

The Genetic Algorithm in Economics

Jonathan Lucas Reddinger
Department of Agricultural Economics and Economics
Montana State University, Bozeman, MT, 59717
jlr@lucasreddinger.com

4 May 2007

Abstract

In computer science, the genetic algorithm (GA) is known as a very useful optimization technique. The GA uses the theory of natural selection to weed out the less desirable data and arrive at the optimal solution. In economics, the GA can be used as a model of social learning, which is how individuals get information from others. The GA does this analogously to natural selection, with economically beneficial information in place of preferable traits for selection. This paper explores the use of the GA to model the spread of new agricultural technologies in Ghana during their green revolution. When technology adoption is thought of as a necessary trait for survival in a competitive environment, technology diffusion is analogous to traditional evolutionary theory. The GA can be used to model such learning environments, as it simply imitates natural selection. Such a model may be extremely useful for modeling many economics problems, including fertilizer and high yield variety seed adoption in developing countries.

1 Introduction

This paper aims to bridge gaps between three fields of study: technological diffusion, social learning, and genetic algorithms. These topics have been written on at length; in addition, technological diffusion has been related to social learning, and social learning has been related to the genetic algorithm. However, the literature at present still lacks an empirical work relating the GA to technological diffusion.

There have been numerous works establishing technological diffusion as a case of social learning—both theoretical papers and empirical results. Perhaps the most widely known paper is “Diffusion as a Learning Process: Evidence from HYV Cotton” by Besley and Case[2]. This paper lays out some of the necessary groundwork for my study, as it used an empirical study of HYV technology diffusion as modeled by a social learning process. Another widely referenced paper

is “Learning by Doing and Learning from Others: Human Capital and Technical Change in Agriculture” by Foster and Rosenzweig[4]. In this paper, the authors look at the green revolution in India, and analyze the learning process that took place between agents. One of the pioneering papers in the field of learning was Banerjee’s “A Simple Model of Herd Behavior,”[1] where he sets forth the idea that decision-makers look at others who have previously made similar decisions.

This paper begins with a detailed explanation of how the genetic algorithm works. I then discuss how the GA may be applied to economics problems. I subsequently discuss Ghana’s green revolution and other agent-based modeling approaches which I have been working on.

2 The Genetic Algorithm

Genetic algorithms have proved themselves useful in many areas of study. These include any complicated optimization problem imaginable, from traditional problems such as the job shop scheduling and traveling salesman problems, to newer real world problems like airline scheduling of arrivals and departures. They are well suited for use in many computational fields, including Mathematics, Engineering, Computer Science, Biology, Finance, Physics, and Economics[7].

The GA was initially developed by Holland[6] and Goldberg[5], computer scientists, in the 1970’s and 1980’s. It was initially intended to be a discrete optimization algorithm. In computer science (CS), everything is discrete, which means that calculus can be done in two different ways, symbolically or approximately. Symbolic calculus will simply attempt to replace expressions such as $y(x) = x^2$ with $y'(x) = 2x$ when asked to take a derivative. It has limitations simply because every possible expression cannot be coded. However, approximate calculus isn’t the best choice, simply because it can only estimate a solution. Because CS lacks true calculus optimization techniques, the GA was developed to solve these problems. Another motivation for the development comes from ambiguity and intractability in optimal solutions provided by calculus. First, there may be a large number of maxima. Second, the function may not be easily twice differentiable. Lastly, a twice differentiable function simply may not be available. The most interesting cases show both many maxima and a lack of a twice differentiable function.

At this point, we can no longer consider calculus a viable technique for optimization and we must look towards computational techniques. There are a wide variety here to choose from—grid methods, Markov chains/processes, amoeba algorithms, and genetic algorithms. A grid method simply divides up the search space and hopes to find the best number. For example, given a one dimensional search space of $[0, 100]$, the algorithm will respectively evaluate 50, 25, 75, 12.5, 37.5, 62.5, ..., until the given number of allowed computations is reached. The highest value found is declared the optimal solution. This presents

a problem, however, because maxima could exist between the evaluations. In addition, if the search space is large enough, the number of computations is very large. In response, there is the Markov chain. This is a stochastic algorithm, using randomization as the primary tool for finding the answer. This will hopefully help find maxima between the given evaluations; however it is also not great at optimization in a large multi-modal search space for the same reasons. Additionally, the high use of randomization also produces computational constraints. Next, we have the amoeba (or hill-climber). It will begin at a random point, and it will “test the waters” in both directions. It will then move in the better direction (the one with a higher evaluation/fitness), and it will continue the process until there is no move which is better than the current position. Essentially, the amoeba climbs an n -dimensional hill. However, in multi-modal search spaces, this is clearly not the best technique. The amoeba will climb one hill, neglecting the other maxima in the search space. The GA presents a great alternative to these techniques. I will begin by offering an analogy of the genetic algorithm, and then I will go on to describe the process from the computer science aspect.

2.1 An Analogy and Intuition

Think about a sine wave as a bunch of hills. There are thirty hamsters, some sitting in the valleys, and some near the tops of the hills. If we move all of the food to the tops of the hills, the hamsters will eventually migrate there (those that don't will die). Now, create a function for food placement. We already know that $y = \sin(x)$. Let us say that $food = ky \ni k > 0$, so that there is more food at the top of the hills than the bottoms. Now our hamsters are just an x -value, and their probability of natural selection is proportional to food.

Similarly, here's what a genetic algorithm does: randomly places a population of genetic organisms on the search space; evaluates their likelihood (fitness) of selection; selects mates proportionally to their fitness; performs crossover on their genes, producing two offspring for every pair of parents; randomly mutates the children's genes; the children become the new parents, and the cycle repeats itself.

The GA works in two ways: integration of known information (crossover) and experimentation (mutation). The GA is designed around the concept of evolution and natural selection. Given a function, the organisms consist of the independent variables. These variables adapt to maximize the given function. This function is called the fitness function. This function $f(x_1, x_2, \dots, x_n)$ is the objective function to maximize (or minimize, although this point is trivial with the use of a minus sign). Let us now go through the GA as seen through the eyes of a computer programmer.

2.2 Functions

2.2.1 Initialization

This function randomly places a specified number of genetic organisms on the search space. This involves generating a random number for each dimension of each organism placed. Thus, we have some number (commonly around thirty) of organisms, each containing one random number for each dimension. Each organism is then an n -dimensional organism, initialized randomly.

$$x_{i,j} = \text{random_number}() \quad \ni \begin{cases} i = 1, 2, \dots, \text{pop_size} \\ j = 1, 2, \dots, n \end{cases}$$

2.2.2 Evaluation

This function simply involves determining the fitness of every organism i by applying f to each organism's coordinates.

$$\text{eval}(i) = f(x_{i,1}, x_{i,2}, \dots, x_{i,n}) \quad \ni i = 1, 2, \dots, \text{pop_size}$$

2.2.3 Selection

There are many methods of selection; the most important aspect being that selection should always be fitness-proportionate (we want the best solutions, not the worst!) I will focus on random roulette selection, likely the most popular selection routine. First, we must transform every agent's fitness into a non-negative fitness score. We do this by finding the individual with the lowest fitness, and subtracting it from every agent's fitness score.

$$\text{eval}'(i) = \text{eval}(i) - \min_j f(j)$$

Then we sum the population's fitness. Now, we can take each organism's fitness score and divide it by the total population fitness, arriving at the individual agent's contribution. We then get a probability that an i th organism will be selected.

$$\text{prob}_{\text{selection}}(i) = \frac{f'(x(i))}{\sum_{1 < k < \text{pop_size}} f'(x(k))}$$

We are essentially given a dart board, and by picking a random number, we toss a dart and choose an agent for selection. Thus, the probability of selection is higher for those individuals with higher f evaluation, and lower for those with a lower f evaluation.

2.2.4 Crossover

Realize that each organism's coordinates are stored in binary of a given string length. Given two one-dimensional organisms 00101101 and 10001011, a random crossover point is selected. If the bit length is 8, then the random integer is constrained to $[0, 7]$. If 0 is chosen, then the children are identical to the

parents. Otherwise, any number $[1, 7]$ will give a true crossover. For example, if 2 is selected, we exchange the bits around that point (00|101101 and 10|001011 produce 00001011 and 10101101). The idea here is that we are swapping some alleles and the resultant bit-strings are the children. This can be done with multiple cut points, but using one cut point gives us a simple yet effective algorithm.

2.2.5 Mutation

Our mutation function randomly flips alleles (bits). Thus, for every bit of every organism, a random number is generated from $[0, 1]$. If this number is less than the probability of mutation ($prob_{mutation}$ is typically set to around 0.05), then the code flips the bit in question. This provides us with some randomness and experimentation in the genes.

2.2.6 Reiteration

Now consider this set of children as the new parents and repeat the process (each repetition is called a generation, and there are typically anywhere from ten to thousands of generations.) They will eventually converge to the maxima of the fitness function, if there are any.

2.3 Further Modifications

This is called a canonical simple genetic algorithm. From this view, a GA is simply a special case of a Markov Chain (with the addition of the crossover operator). GAs are quickly replacing older optimization techniques due to their efficiency (time to converge) and robustness (ability to scale and adapt). GAs have advanced dramatically from a mathematical standpoint since their conception.

This process can be modified extensively. One of the most significant and helpful modifications is that of the mutation rate, $prob_{mutation}$. Just as the organisms within the genetic algorithm will converge to an optimal solution of f , we can incorporate a mutation dimension into each organism. Then, this mutation rate will also evolve while the algorithm is run and the mutation rate will approach the ideal rate. This helps significantly with stability and robustness. Typically, the mutation rate begins high and ends low. This can be compared to the technique of simulated annealing. Change begins high and ends slow, just like Newton's Law of Cooling. When this change in mutation rate is observed, we typically see a high rate at the beginning (exploration), followed by a lower rate near the end (convergence). A genetic algorithm that changes the way in which it operates is called a meta-GA, a "deep memory" GA, or sometimes a GGA (genetic-genetic algorithm).

3 GA Applications in Economics

Within the field of Economics, GA applications are varied. For example, one can use a GA to optimize an economic strategy (e.g., finding the optimal quantity for a monopoly to produce). One can also apply the GA to real world market data to simulate portfolio allocation between different funds. From a theoretical standpoint, these techniques are simply a demonstration of what a GA represents in economics—learning. GAs show how economic agents with bounded rationality share information and attempt to arrive at the best possible outcome. There are many papers that develop such a model of bounded rationality and demonstrate economic learning as a parallel of the learning of a GA[8].

This use of the evolutionary process by the GA can be used to model agents in a market setting. Because efficiency is the criterion of the market, the market performs a sort of natural selection on its participants. The evolutionary process as presented above poses a few primary components: selection based on agent fitness, crossover, and mutation. First, let us consider selection. Selection based on agent fitness can be equated to market selection based on firms' accounting profits. The market will continually prefer those firms with higher accounting profits; those with the lowest profits will exit the market. Thus, the natural selection that occurs in nature is very similar to what occurs in the market place. Next, let's look at crossover. This function simply performs a weighted average on the inputted agents and outputs offspring agents. Intuitively, this function basically shares multiple agents' information with new agents. To that end, crossover is analogous to firm entrants modeling their business on some combination of existing business models. Of course, there needs to be some innovation in this model; this is where mutation is accounted for. Mutation represents a firm's individual modifications to the information it gains. This is the process of experimentation and innovation that exists both in natural evolution and in the market place. These three components create a simulated market context: selection, crossover, and mutation.

Returning to our original GA example with $pop_size = 30$, all we need to do to model a market is let the fitness function be firm profits. For example, one commonly discussed use of the GA is to model homogeneous firms that do not know their own optimal quantity of production. In this context, quantity becomes our independent x -variable, and our fitness function is profit measured as a function of quantity. The functional form of profit must include some maximum value of profit that can be obtained so that an optimization is legitimate. The GA will then begin with many firms trying many quantities. The firms that produce closer to the optimal quantity will have more profits, resulting in their business model being replicated at an increased rate. Experimentation of other production quantities will help firms find the optimum. Eventually, all firms will be operating at this optimal point of production, because all others would be making negative economic profits. It is in this fashion that GAs are

used in an economic context. What they model depends solely on their fitness function. Thanks to similarities between the market and natural evolution, the GA has been helpful in modeling many social phenomena.

Now let us return to the meta-GA as presented previously in this paper. The most common meta-GA alters its mutation rate as the evolutionary process progresses. Typically, the mutation rate will begin very high, then lower drastically, and eventually become negligible. This is relatable to economic learning as well. Reichmann explains that “meta learning endogenizes the individual propensity to experiment” [8]. Thus, it becomes another element of the experiment and the agents are allowed to determine their own propensity to experiment. The three stages of this learning are exploration, concentration, and convergence. At the beginning, the agents have a high propensity to experiment, with a high mutation rate, high variances in strategies, and widely scattered across the search space. Later, the agents will begin to lower the rate and concentrate on one region. Finally, the agents will not significantly mutate any longer and will converge to an optimal bundle.

Such is the quintessential genetic algorithm—it uses the initialization, evaluation, selection, crossover, and mutation operators. It can optimize anything that can be programmed into the fitness function using n dimensions. Most notably, many of its characteristics are useful in explaining social phenomena. Let us now look at some such cases.

4 Ghana’s Green Revolution

Many developing countries have recently experienced a green revolution, which is simply the use of new agricultural technologies, including high yield variety (HYV) seeds, newly developed fertilizers and chemicals, in addition to new techniques and capital. As a result, the economies of countries like Ghana and India have changed dramatically in the past two decades. Naturally, economists are attempting to model what takes place in these countries. Of course, there is interest in capital and labor markets that affect these transitions, but there is also great interest in how farmers learn about these new technologies that compose a green revolution. After all, if such information can be spread more effectively or efficiently, then these green revolutions would propagate at a much greater pace.

4.1 Theory

To model these green revolutions, economists have equated social learning with technological diffusion [2]. Economists then apply more general social learning models to the case of technological diffusion. In addition, social network models are of particular interest, and they utilize an agent-based approach to analysis of learning. These models look closely at how neighbors (agents) interact with

each other, and what the resulting consequences are. In the case of Ghana[3], physical distance between neighbors inhibits communication, as one would expect in a developing country. There are also agents that act as communication hubs, which transmit information to numerous other agents. The agent-based approach is the most valuable, as it takes individual interaction into account. To this end, the Genetic Algorithm (GA) is valuable.

Most prominently, Conley and Udry studied the green revolution in person and wrote “Social Learning through Networks: The Adoption of Net Agricultural Technologies in Ghana.” In this paper, the authors present their data as a network of farmers who share information amongst themselves, while occasionally experimenting. They mention the rates at which information is shared among farmers, which is analogous to the crossover rate in the GA. The GA seems to be very suitable to modeling this diffusion; in fact, the GA model approximates many points of the model presented by Conley and Udry.

This allows us to use the GA as a model of technological diffusion in Ghana. The crossover function represents the information that one farmer gets from other farmers, and the mutation function represents a farmer

4.2 An Empirical Study

In this project, there are three components that are compared. These include the actual data on the adoption of new agricultural technologies, the GA’s prediction of the adoption of these technologies, and the OLS prediction as well. The hypothesis under consideration is: The GA is a more accurate model of agricultural technology diffusion than OLS. To this end, we must look at the change in adoption over time as predicted by OLS, and determine if it is more accurate (as compared to the true data) than what the GA predicts for change in adoption over time.

I used the data gathered by Conley and Udry, who performed a study on Ghana farmers over the course of 21 months from 1996 to 1998. The survey monitored many aspects of production and the characteristics of 450 farmers in four villages. Being a panel dataset, there is information about the inputs and outputs of the farmers over the course of 21 months. Thus, the data provides us with the number of times new agricultural technologies are used as an input in a given period, and we can determine the change in this utilization (the adoption rate) over time.

Unfortunately, I have experienced significant problems with the analysis of this dataset. Because of the nature of this panel dataset, it is very hard to determine precisely what each variable means, and how to measure certain effects. For example, I was able to generate unique IDs for each of the farmers; however, a farmer may use a new technology in time period 1 but not in time period 2. The result is not that he or she purged knowledge of the new technology—the

round	2	3	4	5	6	7	8	9	10	11	12	13	14	15
seed	114	112	437	305	233	214	317	156	90	139	351	283	165	103

Table 1: Adoption over time

var	coeff	p-value
round	-5.325E-03	0.01
value	8.45E-08	0.01
constant	0.762713	0.00
$n = 2996, R^2 = 0.0039$		

Table 2: OLS regression on *seed*

farmer has clearly learned about it as of both time period 1 and 2. Obviously, a variable that simply represents use of a new technology does not accurately represent actual learning. Then, it would be necessary to create a new dummy variable that is 1 if the farmer has ever used a given technology as of time t .

The results from the analysis of the data were contrary to the nature of the study. There is no evident correlation between time and technology use. This is shown both by raw data computations and OLS regressions. As shown in Table 1, the number of farmers using new technology (the *seed* variable) does not strictly increase with time (the *round* variable). Instead, it seems to fluctuate. Perhaps this is due to fluctuations in the price of pineapples, the good for which this technology is implemented. As a result, I performed an OLS regression that controlled for the value of the crop, as shown in Table 2. When controlling for the value of the sale, we see that *round* still has a negative correlation. However, *value* does explain some change in technology usage over time. The regression tells us that when the value of a crop increases by one unit, the probability that new technology is used increases marginally. Notice that both variables are statistically significant at the 99% level of confidence, and both explain less than one percent of the variation in technology usage.

I discovered another variable in the dataset that was likely to be valuable in explaining technology use. *reproound* is defined as the “actual round of activity when this is not [the] same as round.” The idea is that *round* is the time period in which the seed use and crop value is reported, while *reproound* is when the seeds are actually used as an input. It seems likely that *reproound* would explain more about seed use than *round*. Thus, I ran a final regression using *round*, *value*, and *reproound* to explain technology use over time, as seen in Table 3. The result was dramatically different. In this case, the only variable that was significant at the 90% confidence level was *value*, with *round* and *reproound* only significant at the 85% confidence level. *round* and *reproound* are most likely not

var	coeff	p-value
round	0.201047	0.146
value	7.47E-08	0.044
rebound	-0.209672	0.129
constant	0.853803	0.00
$n = 1283, R^2 = 0.0109$		

Table 3: OLS regression on *seed*

independent variables, so there is probably some correlation occurring between the two.

There are no clear findings in this case. The best regression explains about 1% of the variation of *seed*, and time has a negative marginal effect on the use of net technology. There could be a few problems here that are correctable. One was mentioned in the data section above

5 Problems with the GA as an Empirical Model

Even if the Ghana data were consistent with economic theory of technology diffusion, there would still be numerous problems with using the GA as a model. First, the GA gives results in generations, not in time. So if we would like to make predictions using a genetic algorithm, our time period is arbitrary. We would have to scale generations to represent some amount of time. However, if we have to scale the model each time it is used, we cannot make very good predictions. We may be able to say something about the general shape of the adoption path (percent adoption over time), but we will lack actual figures.

Another significant problem with the genetic algorithm is that it requires a fitness function. This may be easily constructed if the cost curves of firms are known. But often cost curves and profit functions are unknown. We then cannot make decisive predictions if we do not know how to pressure selection in the genetic algorithm.

While these problems may hold us back from making predictions, we can still test if the GA is a good model of technology diffusion. Furthermore, we may be able to test hypothesis by only using the GA (without empirical data).

6 Credit Markets in Developing Economies

As I have discussed, there are many problems with adapting the GA to data. But suppose instead that we use a GA to generate an adoption path, then make a change to the GA and note the resulting change in the adoption path. We

are then not comparing apples to oranges anymore; we are taking note of how certain changes in an economy are reflected in the adoption of technology.

For instance, consider credit markets. Developing economies like India lack a trustworthy, liquid credit market. If we wish to measure the effect that credit markets have on the adoption of a new technology, we can do so with a GA. First, we create a GA that models technology adoption. Then, we alter the GA to include a simulated credit market. We can look at the change that this has on adoption, and relate this information to predict the impact of a credit market in India.

Taking this a step further, we can include many things in our GA. We may wish to include genes that code for risk aversion and accumulated wealth. We can then look at how credit markets impact different individuals and their choice of technology. We may also include a stochastic element in the fitness function, to simulate changing economic conditions or in the realm of agriculture, changing weather conditions.

As discussed previously, the mutation rate, $prob_{mutation}$, can actually be coded as a gene, allowing the rate of mutation to change and converge on a particular value. It may be possible to view the mutation rate of an individual, then, as the propensity to take risks. This risk aversion gene would be interesting, for it may converge on one value, or it may always vary over time. This would tell us if having a distribution of risk aversion is beneficial to a population, or if there is a single optimal value for risk aversion.

Another interesting study would involve altering the GA to include simulated policies like estate taxes. This would also likely have an effect on technology adoption, if the early adopters are the agents with the highest wealth. There is much more work to be done on this subject, and it can have a substantial impact on our understanding of topics like risk aversion, wealth accumulation, technology adoption, and a variety institutions.

7 Conclusion

The genetic algorithm is clearly a very useful technique in many fields, including Economics. However, it is questionable whether the GA will ever produce useful predictions as a model. As an alternative, I am exploring other ways in which the GA can make useful economic predictions. This is done by making changes to the GA (in the form of simulated policies) and looking at the resultant changes in the technology adoption path. Hopefully this will give insight into many problems facing developing countries.

8 Acknowledgements

Special thanks to Robert Fleck and Joseph Atwood. Insightful contributions made by David Buschena, Wendy Stock, Neil Cornish, John Paxton, Tomas Gideon, and Christina Hayes. This project has been made possible by generous funding by the Montana State University Undergraduate Scholars Program and the Montana State University Department of Agricultural Economics and Economics.

References

- [1] Banerjee, Abhijit V. “A Simple Model of Herd Behavior.” *The Quarterly Journal of Economics* 107 (August 1992) 797-817.
- [2] Besley, Timothy, and Anne Case. “Diffusion as a Learning Process: Evidence from HYV Cotton.” Working paper, Department of Economics, Princeton University, 1994.
- [3] Conley, Timothy, and Christopher Udry. “Social Learning through Networks: The Adoption of New Agricultural Technologies in Ghana.” *American Journal of Agricultural Economics* 83 (August 2001): 668-73.
- [4] Foster, Andrew D. and Mark R. Rosenzweig. “Learning by Doing and Learning from Others: Human Capital and Technical Change in Agriculture.” *Journal of Political Economy* 103 (1995) 1176-1209.
- [5] Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, Massachusetts: Addison-Wesley, 1989.
- [6] Holland, J. H. *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press, 1975.
- [7] Mitchell, M. *An Introduction to Genetic Algorithms*. Cambridge: MIT Press, 1996.
- [8] Riechmann, Thomas. *Learning in Economics: Analysis and Application of Genetic Algorithms*. Heidelberg: Physica-Verlag, 2001.