Increasing Genetic Algorithm Efficiency for Wire Antenna Design Using Clustering

D. S. Linden
R. MacMillan
Linden Innovation Research LLC
info@lindenir.com

Abstract

The Genetic Algorithm (GA) is a very robust, powerful technique that is capable of optimizing designs in very multimodal search spaces. However, it also requires significant numbers of simulations to perform such optimizations. If the simulations are expensive, as in the case of antenna design, GAs can be prohibitively expensive to use. A clustering technique has been investigated which cuts the required number of function calls 20-90% with minor or no degradation in the optimization quality. In this technique, a GA using real-valued genes is halted when the population has clustered around portions of the search space, and a local optimization technique completes the optimization quickly. This method has been applied to a variety of test functions and wire antenna designs, and the advantages of this technique seem to have broad applicability.

1.0 Introduction

Communication, radar and remote sensing systems employ thousands of different types of wire antennas, and there is an increasing need for high-performance, customized antennas. However, antenna design is a difficult field of engineering. Antenna designs have non-intuitive, complicated search spaces, and problems with even a few variables are highly multimodal. In addition, most antenna simulations require a significant amount of time to run. Typical simulations can take anywhere from a few seconds to several hours, so it is imperative to use an efficient yet robust method of optimization.

Genetic algorithms (GAs) [1, 2] are currently being explored with great success as a way to automate the antenna design process [3]. GAs are well suited to the multimodal, spiky search spaces of electromagnetic problems. Particularly useful is that the GA does not require an initial guess, and the amount of design information the engineer must supply can be very minimal.

In spite of their success, GAs with conventional convergence criteria require too many cost function evaluations for many antenna design problems. This research investigates using the clustering behavior of real-valued genes during a GA optimization as a way to determine convergence—a method that significantly enhances efficiency.

A GA begins with a random distribution of points across a search space. As the GA run progresses, order begins to appear in the population. For many optimization problems, the initial random distribution begins to cluster around certain points in the search space, and gene values begin to show organization, first in multi-modal, then unimodal, distributions as the GA converges. Once gene value distributions become clustered around points in the search space, the GA has probably found a number of hills which, barring unusually useful mutations, the GA will slowly begin to exploit. Members of the population that are fit enough to survive will generally be from one of these peaks. Peaks with individuals of greater fitness will gain more population members, and eventually the entire population will exist on a single peak and then a single point.

The GA can, however, be stopped as soon as the population has divided itself into a number of discrete clusters. A local optimizer can then be applied to each cluster. Because this clustering can occur early in the GA run, many cost function evaluations can be saved, usually with minor or no impact on the optimization results.

1.1 Real GAs and Adewuya's Method

The reader is probably familiar with binary GAs, in which all parameters are encoded into a string of bits called a chromosome. Any continuous parameters must be discretized, which means that resolution becomes a factor. The crossover processes for these GAs are also straightforward, involving swapping bits in some fashion to create children. However, previous research [4] has shown that real-valued GAs, where each gene in a chromosome is a real number, coupled with special crossover techniques, are much better at optimizing problems with all or nearly all parameters continuous.

These special crossover techniques involve the use of interpolation and extrapolation to create children. The method used by the authors was first investigated in [5], and is called Adewuya's method. Adewuya's method consists of a sequence of crossover methods applied to real genes. First, quadratic crossover is applied, where the child's gene is taken from a predicted minimum of a quadratic curve fit using three parents. If quadratic crossover fails, heuristic crossover is applied, which pulls the child's gene from a range predicted to be better than two parent's genes. See Figure 1 for a graphical representation of what happens in these two methods.

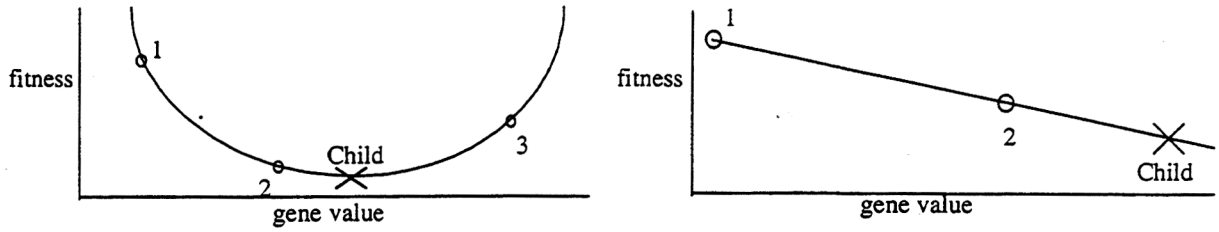

Figure 1. Quadratic and heuristic crossover. Fitness is to be minimized in these examples.

If both quadratic and heuristic crossover fail, the child's gene is one of the parent's genes taken at random. This process is applied gene by gene to create a new child. See [4, 5, 6] for a more complete explanation and comparisons with other methods. This method has been found to be particularly powerful in electromagnetics and mechanical engineering.

Mutation for the real valued GA can take many different forms. The one used here was Gaussian—mutated genes were pulled from a distribution with a mean equal to the unmutated gene, and a standard deviation of 0.1 of the full gene range.

Each gene varied over the same range. We chose this range to be from 0 to 1. Each gene is translated into parameter values as appropriate, and can cover very different ranges in the design space. However, normalizing the gene values in this way allows the accurate calculation of the genetic distance between individuals using Euclidean geometry, a very important quality when determining the clusters in a population.

Regarding other GA parameters, mating selection was accomplished via the weighted roulette wheel method of [2], and a steady-state GA was used, in which the parents of the next generation are the best of a specified percentage of the total population. This percentage is called the overlap, for it is the portion of each generation that carries over to the next. This type of GA has proved to converge quickly, a feature necessary to accommodate the costly simulation time of antenna designs. Fitness scaling was also used for the weighted roulette wheel, basing the amount of the roulette wheel given to an individual on the difference between the scores of that individual and the worst parent carried over from the previous generation.

1.2 Wire Antenna Design

Since F. Braun created the first wire antenna in 1898, a variety of wire antennas have appeared: monopoles (e.g., car whip antennas), log-periodic antennas (e.g., rooftop TV aerials), helix and spiral antennas, and a host of other types. In recent years, GAs have shown sufficiently powerful to optimize even very challenging designs for unusual applications [3,6,7,8]. Following is a definition of several antenna design terms that are important in this paper.

Directivity and *gain* are two related qualities in antenna design. Directivity is the ratio of power density being transmitted by an antenna in a particular direction to the average power density being transmitted in all directions. The gain is the directivity multiplied by the ratio of power radiated to power input. Gain takes into account the losses due to resistance in the antenna, which converts some of the input power into heat. When the losses are considered to be zero, as in this paper, the directivity and gain are equal.

Gain is usually expressed in decibels (dB), which relates to a ratio of power or power densities by the following expression: $dB = 10\log_{10}(P_1/P_2)$. In the case of gain, P_2 is the power density of an isotropic radiator that transmits power equally in all directions. The abbreviation dBi refers to gain compared with an isotropic radiator. However, the "i" is sometimes left off, and is understood from context.

A *gain pattern* or *antenna pattern* plots gain magnitude versus angle, showing the proportion of power an antenna transmits in a particular direction. For 2-D antennas, or antennas symmetric in the third dimension, this angle is simply the elevation angle θ. In 3-D, there are two angles that specify a direction: θ and the azimuth ϕ. Figure 2 shows these angles on a set of axes. An antenna is considered to be *directive* if its gain pattern is heavily weighted in one direction.

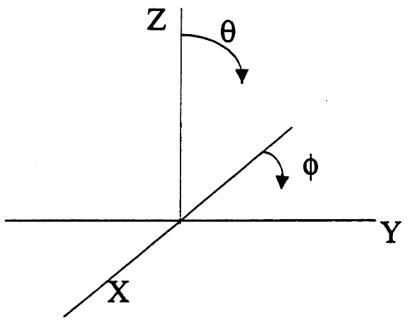

Figure 2. θ and ϕ on a 3-D axis system. Arrows begin where NEC2 defines 0 degrees for θ and ϕ.

A *ground plane*—at its simplest a large, flat metal plate underneath the antenna—is often used in conjunction with a wire antenna. It acts as a mirror for the antenna above it, and therefore changes the antenna gain pattern. A ground plane can decrease the height and/or simplify the construction of the wire antenna. The hood or roof of a car acts as a ground plane, and antennas that will be affixed to such places need to be designed for use with one.

There are several electromagnetic simulators that exist for wire antennas. One particularly suited to the task of creating a general antenna synthesis system is the Numerical Electromagnetics Code, Version 2 (NEC2) [9]. This code was used exclusively on this research. NEC2 has a simple file-interface for input and output that makes it ideal for using with an optimizer. The code is in the public domain, so obtaining and modifying the source code is cost-free and easy, as is copying the simulator between machines. But perhaps most important, it has a long track record of being accurate. The NEC2 code was produced in the early 1980s, and has been used it to simulate antenna structures for many years. It has shown itself to be in very good agreement with actual measurements, and thus one can have more confidence that answers received from simulation have validity.

There are three antennas that will be discussed in this paper: a two-wire Yagi antenna, a loaded monopole, and a 14-wire Yagi antenna.

1.2.1 The Two-wire Yagi

The Yagi antenna is a series of parallel wires, first proposed by Prof. Yagi and his student S. Uda in the late 1920s. One element is driven, one element is behind the driven element and is called the reflector, and, usually, there are other elements in front of the driven element called directors. The highest gain can be achieved along the axis and on the side with the directors. The reflector acts like a small ground plane, allowing power that would otherwise be sent backward to be reflected forward.

In this case, there are no directors—only the reflector and the driven element. This gives a two-dimensional problem, as shown in Figure 3. The chromosome for this antenna is two real genes, encoding length and separation respectively.

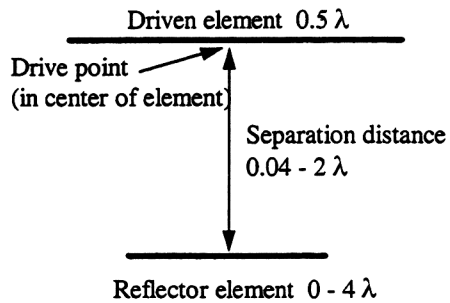

Figure 3. Two-element Yagi antenna search space. λ = 1 wavelength

In spite of the fact that there are only two variables, the response surface is very multi-modal, as shown. This behavior is typical for electromagnetic problems, which are usually filled with local minima. This behavior shows why GAs are one of the most powerful techniques for solving these problems—its parallel sampling of the search space makes it able to resist many of the local minima.

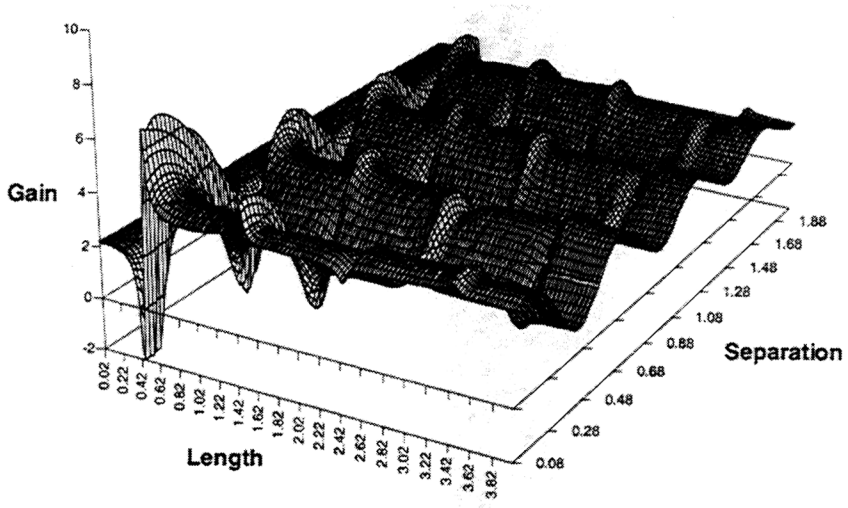

Figure 4. Response surface for gain vs. separation and reflector length.

The goal for this antenna is to maximize the forward gain, so the objective function for this antenna is simply the gain. As can be seen on the graph, the best parameter settings to maximize gain are a length of about 0.48λ and a separation of about 0.14λ. The figure below shows what the antenna pattern looks like near this maximum.

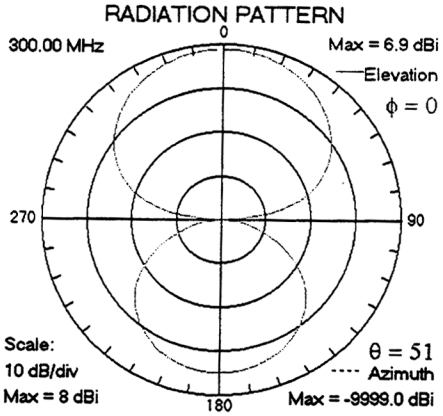

Figure 5. Radiation pattern of an antenna near the maximum

GAs optimizing this antenna show clustering in a very clear way, as will be described in the next section. But first, the other wire antennas, the loaded monopole and the 14-wire Yagi, need explanation.

1.2.2. The Loaded Monopole

A monopole loaded with a modified folded dipole has been previously investigated [10]. It has a search space as shown below.

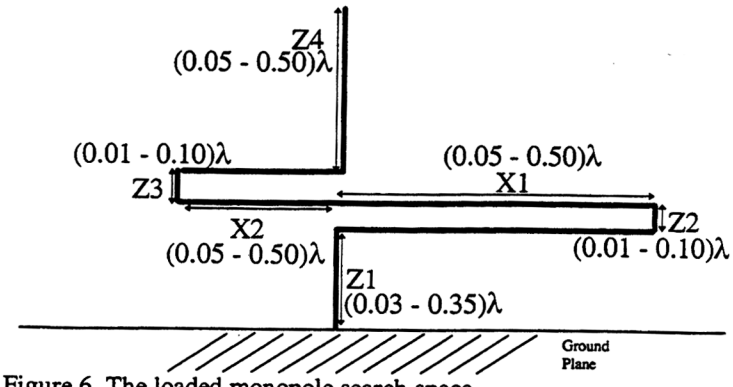

Figure 6. The loaded monopole search space

The chromosome for this antenna is six real-valued genes, encoding Z1 through Z4, then X1 and X2. However, the ordering makes no difference, because the crossover techniques described in section 1.1 are applied separately for each gene.

This antenna is capable of having even coverage over the upper hemisphere given the proper set of parameters [8]. The resulting pattern for one such configuration is shown in Figure 7.

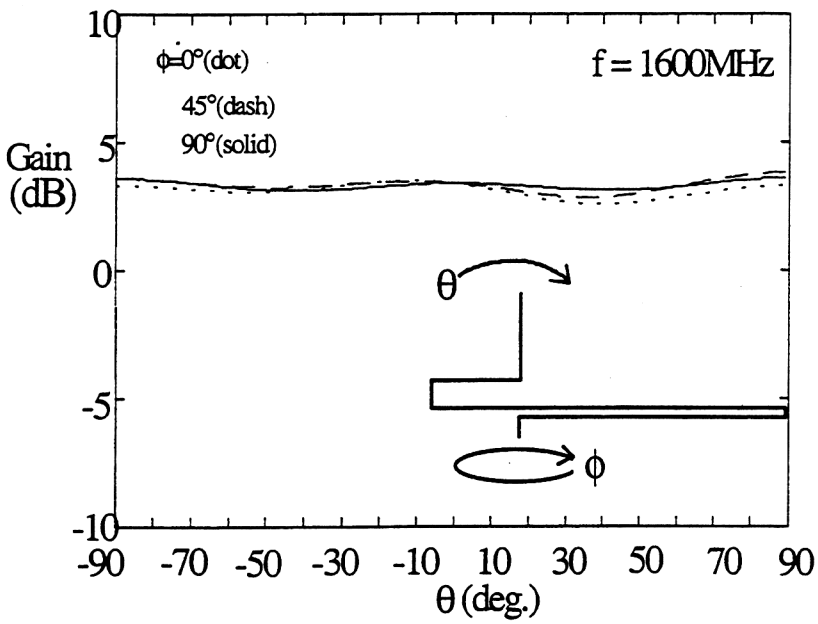

Figure 7. Folded monopole pattern and corresponding optimized design.

What is unusual is that the shape is so asymmetric. This asymmetry was an unexpected result, but further study showed it to be necessary to achieve the very flat pattern shown in Figure 7.

The objective function for this antenna is the sum of the squares of the deviation of all calculated gains from the mean. In equation form:

$$\text{Fitness} = \Sigma_{\text{over all } \theta,\phi}(\text{Gain}(\theta,\phi) - \text{Avg. Gain})^2 .$$

The GA's goal is to minimize this function.

1.2.3. The 14-wire Yagi

The 14-wire Yagi antenna is a more traditional Yagi antenna than the two-wire Yagi above, with a reflector, driven element, and 12 directors as shown below. This antenna optimization is the most challenging of all the examples, with 28 variables, multiple criteria, and a difficult, sensitive search space. It will show whether the clustering technique described in the paper will work on a truly difficult problem.

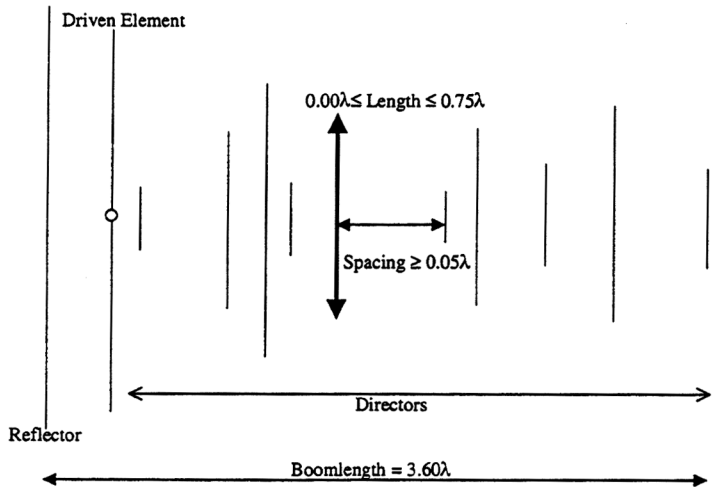

Figure 8. 14-wire Yagi design.

The real-valued chromosome consists of 14 length genes, 13 spacing genes, and a gene for wire diameter. Each wire length is allowed to vary between 0.0 λ (effectively removing the element) and 0.75λ. They are constrained to be symmetric.

The spacing between wires is constrained to be greater than 0.05λ. However, the boomlength is constrained to be 3.60λ, so the 14 wires are spaced along this length as follows: the values of the genes corresponding to the spacings are totaled, then the boomlength is divided by this total. This result is multiplied by each spacing gene value to give the required spacing between each pair of wires. The last variable is the wire diameter, which is allowed to vary between 0.004λ and 0.012λ.

The criteria were VSWR and endfire gain. The score was given by:

$$\text{Score} = G - C_1 \times (\text{VSWR})$$

where G is the endfire gain and C_1 is 10 when the VSWR is greater than 3.0 and 0.50 when the VSWR is less than 3.0. (It should also be noted that (VSWR-1.0) is used instead of VSWR when it is less than 3.0 to further decrease the importance of this factor on the score.) The objective was to maximize the score.

These three antennas will show, on a preliminary level, the applicability of the clustering technique described in the next section.

2.0 The Clustering Method

GAs usually begin with a randomly generated population, scattered stochastically around the search space. As survival of the fittest is applied, the population quickly begins to avoid unfruitful areas. Then, the population begins to cluster around certain places in the search space. What is happening is that those regions are loci of good fitness, and individuals produced within them are viable—i.e., they will have sufficient fitness to survive. Those that are produced outside of these regions will probably not have enough fitness to survive once the population is firmly clustered around these points. This effect can also be regarded as speciation, for intraspecies individuals, likely to remain inside a cluster, will survive, while interspecies individuals, likely to fall outside of any cluster, will perish.

One the population is clustered, there will be little exploration of the search space. What will happen is a "battle" in which the clusters fight for individuals. The better-scoring clusters will generally receive more of the new children, and as the scores increase, the lesser clusters will lose individuals, finally dying off one by one until only one cluster is left.

Following is a graphical representation of this process, taken from an optimization of the two-wire Yagi antenna using a real-valued GA.

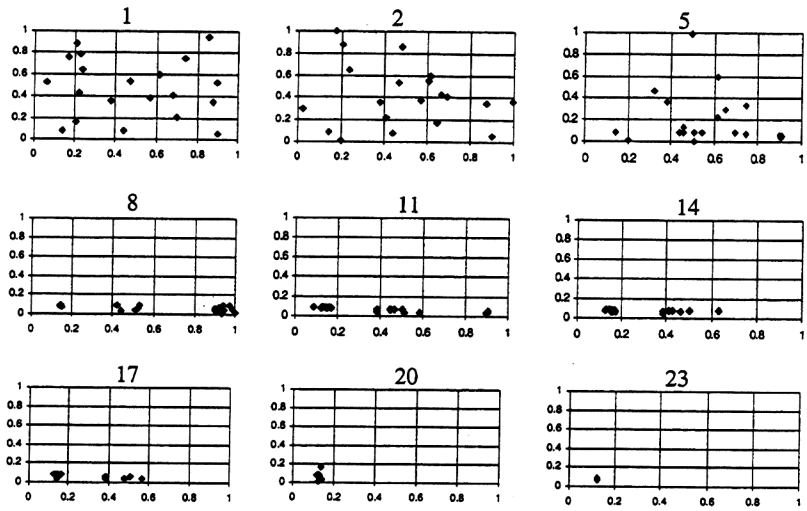

Figure 9. An example of clustering in the case of the 2-wire Yagi. From [6].

Generation 1 is randomly scattered throughout the space. As can be seen, the worst areas are avoided even beginning with generation 2, and by generation 8 the population is clustered around three points. From generation 8 through 23, nothing happens except gradual extinction of clusters until only the best remains. The GA requires just as many resources during this last process as it did when it was very effectively finding good regions in the search space.

It makes intuitive sense, then, that this battle is simply a waste of resources. Why not stop the GA when the population is clustered, and use a local optimizer on one or more of the clusters, since each cluster is probably a single peak in the search space?

The challenge in applying this idea is finding an automated technique that can detect when the population is properly clustered. Though there are many ways to determine this process, a simple approach was taken in this research, which involved using a threshold value for cluster radius, similar to [11].

To start the first cluster, the two closest individuals in the population, as determined by Euclidean distance, are clustered, if they are closer than the cluster threshold. The center point between them is calculated, then the nearest individual to this center point is added if its distance is less than the cluster threshold. The new center point of the cluster is calculated, the next-closest individual added, etc., until there are no other individuals within the cluster threshold distance from the cluster center.

The closest pair of individuals not already clustered is then checked to see if the distance between them is less than the cluster threshold. If it is, then a new cluster is formed in the manner of the first one. This process continues until there are no unclustered individuals closer to each other than the cluster threshold.

Once a specified percentage of parents is clustered, the GA is halted. As will be shown, this percentage makes a large difference on the effectiveness of this procedure, for if one halts the GA before a sufficient number of parents are clustered, the local optimization will not be very effective, for the best peak has not been sufficiently defined.

In addition, an elitist cluster routine was found to be the most effective. An elitist routine is one that specifies that regardless of the percentage of the parents that are clustered, the GA will not be halted until the best individual is clustered as well. A study comparing the elitist and non-elitist routines showed far better results with small additional computational expense for the elitist routine. This result is intuitive, for if the best individual is not in a region with a cluster, there is a good likelihood that the GA is not done exploring the space yet and there will still be some shifting in clusters before it is ready to be halted.

It was initially thought that the local optimizer might be most effective if it operated on the center of the cluster, as opposed to the best individual from the cluster. Study showed this was not the case; results were disappointing from the cluster center, but were very good from the best individual. For this reason, the score of the cluster is taken as the score of its best individual, and that individual is passed to the local optimizer when the GA is halted.

Before tuning the method, it was not known if one needed to optimize all clusters to be reasonably certain of getting the best answer, or if it was sufficient to optimize only the best cluster. We were surprised to learn that optimizing only the best individual, which is contained in the best cluster by default, is sufficient to produce excellent results. On rare occasions, the

second-best cluster actually was located on the best peak, but this happened so infrequently (less than 5% of the time) that the extra function calls necessary to optimize the second-best peak were deemed not worth the expense. However, this behavior needs to be explored for more problems, for it is conceivable that more complex problems in very spiky search spaces may show greater benefit when the less-fit clusters are optimized.

Though this process is tied to a specified cluster threshold, its effectiveness seems universal, and seems to be more effective with more difficult problems. The results of our experiments with this method will now be discussed.

3.0 Results

In this section, the effectiveness of the clustering routine is discussed for many different problems, including simple test functions, the two-wire Yagi, the loaded monopole, and the 14-wire Yagi. The routine seems to be effective on both simple and complex problems, as will be shown.

In order to compare the clustering method with a standard GA, a standard baseline GA needed to be created. This GA has the following convergence criteria, which are not particularly unusual: halting after the best individual has been static for 11 generations, or when the range of values present in the parents for each gene fall within 1% of the total gene range. After the GA is halted, a conjugate gradient local optimizer, the same as is used for the clustering method, is used to optimize the best individual.

3.1 Test functions

These functions were used to create and debug the clustering routine, though they could not be used to fine-tune the routine because they are so simple. However, they do show that the clustering routine is effective even for simple problems, and are included for completeness.

Three simple test cases were used. The first test case is De Jong's F5 [12], shown below. It has two dimensions, and a maximum value of 1.002.

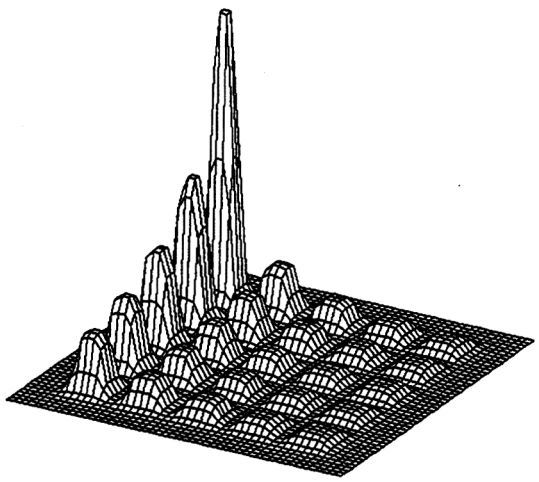

Figure 10. De Jong's F5 test function

The second test case is a low-modality sinusoidal function, given by the equation:

$$\text{Score} = \Sigma_{i=1 \text{ to } 6}(\sin(\pi x_i)-\cos(3\pi x_i))$$

One dimension is shown below:

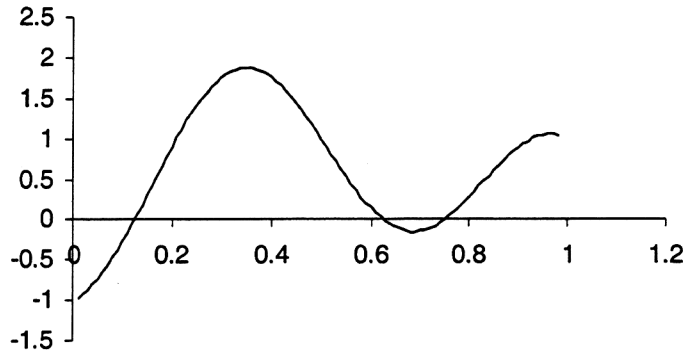

Figure 11. One dimension of sinusoidal test function #1

This function was tested in six dimensions, which has a maximum value of 11.272.

The third test case is a more challenging sinusoidal function, tested in 10 dimensions, has a maximum value of 20.0. Its equation is: Score = $\Sigma_{i=1 \text{ to } 10}(\sin(\pi x_i)-\cos(10\pi x_i))$

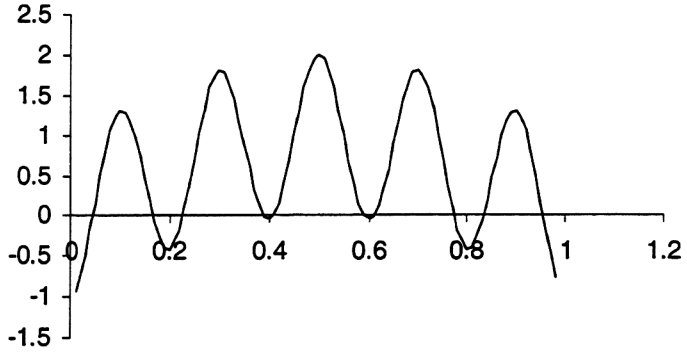

Figure 12. One dimension of sinusoidal test function #2

Each test case was tried over a range of population sizes (25, 50, 75, 100, 150, 200) and overlaps (0.25, 0.5, and 0.75). The results were averaged over all combinations of these two variables, to determine the overall effect of the method on results without bias toward a particular population or overlap value. The results of these experiments are contained in the table below.

	Average score			Average objective function calls		
	Baseline	Cluster method	% difference	Baseline	Cluster method	% difference
F5	0.789	0.724	-8.3%	693	459	-33.7%
Sin #1	10.9	11.0	0.3%	1236	669	-45.9%
Sin #2	18.9	17.9	-5.1%	1763	1358	-23.0%

Table 1. Results from test functions

For the F5 test function, the clustering method loses 8.3% in score while decreasing function calls by 33.7%, and both changes are statistically significant as shown by the student's t-test statistic.

The 2nd test case performed quite well. There was a statistically insignificant difference in the score between the clustering method and the baseline GA, while the decrease in function evaluations was 45.9%!

For the third case, the clustering technique took a loss of 5.1% on average score while providing only a 23.0% gain in efficiency. This is not particularly spectacular, though it is significant.

All three test cases showed varying degrees of improvement in runtime, between about 20% and 50%, and varying degrees of change (generally degradation) in optimization quality, between 0.3% and -10%. However, the real test of the method is in solving actual design problems, which will now be discussed.

3.2 The Two-wire Yagi

Two experiments were run with the two-wire Yagi: a parameter tuning experiment, and a confirmation of the clustering effect over a wide range of population sizes and overlaps.

3.2.1 Fine-tuning the clustering parameters

Though simple test cases showed improvement with this method, they were too simple to use in fine-tuning. The parameters for this method are the percentage of parents to be clustered before halting the GA and the cluster threshold (also called cluster size). Each was tuned preliminarily using the test functions above. However, the parameter values that worked for the test functions did not work at all well for the two-wire Yagi. Therefore, an experiment was run to determine the best values for these parameters for this more realistic engineering problem.

The data points shown in Table 4 are the average performance over three population sizes (25, 50 and 100) and two overlaps (0.25 and 0.5), which gives a broad indication of its effectiveness. The Cluster threshold (which is the maximum Euclidean distance between any two members in the cluster) was varied between 0.1 and 0.3, and the percentage of the parents that were required to be clustered varied from 70% and 90%. The resulting average scores and number of objective calls required to complete the optimization are shown below.

Cluster threshold	% clustered	Avg. score	Avg. objective function calls
0.1	70%	5.25	703
0.3	70%	5.55	523
0.1	90%	6.40	650
0.3	90%	5.29	592

Table 2. Cluster parameter experiment

The results show that the best scores resulted from a tight clustering threshold, and as large a percentage of the parents clustered as possible before halting the GA. Though these settings do not give the best time savings, the difference in scores make the extra simulations worthwhile. These settings make intuitive sense as well, for if the clusters are too large, the cluster may actually cover more than one local minimum, causing the local optimizer to fail. In addition, if some parents are not clustered, that means that some viable individuals are alone in their region of the search space, and they are probably on some sort of peak that should be investigated before halting the GA.

Further investigation showed that increasing the parent percentage clustered gave still better results, thus the parameter settings that were used for the rest of the experiments with clustering were 99% of parents clustered and 0.1 cluster threshold.

3.2.2 Confirming the effectiveness of the clustering method

Another full-factorial experiment shows the effectiveness of the clustering method on saving objective function calls while not significantly disrupting performance. The results are shown below.

As with the test cases, the baseline and clustering method were tried over a range of population sizes (25, 50, 75, 100, 150, 200) and overlaps (0.25, 0.5, and 0.75). The results were averaged over all combinations of these two variables, to determine the overall effect of the method on results without bias toward a particular population or overlap value. The results of these experiments are contained in the table below.

Average score			Average objective function calls		
Baseline	Cluster method	% difference	Baseline	Cluster method	% difference
6.241	6.395	2.5%	884	415	-53.0%

Table 3. Comparison of the baseline GA and the GA using the clustering method for the two-wire Yagi.

A student's t-test showed the difference in the baseline and clustering GA average scores were statistically insignificant, with a 44.5% probability it arose by chance. On the other hand, the difference in objective function calls is so significant that there is less than a 0.002% chance that it occurred by accident. The experiment also showed the best predictor of score performance was not clustering but population size. The data shows that the larger the population, the better the score, at least to the 200 individual population size. (Incidentally, previous research [6] has shown that too large a population can actually decrease performance.)

Of course, the major significance of these results is that the clustered GA requires less than 50% of the function calls that the baseline does, for essentially no change in score. This is a phenomenal result, but the next design case shows even greater improvement.

3.3 The Loaded Monopole

Using the tuned parameters of 99% of parents clustered and 0.1 cluster threshold size, the loaded monopole was optimized using the clustering method. A comparison of the two GAs follows, with population sizes and overlaps chosen for each at their optimal point as tuned by the two-wire Yagi experiment:

	Population	Overlap	Average Objectives	Average Score
Baseline	200	0.42	17736	18.1
Clustering	200	0.25	2078	67.3

Table 4. Baseline vs. Clustering GA performance for the loaded monopole

The baseline case was run 6 times, the clustering case 5. Both methods achieved very good designs, but there is an 88.3% savings in objective function calls using the clustering method! However, there is a statistically significant increase in the score for the clustered case. Recall that this objective function is to be minimized, with the ideal being zero. While this degradation may seem significant, the average difference of 49.1, distributed over the 1,188 angles in the objective function, equates to an additional 0.20 dB of variation per angle. This extra variation is insignificant to the design, especially in light of the expected fabrication tolerance and simulator accuracy.

However, this problem was fairly easy, so a more difficult problem is needed to show whether this method will be generally useful.

3.4 The 14-wire Yagi

Using the tuned parameters of 99% of parents clustered, the 14-wire Yagi was optimized using both methods. However, the cluster size made a significant difference in the resulting score of the Yagi antenna. Several runs were conducted with various cluster threshold values as shown below.

	Population	Overlap	Cluster threshold	Average objectives	Average score
Baseline	200	0.42	-	22299	16.29
Clustering	200	0.25	0.53	3549	14.94
	200	0.25	0.26	4930	15.51
	200	0.25	0.053	12898	16.22

Table 5. Baseline vs. Clustering GA performance for the 14-wire Yagi

Note that the largest two cluster threshold sizes used are larger than in the folded monopole, to account for the larger number of dimensions. However, the results show that increasing the cluster threshold caused significantly poorer scores.

In this case, a one-point difference in the score makes a big difference in the quality of the design, since Yagi antennas are desired to be as well-matched and as high-gain as possible. A drop of 1 point means a decrease of 1 dB of gain or a VSWR over 3.0. Thus, the difference in score between the baseline and the clustering method using a cluster size of 0.53 was unacceptable. The search space was too difficult to search with a local optimizer if the cluster had only converged to that size. However, by tightening up the size of the cluster, the clustering method was able to essentially match the baseline score, but in about 58% of the objective calls!

Incidentally, the gain of a typical Yagi with a score of 16.2 is 16.23 dB, with a VSWR of 1.06. A typical Yagi designed using conventional means has a gain of 15.9 and a VSWR of 1.23 [6].

Thus, there is a tremendous speed advantage to using this method for this and the previous time-intensive problems, and the price in design performance can be trivial if the proper settings are used.

Conclusion

In general, the clustering method shows significant, even remarkable, time savings over more typical methods of determining convergence. The time saved by using the clustering method is directly proportional to the decrease in the number of objective function calls for problems with any time-consuming simulations, as in wire antenna design. These savings can be as much as 90% without significant degradation to design performance.

However, the ideal settings for the method have been shown to be problem dependent, though the trend we have found is that the more the population is converged and the tighter the clusters are required to be, the less design performance degrades. Naturally, this performance is achieved at the expense of objective function calls. While good starting values seem to be 99% of the population clustered, and 0.1 cluster size (given a range of 0-1 for all genes), the best settings have to be determined on a case-by-case basis.

While the results presented here are very promising, there is much work that remains. First, an adaptive method of clustering that does not depend on an *a priori* setting of a cluster threshold is desired. Speciation techniques like mating restriction need to be tried with clustering to see if there is any advantage for encouraging early cluster formation beyond what the GA does normally. It would also be of interest to apply this method to a binary GA.

In addition, work must be done to refine the local optimizer, perhaps enhancing its ability to escape from small "traps," because it did not perform as well as expected, given that the cluster methods nearly always placed the local optimizer starting point fairly close to the optimum value. A "fully" converged GA often placed the local optimizer just a little closer to the true optimum—closer enough to produce a statistically significant difference in design performance in many cases.

In summary, then, the method of clustering described in this paper, though simple and relatively unsophisticated, shows tremendous promise at enhancing the efficiency of a GA. It showed time savings in every case it was applied, with the antenna design problems showing greater efficiency enhancement for less degradation in fitness than the test functions. This indicates that this method may be most effective for the problems where efficiency is most needed: large, time-consuming problems that are currently very difficult or even intractable using standard GA optimization.

References

[1] J.H. Holland, "Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975).
[2] David E. Goldberg, *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley (1989).
[3] Y. Rahmat-Samii and E. Michielssen, eds. *Electromagnetic System Design using Evolutionary Optimization: Genetic Algorithms*, Wiley, 1999.
[4] D.S. Linden. "Using a Real Chromosome in a Genetic Algorithm for Wire Antenna Optimization." Proceedings of the IEEE APS International Symposium, Montreal, Canada, 13-18 July 1997.
[5] A. Adewuya, " New Methods in Genetic Search with Real-valued Chromosomes," Master's Thesis, Mech. Engr. Dept., MIT, 1996
[6] D.S. Linden. "Automated Design and Optimization of Wire Antennas using Genetic Algorithms." Ph.D. Thesis, MIT, September 1997.
[7] D.S. Linden and E.E. Altshuler, "Automating Wire Antenna Design using Genetic Algorithms," Microwave Journal, Vol. 39, pp. 74-86, March 1996.
[8] E. E. Altshuler and D.S. Linden, "Design of a loaded monopole having hemispherical coverage using a genetic algorithm," IEEE Trans. Antennas Propagat., Vol. 45, pp. 1-4, Jan. 1997.
[9] G.J. Burke and A.J. Poffio, "Numerical Electromagnetics Code (NEC)-Method of moments, " Rep. UCID18834, Lawrence Livermore Lab. CA, Jan 1981.
[10] E.E. Altshuler, "A monopole antenna loaded with a modified folded dipole," IEEE Trans. Antennas Propagat. Vol. 41, pp. 871-876, July 1993.
[11] J.W. Duda and M.J. Jakiela. "Generation and Classification of Structural Topologies with Genetic Algorithm Speciation," Journal of Mechanical Design, Vol. 119, pp. 127-131, March 1997.
[12] De Jong, K.A. "An analysis of the behavior of a class of genetic adaptive systems." Doctoral Dissertation, University of Michigan, Dissertation Abstracts International 36(10), 5140B, University Microfilms No. 76-9381, 1975.