

USING GENETIC ALGORITHMS TO SHAPE SHORT LASER PULSES

Michael Ackerman
Advisor: Dr. Douglass Schumacher

National Science Foundation's Research Experience for
Undergraduates in Physics at The Ohio State University, 1999

OVERVIEW

The purpose of this project was to determine if genetic algorithms could be used to control continuum generation by short laser pulses. Previously, short laser pulses needed to be reshaped by hand using trial and error methods to generate the desired spectrum. The highly nonlinear nature of the interaction that generates these spectra makes this process both arduous and inexact. Our hope in embarking on this project was that by allowing a genetic algorithm to control and reshape the short laser pulse shape, we could more quickly and accurately control continuum generation.

BACKGROUND ON GENETIC ALGORITHMS

Prior to beginning my work with an actual genetic algorithm I referenced *Handbook of Genetic Algorithms* (Davis, Lawrence editor, 1991) and *Genetic Algorithms in Search, Optimization, and Machine Learning* (Goldberg, David E., 1989) for background information. The documentation that accompanied the GENESIS program we eventually chose to employ also provided a great deal of insight into how genetic algorithms work.

We decided to try to solve this problem using a genetic algorithm because genetic algorithms are well suited to solving highly complex problems such as that of continuum generation. Previous attempts at controlling continuum generation by manually shaping the short laser pulses had been made with some success.

Examples of spectra produced by these hand-shaped pulses can be seen in Figure 1.

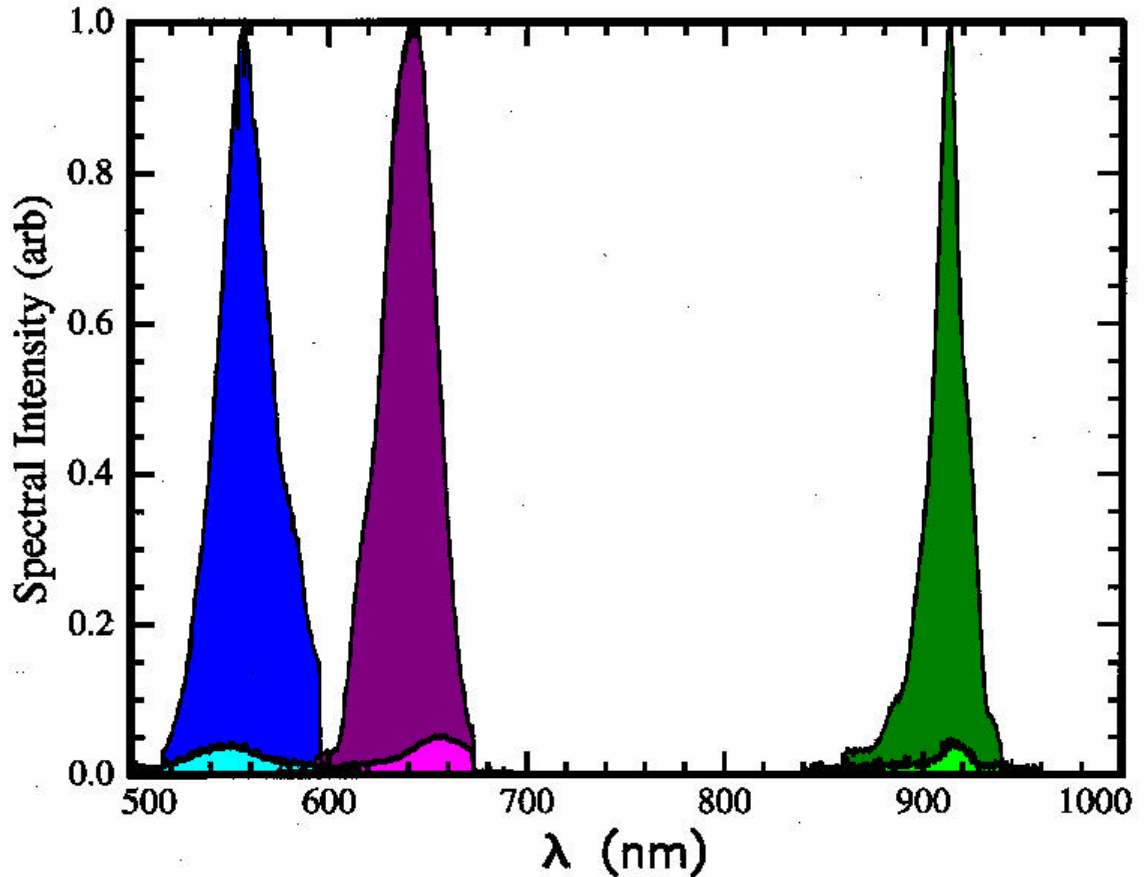


Figure 1: The lighter colors represent the spectra from an unshaped (Gaussian) pulse. The darker colors represent the spectra from hand-shaped pulses attempting to produce a Gaussian spectrum. The three peaks are the results of three independent experiments by Dr. Douglass Schumacher, each across a different range of wavelengths.

We also believed genetic algorithms to be potentially useful in this project because genetic algorithms operate without any "knowledge" of what they're working on. In other words, a genetic algorithm evolves a population using the same techniques regardless of how data is represented in the population. This was desirable because the physics underlying continuum generation is highly non-linear and therefore very complex.

THE STRUCTURE OF GENETIC ALGORITHMS

Genetic algorithms are programmed routines designed to mimic natural evolution. Just as nature evolves populations fit to fill a niche in the environment, genetic algorithms evolve populations adapted to solving given problems. Genetic algorithms attempt to do this by mimicking the mechanisms of natural evolution such as crossover, mutation, natural selection, and survival of the fittest. The details of how genetic algorithms implement these mechanisms can be found in texts such as Davis (1991) and Goldberg (1989).

A genetic algorithm applies these mechanisms to a population of "chromosomes", each of which has the same user defined structure but differs from other members of the population in its details. At the most fundamental level, each chromosome is simply a binary string of some set length. This string can represent a single number or a set of numbers in sequence, each of which represents some variable. By evaluating the fitness of each member of the population and applying the mechanisms of evolution, the genetic algorithm evolves a population whose members become more and more suited to solving the given problem with each generation.

DEVELOPING A GENETIC ALGORITHM

The first step in my project was to find a working genetic algorithm and familiarize myself with it. The genetic algorithm eventually chosen was John J. Grefenstette's GENESIS v5.0. We chose this program because it was fairly well documented, reputable, free, and easy to use. Another great advantage of GENESIS v5.0 was its "floating point representation" option. This option allowed us to think of each chromosome as a one-dimensional array of real numbers in some user-defined range. The conversion of this array to a bit string was done automatically by the GENESIS program. This saved us from having to decide how to encode our variables in a bit string.

The first alteration which needed to be made to the original GENESIS code was the combination of the setup program, which defined the structure of the chromosomes and set the options for the genetic algorithm, with the main program which ran the genetic algorithm itself and actually evolved a population. This was trivial, as the setup program was simply inserted into the main program as a subroutine.

The second alteration to GENESIS was much more difficult and time consuming. Grefenstette's original version of GENESIS was developed to run under UNIX. We eventually wanted GENESIS to be able to communicate with the software that controlled our lasers and interpreted the data from the spectrometer/line camera combination. This software was designed to run under Windows. The conversion of GENESIS to the Windows format was done with

Microsoft Visual Studio C++. The two primary problems that needed to be addressed in converting GENESIS were differences in the user interface and slight differences in how the C++ compiler executed some lines of code. I will not go into detail here about these problems, as it is not of particular interest, but simply state that eventually a version of GENESIS was created which ran stand-alone under Windows.

The final alteration made to the GENESIS code was to enable it to interface with the laser and spectrometer/line camera as part of its evaluation subroutine. It was this evaluation subroutine which determined the fitness of each member of the population for purposes of selection and reproduction. The code alterations required at this point were wholly beyond my programming capabilities and were done by my faculty advisor Dr. Douglass Schumacher.

EXPERIMENTAL SETUP

With a fully operational and implemented GENESIS genetic algorithm, we were ready to actually start running our experiment. The setup of the experiment was as illustrated in Figure 2. Each chromosome in our population was an array (using the floating point representation option in GENESIS) of sixty-four real numbers. The first thirty-two of these numbers represented phases (0-2) and the final thirty-two represented amplitudes (0-1). Each short laser pulse's shape was therefore

the sum of thirty-two component waves, each with its own amplitude and phase.

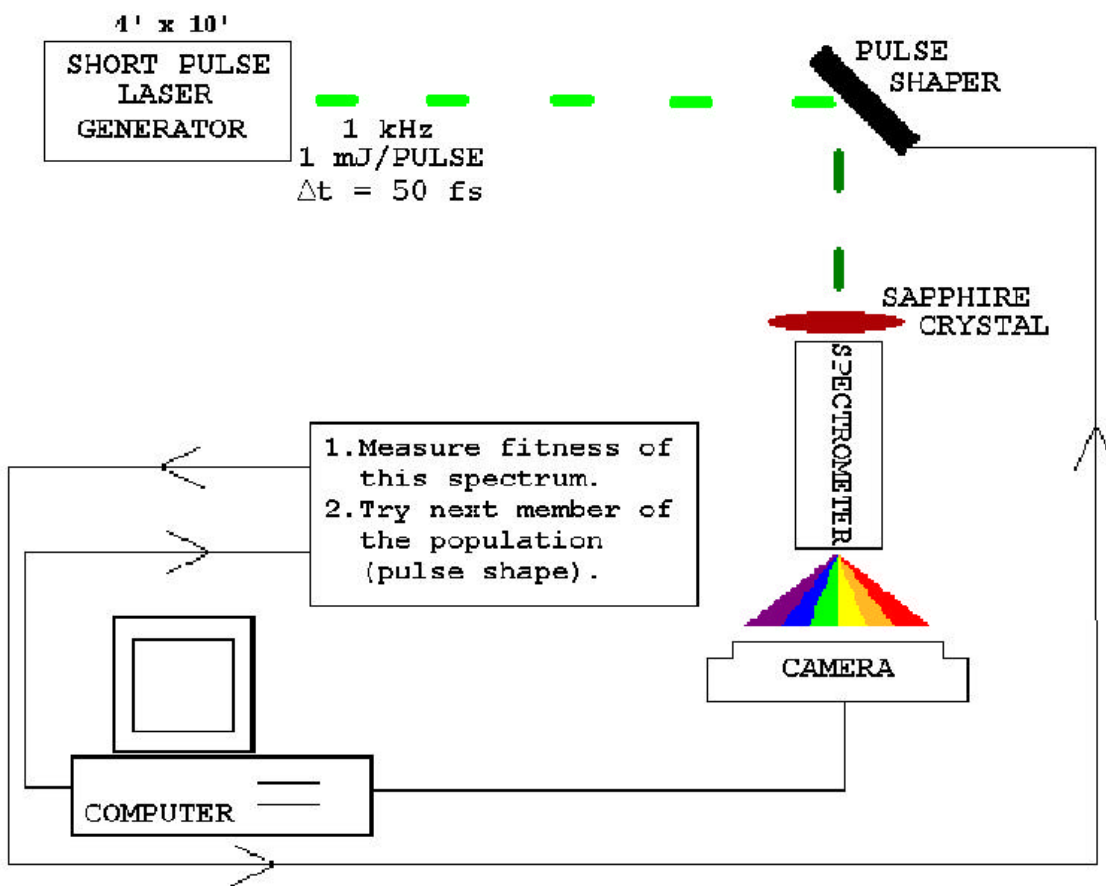


Figure 2

A short laser pulse (actually created in many steps but shown being emitted from a single box for simplicity) was shaped according to the variables encoded in each chromosome by a computer controlled liquid crystal pulse shaper. The reshaped pulse then passed through a spectrometer and a line camera read the resulting spectrum. The line camera sent its data to the

computer, which compared the produced spectrum to a preprogrammed target spectrum. The more closely the spectrums matched one another, the better the chromosome that shaped the incident pulse was considered to be for purposes of evolving the next generation.

OUR RESULTS

Our results were encouraging for a first run. Figure 3 and Figure 4 show results obtained using our first method of spectrum evaluation. This method simply took the squares of the differences in intensity between the actual and target spectrums at each point in the range of interest and summed them together. A higher number indicated a poorer match and therefore a less fit chromosome.

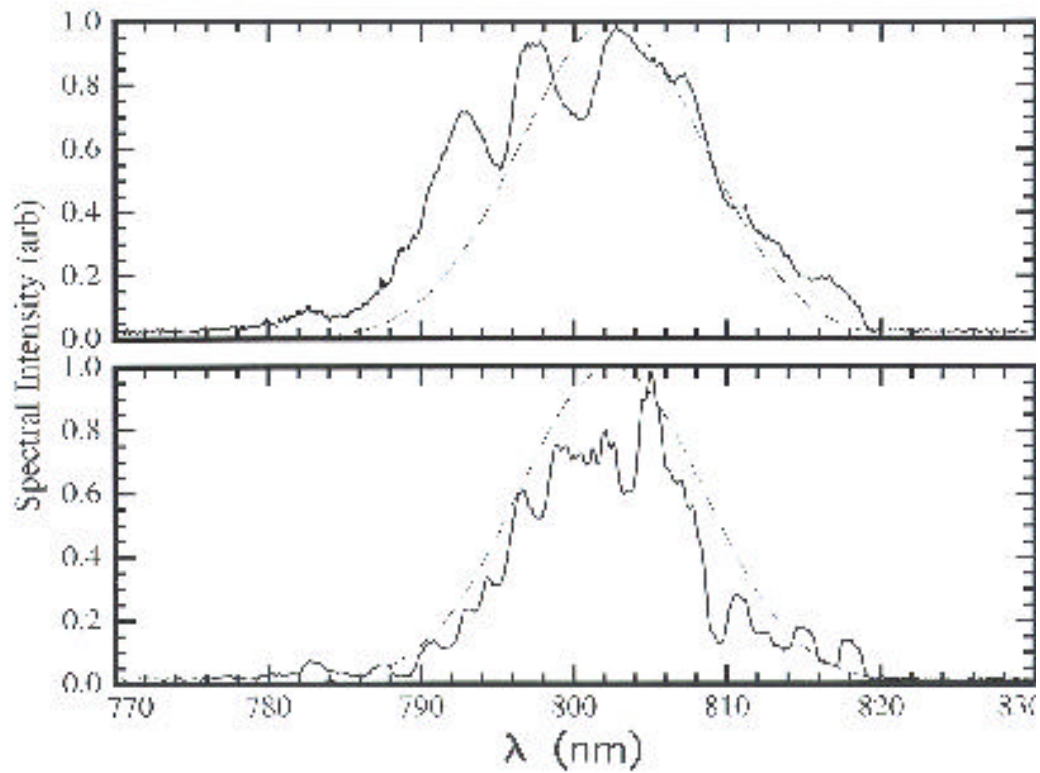


Figure 3: Best results of one run of the experiment. The dashed lines represent the target spectrum. The top solid line represents the spectrum from an unshaped (Gaussian) pulse. The bottom solid line represents the spectrum obtained using the pulse shaped by the genetic algorithm.

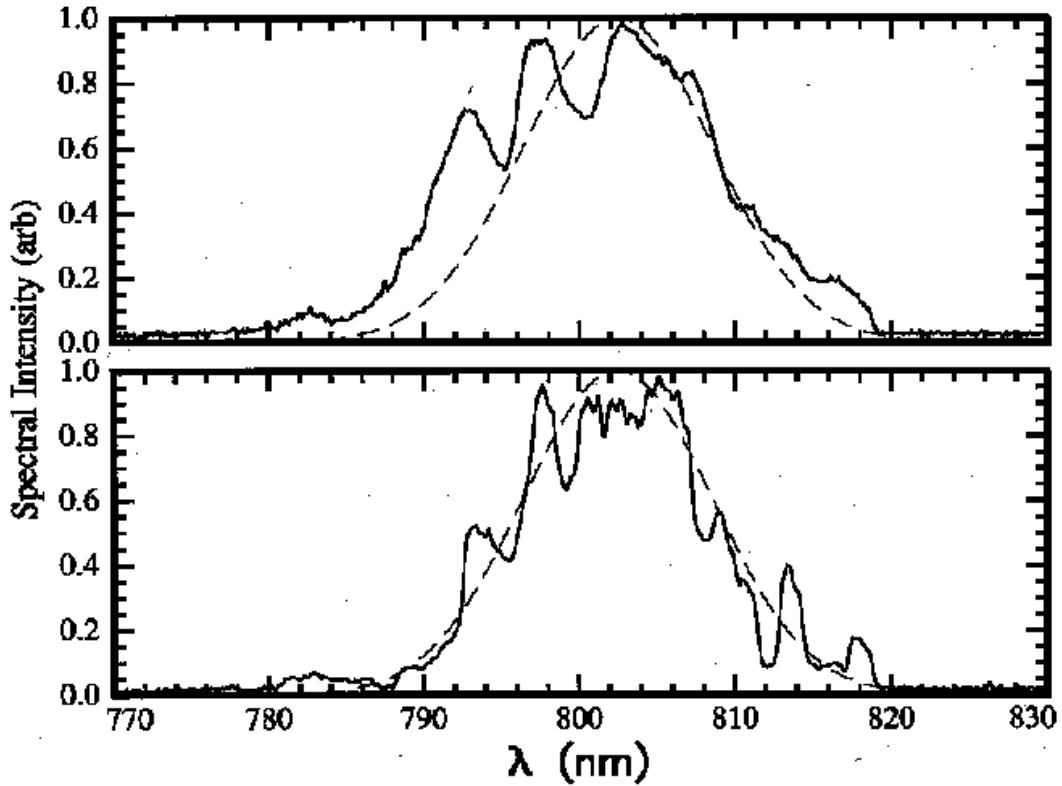


Figure 4: Best results of one run of the experiment with the same parameters as Figure 3 but a different initial population. The dashed lines represent the target spectrum. The top solid line represents the spectrum from an unshaped (Gaussian) pulse. The bottom solid line represents the spectrum obtained using the pulse shaped by the genetic algorithm.

Figure 5 shows results obtained by a second means of spectrum evaluation. This method was comprised of multiplying our target and actual spectrums and integrating the product of the two. In this case, a higher number indicated a better match and therefore a more fit chromosome.

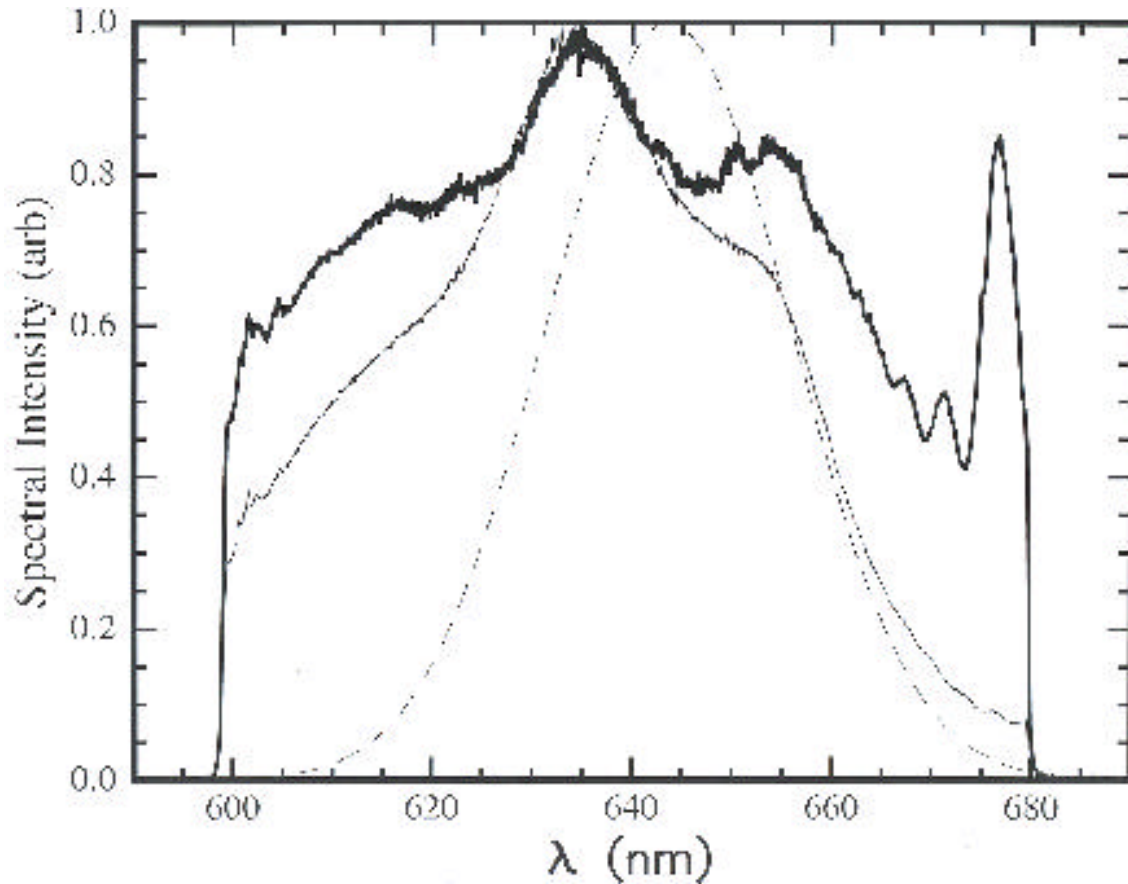


Figure 5: The dashed line represents the target spectrum. The darkest line represents the spectrum from an unshaped (Gaussian) pulse. The lighter line represents the spectrum from a genetic algorithm shaped pulse.

Each of these trials ran through approximately seventeen generations with a population size of sixty chromosomes. In each case, the improvements made by the genetic algorithm are clear. The genetic algorithm approach to controlling continuum generation definitely seems promising.

FUTURE ALTERATIONS TO THE EXPERIMENT

Increasing the size of the chromosomes to include more phases and amplitudes will allow the genetic algorithm to take

more control and make finer alterations to the pulse shape, hopefully resulting in even better matching spectra. A second improvement to the experiment would be to increase the speed at which the spectrometer/line camera can take data. At this point, the speed at which the experiment runs is not constrained by the speed of the genetic algorithm but by the speed of data taking. This alteration would of course not change our final results but would allow us to get better results in a shorter amount of time as more chromosomes could be evaluated and more generations run through. Finally, there is the genetic algorithm itself. We will continue to try new ways of evaluating the fitness of each chromosome's resulting spectrum as well as continue to try new combinations of the options that dictate how GENESIS produces a new generation from the previous generation.

CONCLUSION

We have found that genetic algorithms can indeed be useful in shaping short pulse lasers to control continuum generation. Our initial experimental results are encouraging and we expect further improvements as we fine-tune our experimental setup, including the genetic algorithm itself. Eventually, the results produced by genetic algorithms might help us to understand the complex physics underlying continuum generation, serving to further our ability to control this phenomenon. The implications of this research could be substantial to our understanding of short pulse laser interactions and therefore to any field of science which makes use of them.

REFERENCES

Davis, Lawrence; *Handbook of Genetic Algorithms*;
International Thomson Computer Press, London; 1991

Goldberg, David E.; *Genetic Algorithms in Search,
Optimization, and Machine Learning*; Addison-Wesley Pub.
Co., Reading, MA; 1989

For a copy of John J. Grefenstette's original GENESIS v5.0 code,
go to <http://www.aic.nrl.navy.mil/galist/src/#C>