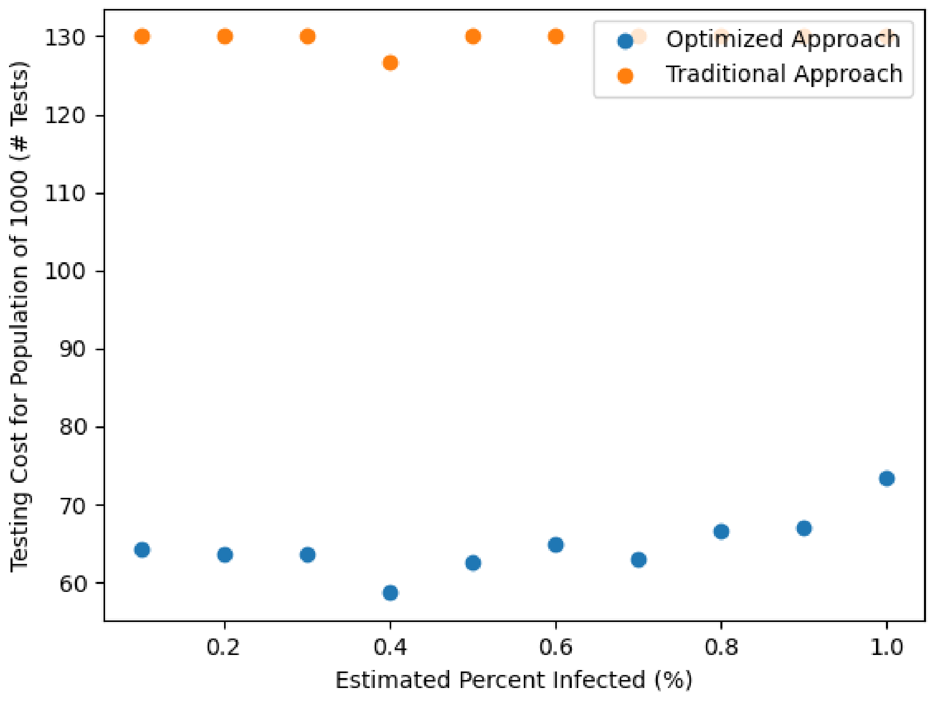


Figure 13: Compares the testing costs of the optimized and traditional pooling approaches for a population of 1000. A static p_actual of 0.3% was used, and p_estimate was varied from 0.1% - 1%.

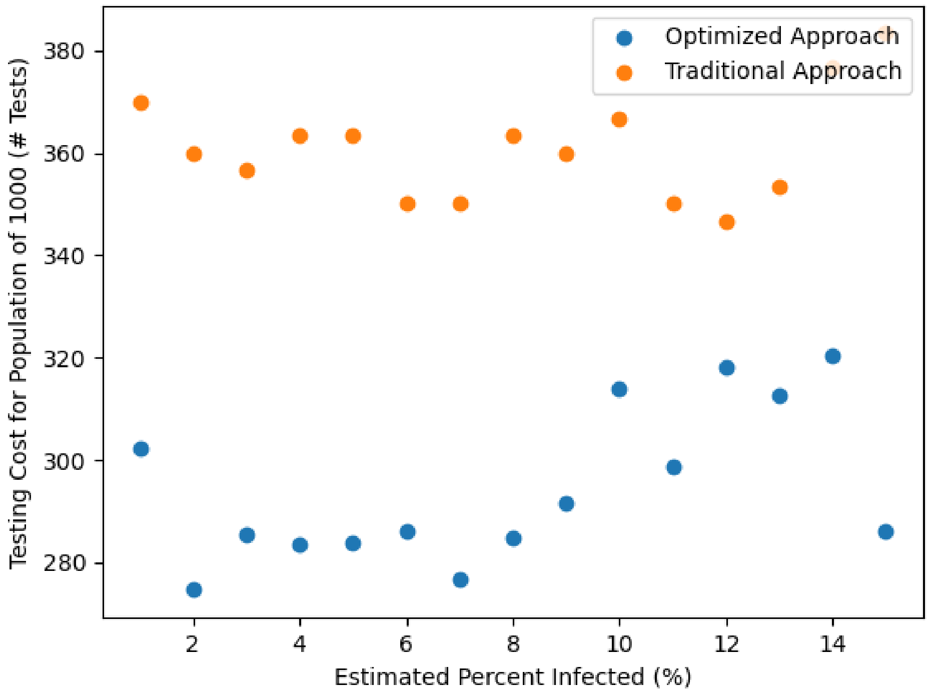


Figure 14: Compares the testing costs of the optimized and traditional pooling approaches for a population of 1000. A static p_actual of 3% was used, and p_estimate was varied from 1% - 15%.

It becomes apparent that the resulting testing costs for both testing methods is more dependent on p_actual than p_estimate, due to the relatively unchanging testing costs in Figures 13 and 14. An exception occurs as p_estimate becomes wildly inaccurate; as p_estimate becomes about 7% inaccurate, the testing costs suffer. Even still, the optimized approach is more effective than the traditional pooling approach. This is encouraging, because it is impossible to expect users of the model to possess extremely accurate estimates of the infection rate.

It should be noted that each data point in Figures 11-14 is produced as the average of three runs of the simulation on randomly generated populations (within adherence to the selected p_actual), so these exact graphs should not be directly reproducible, and slight deviations by individual data points do not necessarily hold any importance. However, while each run of the simulation will produce slightly different results, general trends can still be observed.

8 Conclusion

Our plots demonstrated that pool testing was generally more cost-effective as long as no more than $\sim 30\%$ of the population was sick. Pooling pools held an edge over plain pool testing for populations infected no more than $\sim 15\%$. In the combinatoric model where pools and subpools were split recursively, for 5% of a population sick, dynamic single-level pool testing could use only 42.6% and dynamic subpooling only 35.6% of tests compared to testing all individually. Additionally, single-level pools could cover a population more quickly when the infected population was below $\sim 15\%$ (in terms of people being tested over the average wait time). While they are efficient in terms of cost, the recursive splitting of subpools results in a large wait time - more than three test cycles for a realistic percent infected.

The number of variables to take into account while designing this testing plan is great enough that it is impossible to devise one sturdy "perfect optimization" guided by nothing but fact; however, we have come up with a solution based on a fairly informed opinion on testing cost, optimal pool size, and testing speed, guided by model-based analysis of which variables are important, as well as a sensibility for what is realistic and desirable in application. Furthermore, in the case that a specific organization holds certain variables as more important, this plan can be adjusted accordingly to fit those needs.

In our method, we wanted to balance the wait time and the test savings. Capping the testing to three rounds, we wrote an algorithm to find the optimal n and nprime for a given population size and percent of the population infected. Our method improves upon the current method of pool testing, saving roughly 25% of tests by comparison for up to slightly over 10% infected. As the gains from subpooling diminish with a high infection rate, the gap between the two methods closes, but our method still consistently outperforms the traditional one due to the dynamic pool size. Additionally, we demonstrated how our method holds its advantage even when the user is unsure of the actual percent of their population which is infected. Keeping the actual infection rate static, we adjusted the "estimated" infection rate, and the optimized n and nprime value still created a model which consistently outperformed traditional pool testing.

In the future, this investigation could consider the reality that the tests themselves do not have perfect accuracy, as is assumed here. This could potentially impact the optimal pool sizes selected for efficiency.

9 References

Cherif, A., Grobe, N., Wang, X., & et. al. (2020). Simulation of pool testing to identify patients with coronavirus disease 2019 under conditions of limited test availability. *JAMA Network Open*. doi: 10.1001/jamanetworkopen.2020.13075