A Genetic Algorithm for Modeling Germinal Center Dynamics

Rick Wolfley

CS423

Abstract

The human immune system is a complex system of interest to immunologists and computer scientists alike. Germinal centers play a vital role in the immune system's adaptability. This paper explores the relationship between genetic algorithms and germinal centers. A specific implementation of a genetic algorithm for modeling germinal center dynamics is proposed. Performance of the genetic algorithm is then tested and shown to be effective.

1 Introduction

The immune system's goal is to defend the body against infection from dangers like viruses and germs. These invasive bodies are characterized by their *antigens*, which are molecules that directly interact with the immune system. Antigens have surfaces called *epitopes* which bind to analogous molecules produced by the immune system called *antibodies*. Antibodies are produced by lymphocytic cells called *B cells*. *Antibody affinity* is the strength of the bond between an antibody and an antigen's epitope. This affinity is determined by similarity between the genetic makeup of the antibody and epitope. The more closely the genetic makeup of the antibody is to the antigen's epitope, the greater the antibody affinity and overall suppression of the infection. The immune system increases antibody affinity through strategic genetic alteration of antibodies, called *affinity maturation*. This process takes places in special zones called *germinal centers* which are found in organs such as the spleen and lymph nodes.

We can attempt to model the above scenario using a *genetic algorithm* (GA). Inspired by the process of natural selection, a genetic algorithm tests and evolves various solutions to a problem in an attempt to "breed" an optimal solution. In modeling the problem of affinity maturation, we can start with a set of antibodies (think of these as the antibodies your body would have if it were not fighting any particular infection) and a set of antigens of varying genetic makeups that our immune system needs to fight against. We can think of different B cells in germinal centers as having different rules for gene alteration of antibodies. Rules which produce antibodies with a high affinity to the set of antigen epitopes have a greater *fitness* than rules which produce antibodies with low affinity. The genetic algorithm steps through many *generations* of evolution, measuring the fitness of the various rules the B cells use for affinity maturation. B cells whose rules meet a certain fitness standard produce *children*, slightly modified versions of the parent, who are tested in the next generation. B cells whose rules do not meet the fitness standard die out.

The question we want to answer is: using a genetic algorithm, can we define a broad rule set for affinity maturation, a heuristic for measuring a rule's fitness, and a method of breeding "fit" rules that can produce antibodies with high affinity in relation to a set of epitopes. A specific implementation of such a genetic algorithm is described in the subsequent section, followed by a description of the GA's performance. Testing reveals that the genetic algorithm is well suited for breeding affinity maturation rules for suppression of randomized epitope populations given a randomized set of initial antibodies.

2 Methods and Results

2.1 Implementation of Epitopes, Antibodies, and B Cells

Epitopes and antibodies are represented in the genetic algorithm as strings of eight amino acids. There are twenty possible amino acids, represented using the digits 1 through 20. We can populate the epitopes and antibodies initially with a random configuration of amino acids. B cells are represented as structures containing a starting set of antibodies, a set of numbers for measuring changes in affinity between subsequent antibody generations ($\Delta_{affinity}$), a number to track fitness, and a rule for modifying the amino acids during affinity maturation.

2.2 Measuring Antibody Affinity

To measure the affinity between an epitope and an antibody, compare the amino acids at corresponding indexes in the string of amino acids making up each molecule's genotype. Using the bitwise comparison `a XOR NOT(e)`, where `a` is the antibody's amino acid and `e` is the epitope's amino acid, we generate a string of all 1's (the highest possible affinity) if the amino acids are the same. This can equivalently be implemented as `NOT(a) XOR e`. We add the affinity values for each amino acid index to get an affinity score for the antibody. When recording affinity changes in $\Delta_{affinity}$ between subsequent trials, we record a 1 if the affinity increased, 2 if it decreased, 3 if it stayed the same at the maximum (the amino acids match), or 4 if it stayed the same at a value other than the maximum.

2.3 Affinity Maturation Rules

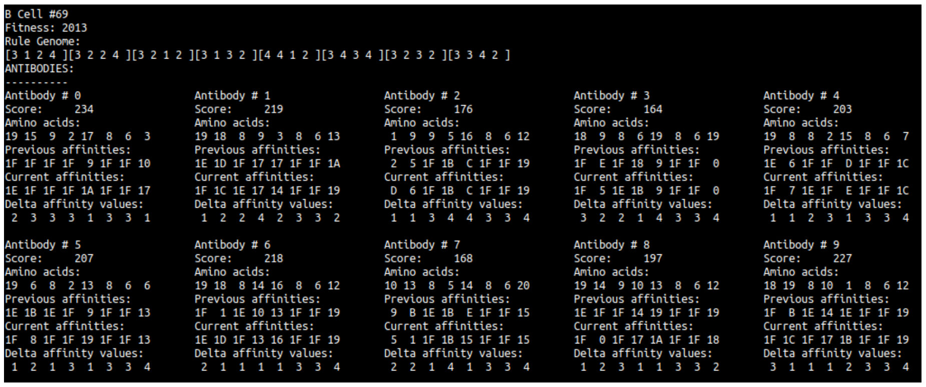
During affinity maturation, each B cell uses a rule to modify the amino acids of the antibodies before retesting their affinity with the antigens. There are many different rule sets we could implement for doing this. We will limit ourselves to four operations: (1) increment the amino acid, (2) decrement the amino acid, (3) keep the amino acid the same, or (4) randomize it. When incrementing at a maximum value or decrementing at a minimum value, the amino acid number is wrapped back around to the opposite maximum or minimum. Combining these operations with the possible values given by $\Delta_{affinity}$, we can implement a "genome" for a rule set that modifies amino acids during affinity maturation. For each of the eight amino acid indexes in an antibody, we define the operation that will be performed in the event of the four possible values in $\Delta_{affinity}$. For example, given the amino acid in index 1, we may decrement when $\Delta_{affinity}$ is 1 (increases), increment when $\Delta_{affinity}$ is 2 (decreases), stay the same when $\Delta_{affinity}$ is 3 (stays the same at the maximum), and randomize when $\Delta_{affinity}$ is 4 (stays the same at a value other than the maximum). For each amino acid index, we have four possible $\Delta_{affinity}$ values, each with four possible rules. This gives us 256 possible rules for each amino acid position, for a total of 256^8 (about 18,446,744,000,000,000,000) possible rule genomes.

2.4 Fitness Testing

To test the fitness of a rule, we initialize a set of B cells, each with a set of random antibodies. We also initialize a set of global epitopes that the antibodies need to suppress. We then measure the antibody affinity at each amino acid index for an antibody-epitope pair and add them to get an affinity score for an antibody. In doing this for each pair, we have two options: add the affinity scores for each pair together to make a fitness score for the B cell and its associated affinity maturation rule, or save only the highest affinity score for a specific pair and use that to gauge the fitness of the B cell. The intuition behind the first option is that it might create more versatile antibodies. The second seems like it might be way of forming antibodies specialized to target a specific antigen. The first implementation more closely characterizes what we see in germinal center dynamics, as the B cells are "blind" to the epitopes and have no way of knowing the genetic makeup of specific epitopes, only the overall effectiveness of the ability of set antibodies to suppress the set of epitopes as a whole. After the antibodies have been tested against the antigens, we apply the rule each B cell has for modifying the antibodies. This process of testing antibody affinity and modifying the antibodies according to their change in affinity is repeated over a series of trials. We want to walk through enough trials that the antibodies evolve under the B cells' rule sufficient to measure the rule's fitness. After completing a trial cycle, we add the total antibody scores to compute the B cell's overall fitness score.

2.5 Child Generation

To generate children for the next generation, we take the rules which exhibit the best fitness scores for a given generation of B cells, split the rules at a randomized amino acid index, and combine the two complementary parts to



```
B Cell #69
Fitness: 2013
Rule Genome:
[3 1 2 4 ][3 2 2 4 ][3 2 1 2 ][3 1 3 2 ][4 4 1 2 ][3 4 3 4 ][3 2 3 2 ][3 3 4 2 ]
ANTIBODIES:
----------
Antibody # 0                Antibody # 1                Antibody # 2                Antibody # 3                Antibody # 4
Score:     234              Score:     219              Score:     176              Score:     164              Score:     203
Amino acids:                Amino acids:                Amino acids:                Amino acids:                Amino acids:
19 15  9  2 17  8  6  3     19 18  8  9  3  8  6 13      1  9  9  5 16  8  6 12     18  9  8  6 19  8  6 19     19  8  8  2 15  8  6  7
Previous affinities:        Previous affinities:        Previous affinities:        Previous affinities:        Previous affinities:
1F 1F 1F 1F  9 1F 1F 10     1E 1D 1F 17 17 1F 1F 1A      2  5 1F 1B  C 1F 1F 19     1F  E 1F 18  9 1F 1F  0     1E  6 1F 1F  D 1F 1F 1C
Current affinities:         Current affinities:         Current affinities:         Current affinities:         Current affinities:
1E 1F 1F 1F 1A 1F 1F 17     1F 1C 1E 17 14 1F 1F 19      D  6 1F 1B  C 1F 1F 19     1F  5 1E 1B  9 1F 1F  0     1F  7 1E 1F  E 1F 1F 1C
Delta affinity values:      Delta affinity values:      Delta affinity values:      Delta affinity values:      Delta affinity values:
 2  3  3  3  1  3  3  1      1  2  2  4  2  3  3  2      1  1  3  4  4  3  3  4      3  2  2  1  4  3  3  4      1  1  2  3  1  3  3  4

Antibody # 5                Antibody # 6                Antibody # 7                Antibody # 8                Antibody # 9
Score:     207              Score:     218              Score:     168              Score:     197              Score:     227
Amino acids:                Amino acids:                Amino acids:                Amino acids:                Amino acids:
19  6  8  2 13  8  6  6     19 18  8 14 16  8  6 12     10 13  8  5 14  8  6 20     19 14  9 10 13  8  6 12     18 19  8 10  1  8  6 12
Previous affinities:        Previous affinities:        Previous affinities:        Previous affinities:        Previous affinities:
1E 1B 1E 1F  9 1F 1F 13     1F  1 1E 10 13 1F 1F 19      9  B 1E 1B  E 1F 1F 15     1E 1F 1F 14 19 1F 1F 19     1F  B 1E 14 1E 1F 1F 19
Current affinities:         Current affinities:         Current affinities:         Current affinities:         Current affinities:
1F  8 1F 1F 19 1F 1F 13     1E 1D 1F 13 16 1F 1F 19      5  1 1F 1B 15 1F 1F 15     1F  0 1F 17 1A 1F 1F 18     1F 1C 1F 17 1B 1F 1F 19
Delta affinity values:      Delta affinity values:      Delta affinity values:      Delta affinity values:      Delta affinity values:
 1  2  1  3  1  3  3  4      2  1  1  1  1  3  3  4      2  2  1  4  1  3  3  4      1  2  3  1  1  3  3  2      3  1  1  1  2  3  3  4
```

Figure 1: An example B cell configuration, taken from a simulation printout. The B cell's fitness is given at the top. Below the fitness score is the rule genome used for modifying antibodies, split up in brackets corresponding to each amino acid index. The numbers in the bracket are the operations to be performed for the $\Delta_{affinity}$ values at the corresponding index. For example, in the first amino acid index, operation 3 is to be performed for $\Delta_{affinity}$ 1, operation 1 for $\Delta_{affinity}$ 2, and so on. Below the rule genome are the antibodies. They each have a fitness score as an overall measure of their affinity, a list of their amino acids, a list of affinity values for the previous set of amino acids (given in hexadecimal), a list of current affinity values (also in hexadecimal), and a list of the $\Delta_{affinity}$ values.

form a child rule. We breed random parents within the set of fit rules to create enough children to test in the subsequent generation of trials.

2.6 Simulation Parameters

A number of the parameters used in the model can be varied. These include the number of B cells in each generation, the number of antibodies in each B cell, the number of global epitopes, the number of fit B cells to select for breeding a subsequent generation, the number of B cell generations to evolve, and the number fitness evolution trials to perform for each generation. A variety of parameter values were used during testing of the GA and were generally found to be scalable, meaning that given a certain ratio of B cells, antibodies, generations, and trials per generation, the results stayed consistent. A few general notes on parameter values are:

- The number of B cells needs to be sufficiently large for significant genetic variation to occur during breeding of the fittest B cells for a subsequent generation.
- Likewise, the number of fit B cells selected for breeding needs to be large enough for sufficient genetic variation. For simplicity, the number of fit B

cells selected between generations remained constant.

- The number of epitopes also remained constant between generations.
- The number of generations needs to be large enough that it gives the initial rule genome space sufficient time to mix.
- The number of trials is analogous to the lifespan of a B cell. A large number of trials is needed to see whether or not the affinity maturation rule used by a B cell is effective.

2.7 Test Results

All tests were performed using a global set of epitopes that each B cell created antibodies to fight against. In figure 2 we see how well the genetic algorithm was able to evolve affinity maturation rules to create antibodies which suppress randomized sets of epitopes.

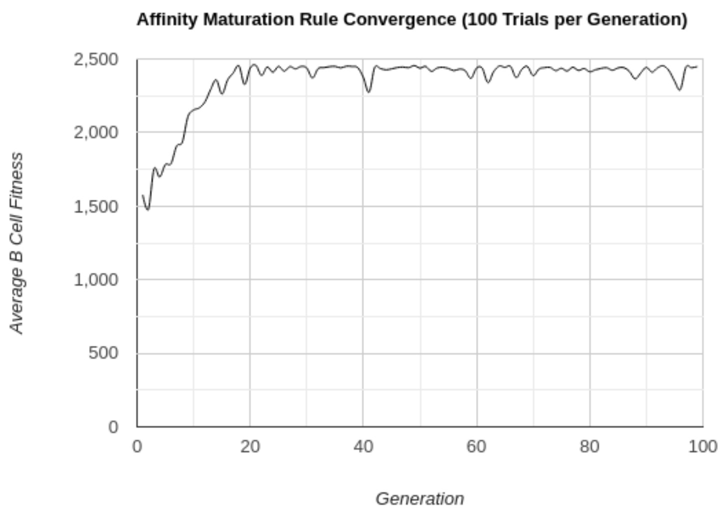


Figure 2: A visual of the GA's quickness in converging an an optimal solution. The Y-axis measures the average fitness B cells in a generation, i.e. how well its affinity maturation rule produces antibodies that suppress a random set of epitopes. This fitness is shown in correspondence to the number of B cell generations the afffinity maturation rule has evolved over along the X-axis. The maximum possible fitness score is 2480 (graphed using https://www.rapidtables.com/tools/line-graph.html).

[3 4 3 2]	[3 1 3 1]	[3 4 3 2]	[3 3 3 1]	[3 3 3 4]	[3 4 3 4]	[3 2 3 2]	[3 3 3 2]
[3 4 3 2]	[3 4 3 2]	[3 1 2 1]	[3 2 3 2]	[3 4 3 1]	[2 4 4 4]	[3 2 1 4]	[3 1 3 1]
[3 2 1 4]	[3 2 3 2]	[3 4 3 2]	[3 1 3 1]	[3 1 3 4]	[3 2 1 2]	[3 3 3 2]	[3 1 2 4]
[3 4 3 2]	[3 4 3 2]	[3 1 3 1]	[3 1 2 1]	[3 3 3 2]	[3 1 3 1]	[3 1 3 1]	[3 2 3 2]
[3 4 3 1]	[3 2 3 2]	[2 1 1 2]	[3 2 3 2]	[3 1 2 2]	[3 4 3 2]	[3 1 3 1]	[3 4 3 1]
[3 4 3 2]	[3 3 3 2]	[3 4 3 2]	[3 1 2 4]	[3 1 2 4]	[3 3 1 2]	[3 3 3 1]	[3 2 1 2]
[3 4 3 1]	[3 4 1 2]	[2 1 3 1]	[3 3 3 4]	[3 1 3 1]	[3 2 3 4]	[3 2 3 4]	[3 3 3 1]
[3 4 3 2]	[3 3 3 1]	[3 2 1 2]	[3 4 3 4]	[3 4 3 1]	[3 3 3 1]	[3 3 2 1]	[3 1 2 1]
[3 1 2 1]	[3 2 3 2]	[3 1 3 1]	[3 4 3 2]	[3 2 3 2]	[3 4 3 1]	[3 2 3 2]	[3 4 3 1]
[3 3 3 1]	[3 1 2 1]	[3 2 3 4]	[3 3 3 2]	[3 2 3 2]	[3 1 3 4]	[3 1 3 1]	[3 3 3 4]
[3 4 3 2]	[3 3 3 2]	[3 4 3 1]	[3 4 3 4]	[3 4 3 1]	[3 1 3 1]	[3 1 2 4]	[3 3 3 2]
[3 3 3 1]	[3 4 3 2]	[3 1 3 1]	[3 3 2 1]	[3 4 3 2]	[3 4 3 4]	[3 1 3 1]	[3 3 3 1]
[3 4 3 1]	[3 3 3 1]	[3 3 3 4]	[3 1 3 1]	[3 2 1 4]	[3 1 2 4]	[3 2 3 4]	[3 3 3 2]
[3 3 3 4]	[3 2 3 4]	[3 2 3 4]	[3 3 3 1]	[3 2 3 4]	[2 4 4 1]	[3 4 3 1]	[3 3 3 4]
[3 3 3 4]	[3 4 1 4]	[3 2 3 2]	[3 2 3 4]	[3 4 3 2]	[3 1 4 1]	[3 2 3 4]	[3 3 3 2]
[3 2 1 2]	[2 4 4 4]	[3 4 3 2]	[3 2 3 2]	[3 2 3 4]	[3 3 3 4]	[3 3 3 1]	[3 3 2 1]
[3 3 2 1]	[3 1 3 1]	[3 4 3 2]	[3 1 3 4]	[3 4 3 1]	[3 2 1 4]	[3 1 3 1]	[3 3 3 2]
[3 4 3 1]	[3 3 3 2]	[3 1 3 1]	[3 3 3 2]	[3 2 3 4]	[3 3 3 2]	[3 2 3 4]	[3 1 2 1]
[3 3 3 1]	[3 3 1 2]	[3 4 3 2]	[3 3 3 1]	[3 2 3 2]	[3 4 3 2]	[3 2 3 4]	[3 1 3 1]
[3 2 3 2]	[3 4 3 2]	[3 3 3 4]	[3 4 3 2]	[3 2 1 2]	[3 3 3 1]	[3 2 3 4]	[3 2 3 4]

Table 1: Genomes of the fittest rules from 25 simulation runs. Each simulation was run using 100 B cells over 100 100 generations, each running 100 trial steps.

Table 1 gives the genome of the fittest affinity maturation rule for 50 simulation runs. Each simulation run covered 100 generations, each with 100 trials, using 100 B cells.

3 Discussion and Conclusion

Looking at figure 2, we see that the GA was able to evolve B cells with near optimal affinity maturation rules in approximately 20 generations. With an initial randomized set of antibodies, the B cells' fitness was about 60% of the maximum possible fitness. There are a few observations we can make about the table of fittest rule genomes. For every amino acid index, if $\Delta_{affinity}$ increased, we keep the amino acid the same. If $\Delta_{affinity}$ stayed the same at a value other than the maximum, we always change it. A further exploration of the behavior of these rules is the topic for another paper.

These simulation results offer insight into immunology in general. When exposed to a novel infectious agent such as COVID-19, our bodies will attempt to develop antibodies to fight it as quickly as possible. The genetic makeup of existing antibodies in relation to the novelty of the infectious epitopes, the number of germinal centers and B cells able to produce and modify antibodies, and the speed at which affinity maturation occurs are the key factors in determining how the immune system responds to an infection. Given a small set of initial antibodies, affinity maturation over many generations can only produce antibodies with limited genetic variance. A healthy mutation percentage and high genetic variance in antibodies is essential for having a versatile immune system. Vaccines are a way of artificially inserting effective antibodies into the immune system, which can then be modified over time in response to slight variations in the genetic makeup of of whatever infectious agent they were formed to fight against.

Complex systems such as germinal cell dynamics and the immune system in general exist all around us. Understanding such systems can be extremely useful, but also extremely difficult. As has been shown in this paper, creative computational techniques can be used to model complex systems. Using a genetic algorithm we were able to "breed" increasingly optimal solutions to a problem, which is analogous to what happens in germinal centers when creating antibodies to fight an infection.

In practice, there are many other factors involved germinal center dynamics which our model omits or simplifies. To name a few:

- Epitopes evolve gradually over time in response to antibody evolution. In our model, epitopes stay the same until they are randomized again for another generation.
- Real B cells share a significant amount of antibody types, whereas in our model they are randomized between B cells to more strongly test affinity maturation rules.
- We use a simplified scheme for creating affinity maturation rules. In an actual immune system it is a much more complex paradigm.

These simplifications (and many more) aside, our genetic algorithm effectively models germinal cell dynamics and provides insight into the intricacies of complex systems in general.

References

[1] Pae, Juhee et al. *Imaging the different timescales of germinal center selection.* Immunological reviews vol. 306,1 (2022)

[2] Walker, Sara & Cisneros, Luis & Davies, Paul. *Evolutionary Transitions and Top-Down Causation.* (2012)

[3] M. Mitchell. *Complexity: A Guided Tour.* Oxford University Press. (2011)

[4] https://en.wikipedia.org/wiki/Immune_system

[5] Genetic algorithm source code: https://github.com/rick6w/GCGA