Optimization and Expansion of an Approach to Group Testing

Christy Callear
SUNY Geneseo

The Undergraduate Biomathematical Research Career Initiative*
Faculty Mentors: Dr. Anthony J. Macula (Mathematics), Dr. Gregg Hartvigsen (Biology)
*supported by The National Science Foundation-UMB Project 0436298

The Problem

We are given a graph, to which we will add several random edges. The goal of our group testing algorithm will be to identify these added edges, which represent a relatively small portion of the whole graph, by performing the same test on each vertex a large number of times.
Methods

We have now added three complexes of degree 2: (4,7), (1,7) and (5,10).

A Maple program written by Niels Hansen would then conduct a series of tests that consist of selecting random subsets of the vertices in this graph by selecting each vertex with a given probability.

Applications

Scientists need libraries of single strands of DNA that will NOT combine. However, DNA often binds in unexpected ways, other than the typical A-T, G-C pairing. Our goal is to find these unexpected combinations of DNA (think of it as an added edge) so we can annihilate at least one of the pair to create a useful library.

5’–T–G–G–T–3’

5’–A–C–A–C–A–3’

5’–A–C–A–C–A–3’
Interpreting the Output

This is an example of a possible output with the probability of selection set at 0.72.

Positive and Negative Tests

A **positive test** is any in which for some target complex, each vertex incident to that complex is selected.

Consider the first test on the previous slide. Here all three complexes are completely identified because vertices 1, 4, 5, 7 and 10 all have ones in the first row.

<table>
<thead>
<tr>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
<th>V9</th>
<th>V10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

A **negative test** is any in which no targets are completely identified. Consider the second test on the previous slide. Neither (1,7), (4,7) nor (5,10) was completely identified.

<table>
<thead>
<tr>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
<th>V9</th>
<th>V10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Positive and Negative Tests, Cont.

In this example, positive tests have a one in the far right column, and that column is shaded to indicate which target was completely identified.

Our Goal

While executing a run of tests, for each vertex the program will keep track of the number of times that vertex was not selected in a negative test. At the end of the run, the program will identify those vertices whose counter is greater than the product of the total number of negative tests and a given cutoff value.

Counter > # Negative tests * cutoff value

We will manipulate the probability of selection in order to guarantee that we identify at least a given number of vertices from each complex and have a minimum number or errors.
Notation and Assumption

- \( p \) is the probability with which any vertex is selected in any test.
- \( k \) is the degree of the target complex, the number of vertices incident to the complex.
- \( d \) is the number of target complexes in the sample.

A vertex is “bad” if it is part of a target complex.
A vertex is “good” if it is NOT part of a target complex.

We have assumed that the target complexes are disjoint.

---

Notation and Assumption, Cont.

To return to this example:

- \( k = 2 \)
- \( d = 3 \)

Vertices 1, 4, 5, 7 and 10 are “bad”
Vertices 2, 3, 6, 8 and 8 are “good”

From here on complexes will be disjoint, so (1,7) and (4,7) will not both be added edges as in this example.
Useful Probabilities

In order to find an optimal probability, we must be able to compute and compare the probabilities that a vertex is or is not selected given that a test is positive or negative.

<table>
<thead>
<tr>
<th>v is &quot;bad&quot;</th>
<th>negative test</th>
<th>positive test</th>
</tr>
</thead>
<tbody>
<tr>
<td>v is not selected</td>
<td>$b_{0,0} = (1 - p)(1 - p^k)^{d-1}$</td>
<td>$b_{0,1} = (1 - p) - b_{0,0}$</td>
</tr>
<tr>
<td>v is selected</td>
<td>$b_{1,0} = (1 - p^k)^d - b_{0,0}$</td>
<td>$b_{1,1} = p - b_{1,0}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>v is &quot;good&quot;</th>
<th>negative test</th>
<th>positive test</th>
</tr>
</thead>
<tbody>
<tr>
<td>v is not selected</td>
<td>$g_{0,0} = (1 - p)(1 - p^k)^d$</td>
<td>$g_{0,1} = (1 - p) - g_{0,0}$</td>
</tr>
<tr>
<td>v is selected</td>
<td>$g_{1,0} = (1 - p^k)^d - g_{0,0}$</td>
<td>$g_{1,1} = p - g_{1,0}$</td>
</tr>
</tbody>
</table>

Useful Probabilities, Cont.

$b_{0,0} = (1 - p)(1 - p^k)^{d-1}$

- $p$ represents the probability of selection, so $(1 - p)$ is the probability that $v$ will not be selected
- $p^k$ represents the probability that an entire complex consisting of $k$ vertices is selected, so $(1 - p^k)$ is the probability that any given complex is not entirely selected
- since $v$ is not selected but $v$ is bad, the complex that $v$ is part of can clearly not be entirely selected, therefore $(1 - p^k)^{d-1}$ will ensure that none of the rest of the complexes are entirely selected, so the test is negative
More Useful Probabilities

What we really want to consider here is the probability of “success,” that a vertex is either selected in a positive test or not selected in a negative test. So we have:

\[ p_b = p - (1 - p^k)^d - 2(1 - p)(1 - p^k)^{d-1} \]

\[ p_g = p - (1 - p^k)^d + 2(1 - p)(1 - p^k)^d \]

\[ p_b - p_g = 2(1 - p)p^k (1 - p^k)^{d-1} \]

Finding the Optimal Probability

So the probability of selecting a bad vertex in a positive test or not selecting it in a negative test is greater than the probability of those events occurring with a good vertex. We will take advantage of this fact by maximizing this difference to find the optimal probability.

\[ p_b - p_g = 2(1 - p)p^k (1 - p^k)^{d-1} \]

We will then use
\[
N(p) = \left( \frac{z(N, t) \sqrt{p_b(1 - p_b)}}{t(p) - p_b} \right)^2
\]
to find the minimum number of tests, \(N\). This formula was computed by Dr. Macula.
Maximizing the Difference

We maximized the difference in our probability equation by setting the derivative equal to zero.

\[
\frac{2p^k(1-p^k)(d^{-1}(k- kp - p + p^{k+1} - kp^d + kp^{k+1}d))}{p(1-p^d)} = 0
\]

Since in a realistic system, \(p\) will never be equal to one or zero, we can disregard \(2p^k\) and \((1-p^k)^{d^{-1}}\). This leaves us with only \(k - kp + p^{k+1} - kp^d + kp^{k+1}d = 0\) to consider.

Results

Using this method, we calculated the following table.

<table>
<thead>
<tr>
<th>(k=2)</th>
<th>(d=2)</th>
<th>(d=3)</th>
<th>(d=4)</th>
<th>(d=5)</th>
</tr>
</thead>
</table>

We used \(z_s(N(p), t(p)) = -0.253\) and \(z_p(N(p), t(p)) = 3.090\), to obtain the minimum number of tests.

The data in this table is \([p, t(p), N(p), p_s, p_p]\).
The New Direction

Remember how we assumed that the complexes we were searching for were disjoint? In order to make our model more realistic we are now looking at complexes that form regular families. Regular families are groups of sets in which each individual set has the same cardinality, the intersection of each pair of sets has the same cardinality and in general the intersection of a given number of these sets always has a given size.

Regular Families

This is an example of a regular family, with $k_1 = |A_1| = |A_2| = |A_3| = 12$, $k_2 = |A_1 \cap A_2| = |A_1 \cap A_3| = |A_2 \cap A_3| = 4$ and $k_3 = |A_1 \cap A_2 \cap A_3| = 1$.
Finding the $k_i$'s and $j_i$'s

We noticed that it is not always possible to configure a regular family like the previous example from any given $k_i$'s. For example, there is no regular family with $[k_1, k_2, k_3] = [12, 4, 6]$. We know that each $k_i$ must be less than $k_{i-1}$. We also know that if we know how many vertices are 1-bad, 2-bad, ..., $v$-bad, call those numbers $j_i$, then the corresponding set of $k_i$'s can be found and vice versa.

Finding the $k_i$'s and $j_i$'s, Continued

Consider the general diagram to the left. We know that
\begin{align*}
j_1 + 2j_2 + j_3 &= k_1 \\
j_2 + j_3 &= k_2 \\
j_3 &= k_3
\end{align*}
and in turn can find that
\begin{align*}
j_1 &= k_1 - 2k_2 + k_3 \\
j_2 &= k_2 - k_3 \\
j_3 &= k_3
\end{align*}
Notice that each of the coefficients is from Pascal's triangle and also that each combination of $k$'s and $j$'s is unique!
New Equations

For this approach we need only one table of formulas. A bad vertex is now one that belongs to one or more complex while a good vertex is one that is part of zero complexes. Note that we say a vertex is “v-bad” if it belongs to v complexes. We now have:

\[
\begin{align*}
\text{v is “v-bad”} & \quad \text{negative test} & \quad \text{positive test} \\
\text{selected} & \quad P_{0,1}^v = (1 - p) \left( 1 - \sum_{i=1}^{d-v} d_i \sum_{j=1}^{v} p_{ij} (1 - p)^j (-1)^{v-1} \right) & \quad P_{0,0}^v = (1 - p) - P_{0,0}^v \\
\text{not} & \quad P_{1,0}^v = \left( 1 - \sum_{i=1}^{d-v} d_i \sum_{j=1}^{v} p_{ij} (1 - p)^j (-1)^{v-1} \right) - P_{0,0}^v & \quad P_{1,1}^v = p - P_{1,0}^v \\
\text{selected} & \quad P_{1,1}^v = P_{1,0}^v \\
\end{align*}
\]

Current Work

In our previous project, we noticed that

\[ b_{1,1} - b_{0,0} = g_{1,1} - g_{0,0} \quad \text{or} \quad P_{1,1}^i - P_{0,0}^i = P_{1,1}^0 - P_{0,0}^0 \]

We have found that this relationship will always hold because of the way the probabilities are determined.

\[ P_{1,1}^i - P_{0,0}^i = P_{1,1}^0 - P_{0,0}^0 \]

Now, following a similar route, we are working towards finding the optimal probability of selection by looking for a single p value that will maximize the difference

\[ \left( P_{1,1}^i + P_{0,0}^i \right) - \left( P_{1,1}^0 + P_{0,0}^0 \right) \]

and for all i,

\[ \left( P_{1,1}^i + P_{0,0}^i \right) - \left( P_{1,1}^0 + P_{0,0}^0 \right) \]

at the same time.
Results

In this graph, we used \( d=3, k1=12, k2=4 \) and \( k3=1 \) as in our example.

The green curve represents \( \left( p_i^1 + p_i^2 \right) - \left( p_i^0 + p_i^3 \right) \),

yellow- \( \left( p_i^1 + p_i^2 \right) - \left( p_i^0 + p_i^3 \right) \),

and red- \( \left( p_i^1 + p_i^2 \right) - \left( p_i^0 + p_i^3 \right) \).

It appears that \( p=0.85 \) would maximize each of these differences.

Results, Cont.

By looking at the derivatives, we see that in fact the curves have 3 slightly different optimal \( p \) values, ranging from 0.8 to 0.9.
Acknowledgements

I would like to thank

The SUNY Geneseo Undergraduate Biomathematical Research Career Initiative Program

The National Science Foundation-UMB Project 0436298

Our First Approach to Optimization

Ideally we would consider each vertex v, calculate the probability of obtaining k negative tests and then examine the distribution over all possible k. However, it is far simpler to consider the normal approximation to this binomial distribution.

Let N denote the number of tests, which we are trying to minimize

Let t be the threshold, which is given

Let \( z_b(N,t) \) and \( z_g(N,t) \) be the z-scores of the “bad” and “good” distributions respectively

We will also use \( \mu_b = Np_b \) , \( \sigma_b = \sqrt{Np_b(1-p_b)} \) , \( \mu_g = Np_g \) and \( \sigma_g = \sqrt{Np_g(1-p_g)} \)
We thought of these z-scores as parameters, allowing us to use these equations as follows:

\[ z_b(N,t) = \frac{Nt - Np_b}{\sqrt{Np_b(1-p_b)}} = \sqrt{N} \frac{t-p_b}{\sqrt{p_b(1-p_b)}} \]

and

\[ z_g(N,t) = \frac{Nt - Np_g}{\sqrt{Np_g(1-p_g)}} = \sqrt{N} \frac{t-p_g}{\sqrt{p_g(1-p_g)}} \]

or

\[ z_b(N,t)\sqrt{p_b(1-p_b)} = \sqrt{N} = z_g(N,t)\sqrt{p_g(1-p_g)} \]

We can then solve for \( t(p) \) and substitute to obtain:

\[ N(p) = \left( \frac{z_b(N,t)\sqrt{p_b(1-p_b)}}{t(p) - p_b} \right)^2 \]
Results

We can now pick z-scores based on our desired accuracy and graph these equations. This example uses

\[ z_b(N(p), t(p)) = -0.253 \]

and

\[ z_b(N(p), t(p)) = 3.090 \]

We can clearly see that the optimal probability is approximately 0.384 for \( k=2 \) and \( d=5 \) and approximately 0.517 for \( k=3 \) and \( d=5 \).

Results Continued

Using the z-scores specified in the previous example, we calculated the following table where the sequence for the data is \([p, t(p), N(p), p_b, p_g] \).
Interesting Results

However, using different z-scores, we calculated a new table. Note that the optimal probabilities are only slightly different that those in the table above. Seeing this, we decided to try a different approach to optimization.

For example, in the first column starting with the second row, the values of $p$ change from 0.650 to 0.648, from 0.716 to 0.713 and from 0.761 to 0.758.

<table>
<thead>
<tr>
<th>k=2</th>
<th>d=2</th>
<th>d=3</th>
<th>d=4</th>
<th>d=5</th>
</tr>
</thead>
<tbody>
<tr>
<td>[540, 590, 619, 691, 469]</td>
<td>[468, 643, 682, 842, 498]</td>
<td>[149, 810, 256, 608, 494]</td>
<td>[394, 569, 363, 594, 488]</td>
<td></td>
</tr>
<tr>
<td>k=3</td>
<td>[948, 650, 131, 630, 426]</td>
<td>[590, 621, 234, 806, 500]</td>
<td>[549, 801, 351, 597, 501]</td>
<td>[517, 585, 487, 575, 501]</td>
</tr>
<tr>
<td>k=4</td>
<td>[719, 615, 188, 588, 478]</td>
<td>[596, 668, 319, 578, 494]</td>
<td>[631, 565, 457, 571, 499]</td>
<td>[524, 576, 649, 592, 502]</td>
</tr>
<tr>
<td>k=5</td>
<td>[758, 565, 245, 566, 462]</td>
<td>[718, 578, 408, 557, 437]</td>
<td>[888, 572, 568, 554, 406]</td>
<td>[965, 565, 813, 551, 501]</td>
</tr>
</tbody>
</table>

Conclusions

In order to determine why this method worked, we went back to some of our original equations. By substituting our equation for $t(p)$ back into the previous equations we get

\[
\frac{z_a(N,t)\sqrt{p_0(1-p_0)}}{p_0z_a(N,t)\sqrt{p_0(1-p_0)} - p_0z_a(N,t)\sqrt{p_0(1-p_0)}} - p_0
\]

and

\[
\frac{z_a(N,t)\sqrt{p_0(1-p_0)}}{p_0z_a(N,t)\sqrt{p_0(1-p_0)} - p_0z_a(N,t)\sqrt{p_0(1-p_0)}} - p_0
\]
Conclusions, Continued

Simplifying either of those equations gives

\[
\frac{z_s(N,t)\sqrt{p_b(1-p_b)} - z_g(N,t)\sqrt{p_g(1-p_g)}}{p_b - p_g}
\]

This starts to explain why maximizing the difference between \(p_b\) and \(p_g\) is effective. It is clear that when the denominator is maximized the overall fraction will shrink.

However, this is not a complete answer to the question because \(p_b\) and \(p_g\) are still variables in the numerator. We have tried to rearrange the equations algebraically and to examine their derivatives, but have found that there does not seem to be any way to isolate the variables.

Applications, Continued

This is a picture of Syer Green Florescence. The figure illustrates the difference between pools with double stranded DNA (a positive test in which two of the single strands have bonded together) and pools without.

The curves on the top show pools that contain a positive pair, and those on the bottom do not.