Time domain pulse shaping using a genetic algorithm

by

Andrew Mori

Thesis presented in partial fulfillment of the requirements for the degree of Masters of Science

at

Stellenbosch University

Physics Department

Faculty of Science

Supervisor: Professor Erich Rohwer
Co-supervisor: Dr Lourens Botha

Date: December 2009

Declaration

By Submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the owner of the copyright thereof (unless to the extent explicity otherwise stated) and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: October 12th 2009

Abstract

Through the use of complex Laser Pulse Shaping, numerous fundamental laser induced processes may be controlled as well as studied. This work serves as an introduction into Laser Pulse Shaping, with the focus on a simple Pulse Shaping experiment, as well as to determine whether future, more complex processes may be similarly controlled.

A description of Laser Pulse Shaping theory is presented here, along with a full explanation of a simple experiment to maximize second harmonic generation (SHG) through Pulse Shaping. This experiment is simple on a theoretical level yet complicated in both implementation as well as operation. The experimental setup and software integration required hardware compatibility in multiple programming languages. This work was successful in the sense that a fully automated dispersion compensation system, accomplished through the use of a genetic algorithm in a feedback controlled loop, was constructed and tested.

The success of this experiment and the understanding gained in this work has laid the foundation for further complex Pulse Shaping systems to be achieved in future.

Opsomming

Komplekse Laserpuls-vervorming kan gebruik word om verskeie fundamentele laser-geinduseerde prosesse beide te beheer asook te bestudeer. Hierdie navorsingstuk dien as n inleiding tot Laserpuls-vervorming, spesifiek gefokus op n eenvoudige Pulsvervormings-eksperiment. Meer komplekse Pulsvervormingsopstellings kan toegepas word deur die kennis opgedoen in hierdie tesis.

Die teoretiese agtergrond van Laserpuls-vervormings word bespreek, tesame met n eenvoudige eksperiment om die Tweede Harmoniek Skeppingsproses (SHG) te maksimeer deur van Laserpuls-vervorming gebruik te maak. Die eksperiment is teoreties eenvoudig, waar die implimentering asook bedryf meer kompleks is. Die bedryf van die eksperiment word in 2 dele hanteer: die hoofprogram en n Genetiese Algoritme gebruik in optimering. Die werking van Genetiese Algoritmes asook Ultrakort Pulse (USPs) en pulskarakterisering word ook bespreek.

Die suksesvolle opstelling van die experiment en ook die eind resultate wat gevind is, maak dit moontlik om meer ingewikkeld komplekse laserpulsvervorming experimente te bestudeer.

Acknowledgements

I have come into contact with many people over the past 2-3 years who have helped me greatly, by providing academic guidance and support. Firstly my thanks goes mainly to the Femtosecond research group here at the National Laser Center. To Dr Lourens Botha, thank you for all the time and effort you put into helping me complete this work. Your guidance and expertise helped me a great deal. To Dr Anton Du Plessis, thank you for the time spent with me on general academic discussions and the odd technical advice given to me in the laboratory. I wish you the best of luck leading the femto-second group to new heights. To Dr Ted Roberts, thank you for always being available to help me grasp concepts and for the many laughs shared. To professor Erich Rohwer, my supervisor at Stellenbosch University, thank you for all your advice over the past years, including all the knowledge past onto me during lectures and laboratory sessions.

Thanks to Hendrik Maat, the laboratory was always in immaculate condition which allowed me to conduct this research with very few hassles. Thanks for all the support you provided. To Cobus Jacobs, thank you so much for all the time you spent discussing topics with me. Your help was always appreciated, as well as the many hours spent in the late afternoons! To Edward Bernhardi, thank you for the effort you put into helping me stay motivated and for the long friendship and academic path we have followed up until this point. To Melanie Mclaren, thank you for making the bad days bearable and for all the fun we had together. Thanks must be made to all my colleagues at the NLC and at Stellenbosch university, it was a privilege to be part of such a great team.

To my friends who spent hours of time on 'extra curricular' activities with me, your friendship is priceless. Special thanks to 'dave, the daveness, phd', thanks for all the laughs and great times. Thanks to the fish, 'fetus', 'styx' and 'vaas'. Thanks for always being there for me, I wish you all the best in the years that lie ahead.

And finally to my family, thanks so much for always supporting me. Having such a great, functioning family has always left me inspired. I can only hope to be as a good a parent as you have been to me. To my siblings I wish you all success and happiness and hope to see you soon.

To my family.

Contents

List of Figures

List of Tables

Chapter 1 - Problem Statement and Aim

Femtosecond laser amplifiers can generate nearly transform limited pulses. As these pulses are amplified or propagated through optical components they become lengthened and lose their shape. Maintaining a certain pulse duration after any form of propagation or amplification requires the use of a dispersion compensating technique or device. To automate the process for an arbitrarily broadened pulse requires the construction of a feedback loop and an optimization algorithm.

The work presented here is not a novel concept. Second Harmonic Generation maximization and pulse shape optimization for the purpose of creating transform limited pulses has been successfully reported [1, 2, 3]. This involved the use of second harmonic generation (SHG) signal that was produced by an arbitrarily broadened pulse, as input into an optimization algorithm that searched the corresponding parameter space for the shortest pulse duration. The pulse shaping method used in each differs in terms of the device that was used. In most cases either a Liquid Crystal Modulator in a 4f zero order dispersion line or an Acousto Optic Programmable Dispersive Filter (AOPDF) was used as a spatial light modulator.

The purpose of this research is to form a basis for future work on coherent control of atoms and molecules, by developing the genetic algorithm and feedback control loop and then by testing it in a simple coherent control type experiment. Using the SHG signal generated by an arbitrarily 'chirped' pulse as input into a genetic algorithm, the shortest pulse duration was found. The successful outcome of this experiment allows the possibility to use a genetic algorithm to search for optimum pulse shapes. Generating complex pulse shapes requires greater control than simply manipulating pulse duration. The process of searching for optimum complex pulse shapes will require searching more than one parameter space.

The field of coherent control relies heavily on pulse shaping and more specifically on tailoring optimum complex pulse shapes, which are usually unknown [1]. A learning/adaptive algorithm such as the genetic algorithm presented here, is used to search for the optimum shape. This shaping process requires a feedback-loop with some form of input to optimize. Besides forming a direct base for further development, this work also provides an introduction into pulse shaping theory, physical control and experience on a pulse shaping system as well as the intricacies involved in the construction of a functional feedback loop.

The feedback-control loop consists of three components, a dynamic link

library (DLL) containing a genetic algorithm, feedback control coming from the second harmonic generation and photo-detector/oscilloscope combination as well as direct control of the shaping device. These three components are all controlled simultaneously through a main program written in LABVIEW ™.

Chapter 2 - An Introduction to Ultra Short Laser Pulses

2.1 Introduction

A general understanding of ultra short pulses (USP's) is necessary in order to be able to shape them correctly. A pulse should be characterized before an attempt to shape it into a different form is made. Pulsed laser light differs from continuous wave laser light because the light intensity is not constant over time. Pulsed lasers produce light in short bursts with time periods between each burst which consist of significantly small amounts or no light at all. The types of pulsed lasers vary dramatically in terms of the energy per pulse, the time between pulses, the time duration of each pulse, along with numerous other characterization parameters.

2.2 Ultra Short Pulses

Ultra short pulses refer to laser pulses which have full width at half maximum (FWHM) time duration measurements in the sub-picosecond range. More specifically, FWHM measurements between 1×10^{-12} s (pico-second regime), and 1×10^{-18} s (atto-second regime) and even lower if possible, although most current USP research is conducted in the 1×10^{-15} s (femto-second) domain. These extremely short time spans lead to pulses having high peak intensities. In order to understand what peak power means, consider an ordinary household light bulb.

A common light bulb produces a continuous output of 60 Watts, which is equivalent to radiating 60 Joules of energy per second. A common pulsed femtosecond laser produces an average output power P_{ave} of 1 Watt, or 1 Joule of energy per second. This is in fact not a physical description of the laser pulses themselves, but only an average power measurement. This means that the light bulb produces more light energy than an extremely complicated and expensive pulsed laser system. Instead of observing continuous light, laser pulses are actually striking the detector at a specific repetition rate, varying from laser to laser. The laser pulses coming from the laser oscillator used in this work arrive at roughly 76 million times per second, or at a repetition rate of 76 MHz. This means that each pulse has an energy of,

$$E_{\mathrm{pp}} = \frac{P_{\mathrm{ave}}}{\mathrm{reprate}} \tag{2.1}$$

$$E_{\text{pp}} = \frac{1 \text{ W}}{76 \text{ MHz}} = 13.1 \times 10^{-9} \text{ J} = 13.1 \text{ nJ}. \tag{2.2}$$

For a Gaussian shaped pulse, this translates to a peak power of,

$$P_{\text{peak}} = \frac{2 \times E_{\text{pp}}}{\text{FWHM}_{\text{t}}} = \frac{2 \times 13.1 \text{ nJ}}{100 \text{ fs}} = 262 \text{ kW}. \tag{2.3}$$

If the same amount of energy per pulse is maintained while producing shorter and shorter pulses, the peak power would increase. These high peak powers are one of the main driving forces behind moving to shorter and shorter pulse durations. When these high intensities interact with matter, interesting phenomena may be investigated.

2.3 Pulse characterization

Light consists of oscillating magnetic and electric fields that propagate perpendicular to each other, with the latter being the dominant field of the two. An individual, theoretical pulse's oscillatory electric field is plotted against time in figure 2.1. Another way to depict a laser pulse in the time domain is to plot the intensity of the pulse as a function of time. Lasers are able to produce various intensity-time profiles, ranging from Gaussian to Top-Hat profiles. These forms can be achieved through different methods, although some are naturally created in laser resonators. Figure 2.2 illustrates different intensity versus time profile examples.

If one were to measure the pulse intensity as a function of time then the pulse duration could be determined from a FWHM measurement.

Understanding the spectral structure of the pulse can provide valuable information. It is essential to obtain an intensity versus wavelength plot. From this, the wavelength bandwidth and indirectly the frequency bandwidth of the pulse may be determined. An intensity versus wavelength and intensity versus time profile measurement are shown in figure 2.3 along with fitted functions. This curve fitting process is explained in Appendix A.

The reason for stressing the importance of understanding the pulse is that when using the pulse in other applications, the pulse characterization procedure may provide information on limits of what the pulse may and may not achieve. Using USP's in optical systems may cause damage to components which have damage thresholds and wavelength operation restrictions. Pulse characterization can reduce the risk of damaging optical components.

In shaping procedures and experiments falling outside the scope of this work, it is necessary to characterize the electric field of USP's. This is needed in search of an optimum pulse shape [1].

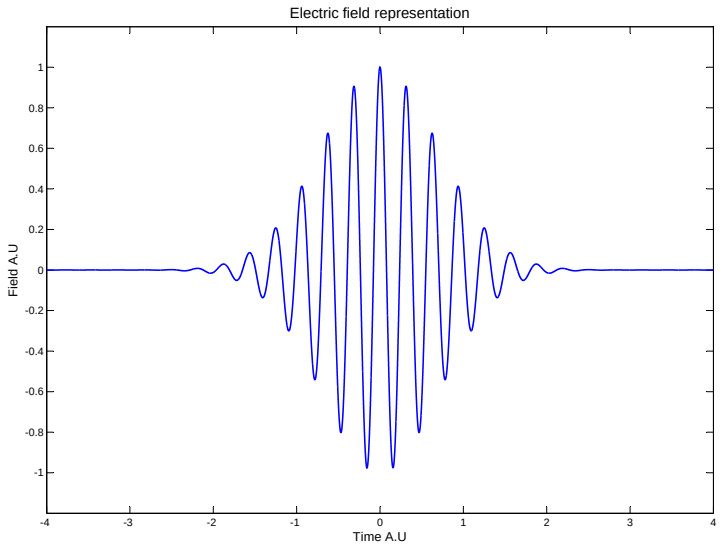

Figure 2.1: Auxiliary field representation of an individual transform limited pulse.

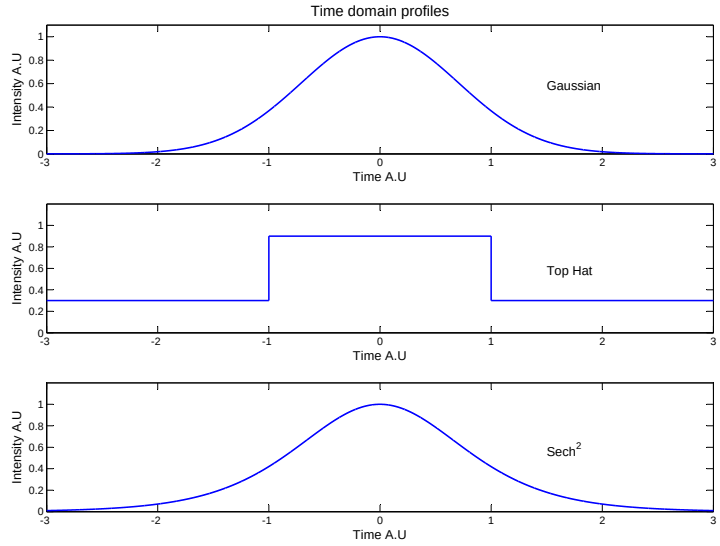

Figure 2.2: Three different intensity versus time profiles. Shapes like these can be achieved through specific mode configurations within the laser resonator or by external optical manipulation.

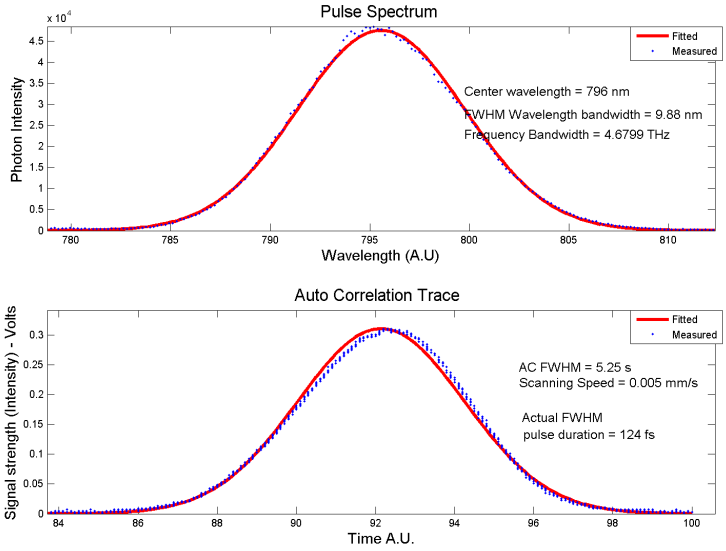

Figure 2.3: An USP's spectrum and time profile. The figures also depict the spectral width, frequency bandwidth and time duration.

2.4 Ultra short pulse generation

Most USP's are generated by modelocked lasers. In this mode of operation, a single USP propagates in the laser cavity, creating an exiting USP every time it strikes the output coupler, which usually is a highly reflective mirror surface, yet does transmit a percentage of light. The pulses that pass through the output coupler are in general quite short in comparison to the round trip time of the cavity. Understanding modelocking is made easier by considering the process in the frequency domain. The laser actually operates on a number of axial cavity modes, whereby all these modes oscillate in phase.

The mode amplitudes interfere constructively only at certain times which occur with the period of the round-trip time of the cavity, given by,

$$T = \frac{2L}{c}. \tag{2.4}$$

This round-trip time also defines the repetition rate of the mode-locked laser and is actually equal to the inverse of the frequency spacing of the different longitudinal modes.

$$f_{rep} = \frac{1}{T} = \Delta\upsilon \tag{2.5}$$

For an inhomogeneously broadened gain medium, the laser oscillates simultaneously over all the resonance frequencies of the cavity for which the unsaturated gain is greater than cavity losses. These frequencies make up the set

of longitudinal modes of the laser. When operating in this multimode regime, the output intensity of the laser is no longer constant in time. Its time dependent intensity distribution depends essentially on the phase relations that exist between these modes. The first graph in Figure 2.4 depicts a single mode. Graph b shows the resultant intensity of two modes in phase. Graphs c and d show the resultant intensity oscillation of eight modes with random phases and fixed phase relations respectively. The eight modes with random phases produce a random distribution of intensity maxima. The eight modes with a fixed phase relation show a time distribution of a periodic repetition of a wave packet. This figure and explanation has been taken from [4].

The term 'modelocking' was derived from the observation that a fixed phase relationship between oscillating modes is needed in order to produce short pulses. The achievable pulse duration is then inversely proportional the number of locked modes times their frequency spacing.

Mechanisms for modelocking are grouped into two categories, passive and active techniques. Active modelocking is achieved with an element generating a loss modulation that is precisely synchronized with the repetition rate or cavity round trips. Passive schemes rely on a passive loss modulation in the form of a saturable absorber. Passive modelocking can generally lead to the generation of shorter pulse durations due to the fact that passive loss modulation can occur on much faster timescales.

Usually, modelocked lasers are optically pumped with continuous wave laser sources. This requires a gain medium that can store the excitation energy for relatively long periods of time, longer than one cavity round trip. Most femtosecond oscillators have cavity round trip times in the order of a few nanoseconds, and most solid state gain media have upper laser level lifetimes in the order of a few microseconds and even into time spans of a few milliseconds.

2.4.0.1 Active mode-locking

Typical actively modelocked schemes include an electrical device such as an Acousto Optic Modulator (AOM). Standing waves with frequencies in the MHz range are generated with a piezoelectric transducer. As a consequence of this a periodically modulated refractive index is obtained. This periodically modulates the laser light by refraction. The refracted beam is lost and does not oscillate inside the cavity. A pulse circulating in the cavity can be ensured minimal loss provided that it always arrives at the AOM at a time where the refractive index is not changed.

Careful attention to cavity losses introduced by the index of refraction modulation and the AOM crystal itself, as well as the limited gain needs to be made.

Actively modelocked lasers making use of gain media such as $Nd^{3+} : YAG$ or $Nd^{3+} : YLF$ [5, 6, 7] can typically produce pulse durations of a few tens of picoseconds. For significantly shorter pulses, passive modelocking techniques need to be employed.

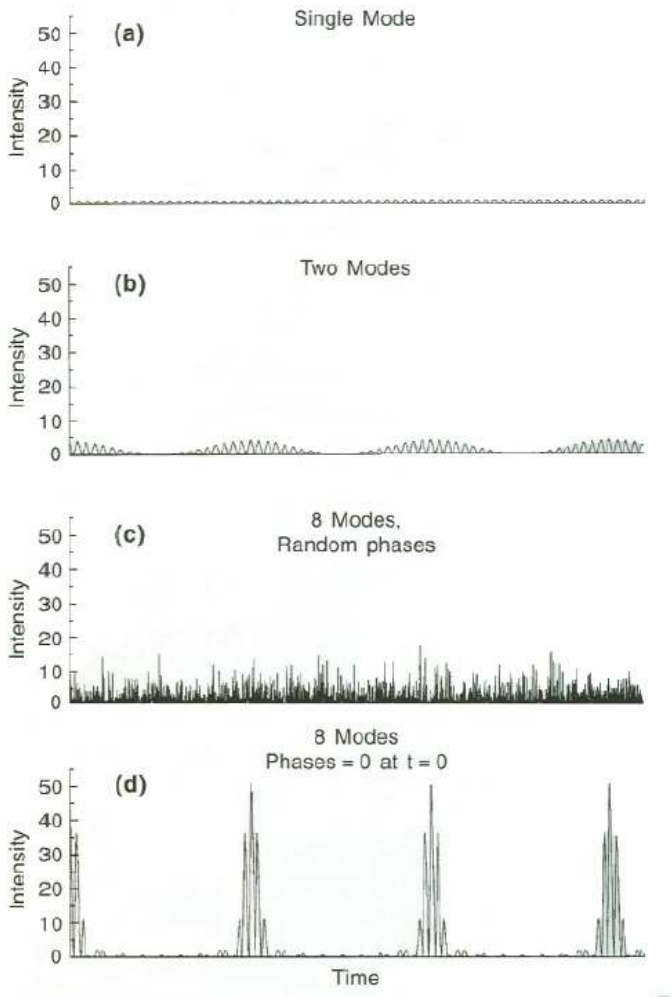

Figure 2.4: This figure stresses the importance of phase relations between oscillating modes to the time distribution of laser intensity (Rulliere, 1998). The intensity scale of the four above graphs remain constant. In graph a.) the intensity profile remains constant with only subtle variation. As more modes are introduced, the intensity profile becomes erratic with random peaks and troughs c.). Only with fixed phase relationships between modes, does a structured intensity profile emerge, visible in graph b.) and d.).

2.4.0.2 Passive mode-locking

This modelocking scheme employs a saturable absorber which favors the generation of a train of short pulses against other modes of operation. Starting from a continuous wave regime, the saturable absorber will favor any small noise

spike, and amplify this spike to the demise of the background. As these spikes grow they will come to a point where they contain a significant amount of the circulating energy. These spikes will saturate the gain so that the continuous wave background will decay. The most energetic noise spike which experiences the least amount of saturable absorption, will eliminate all the other energetic spikes, by saturating the gain to a level where they experience a net loss every round trip. As the outcome of this process, a final single circulating pulse is produced [8] and the pulses appear with frequency as determined by the round trip time.

The duration of the pulse may be reduced even further as the saturable absorber favors the peak of the pulse over the wings, amplifying the peak and reducing the wings. This occurs up until a point where broadening effects become strong enough to prohibit further pulse shortening.

A saturable absorber with long recovery time (low saturation intensity) is the most effective approach to self-starting modelocking. This is not the approach followed in generating the shortest possible pulses as a short recovery time allows the shorter durations to be achieved. Recovery time is an essential USP generating parameter. A fast saturable absorber can recover on a faster time scale than the pulse duration which causes strong shaping of the leading and trailing edges. If the pulse duration is significantly short the absorber recovery can take a longer than the pulse duration. This regime is usually used in ultrafast lasers as the choice of saturable absorbers becomes limited for pulse durations lower than 100 fs. It is possible to generate pulses with durations well below 100 femtoseconds with saturable absorbers that are much slower [9].

Chapter 3 - Time domain pulse shaping

3.1 Introduction

The control of pulse shapes, both simple and complex, have become integral in numerous optical fields including high field physics [10], optical communications [11, 12], ultra-fast spectroscopy [13] and coherent control [17]. The theory of ultra short pulse shaping is introduced in this chapter as well as a description of methods to counteract the effect of pulse broadening due to dispersion. The field of pulse shaping has undergone rapid development, and there are now a number of powerful shaping techniques in use in many optical laboratories. These methods differ drastically yet the theoretical understanding of pulse amplitude and phase manipulation remains consistent in all. In this work, a technique which makes use of an acoustic-optic programmable dispersive filter (AOPDF) is used.

3.2 Linear Pulse Propagation

In all amplifiers, there is an amplification medium through which the laser light is passed. When this amplification medium (usually of crystalline structure) is placed directly in the electric field of the laser light, the electrostatic forces associated with the field act on the atoms and molecules. This physically separates the positive and negative electric charges within the atomic structure of the medium. This effect is known as polarization and the oscillating polarization (induced by oscillating incident electric field) adds an extra term to Maxwell's equation for plane waves propagating along the z-axis,

$$\frac{\partial^2 E(z,t)}{\partial z^2} = \mu_0 \varepsilon_0 \varepsilon_r \frac{\partial^2 E(z,t)}{\partial t^2} + \mu_0 \frac{\partial^2 P(z,t)}{\partial t^2}. \tag{3.1}$$

When looking for solutions to this equation, the polarization needs to be related to the incident electric field in terms of the above discussion. This is the critical junction between linear and non-linear optics. The physical mechanism through which the interaction between the electric field and polarization takes place is known as the electric susceptibility, χ. In the case where the incident electric field is much smaller than the atomic field created by electrons and nuclei, it may be assumed that a linear relationship exists between the vector quantities P and E related by,

$$P(t) = \varepsilon_0 \chi E(t). \tag{3.2}$$

As the incident electric field strength approaches the intra-atomic field strength, the interaction mechanism becomes non-linear. The non-linear terms of the relationship between polarization and incident field can not be ignored (as there is no threshold) and instead must be expanded as a power series,

$$P(t) = \varepsilon_0 \left[\chi^{(1)} E(t) + \chi^{(2)} E(t)E(t) + \chi^{(3)} E(t)E(t)E(t) + ... \right]. \tag{3.3}$$

The first order of the expansion corresponds to equation 3.2 and the following terms are higher order non-linear effects. Typically the second order term $\chi^{(2)}$ is orders of magnitude smaller than $\chi^{(1)}$ and similarly $\chi^{(3)}$ is orders of magnitude smaller than $\chi^{(2)}$. This means that the effects of the higher order polarization terms only have an effect at high optical power densities. In this chapter, linear optics will be mainly dealt with, as well as linear pulse propagation, assuming that the second and third order non-linear susceptibilities are zero.

3.3 Gaussian Pulses

It is possible to deal with different pulse shapes, but for simplicity and convenience, let us consider the Gaussian pulse shape, with a carrier frequency ω_0 and complex envelope [14],

$$\varepsilon(t) = exp(-at^2)exp(i(\omega_0 t + bt^2)). \tag{3.4}$$

The first exponent describes the width of the pulse, according to the size of the parameter a and the second exponent describes the phase of the pulse. The instantaneous intensity is written as,

$$I(t) = |\varepsilon(t)|^2 = exp(-2at^2). \tag{3.5}$$

This relates the FWHM pulse width τ_p to the parameter a by,

$$\tau_p = \sqrt{\frac{2ln2}{a}}. \tag{3.6}$$

A pulse has been theoretically constructed with a known pulse shape, pulse duration, amplitude, intensity and carrier frequency. In order to characterize the pulse completely, the spectrum of the pulse must be described. The simplification of choosing a Gaussian pulse shape (besides the fact that it is a good approximation to the pulse shape coming from typical Titanium sapphire femtosecond lasers) is in fact because a Gaussian pulse in time, Fourier transforms into a Gaussian spectrum in frequency. This allows for an easy calculation to produce the pulse bandwidth, which is centered around its carrier frequency. This means that the pulse is actually constructed from a continuum of frequencies within this Gaussian pulse. Each frequency contributes a different amplitude to the pulse with the center frequency contributing the largest amplitude.

3.3.1 Time bandwidth product

The pulse time-bandwidth product for Gaussian pulses is given by [14],

$$\triangle f_p \tau_p = (\frac{2ln2}{\pi}) \times \sqrt{1 + (b/a)^2}. \tag{3.7}$$

In the field of ultra-short pulses, there are restrictions on the length of the pulse duration and on the width of the frequency bandwidth. This restriction follows directly from Heisenberg's uncertainty principle and states that there is a minimum limit on the value of the product of the pulse duration and the pulse frequency bandwidth. This minimum value varies for different pulse shapes.

To minimize this product, the chirp parameter b must be zero, leaving $\triangle f_p \tau_p \approx 0.44$. A Gaussian pulse with a nonzero imaginary part b is said to have a linearly time-varying instantaneous frequency. This is what is meant when a pulse is said to be chirped, with the parameter b being a measure of this chirp. Pulses with zero chirp are referred to as transform limited pulses.

In equation 3.7, b is the chirp parameter. If the pulses is assumed to be transform limited, where $b = 0$, a 100 femtosecond pulse duration corresponds to a frequency bandwidth of,

$$\triangle f_p \tau_p = (\frac{2ln2}{\pi}) \approx 0.44 \Longrightarrow f_p = 4,4 \text{ THz}. \tag{3.8}$$

3.3.2 Instantaneous frequency and Chirped Pulses

The instantaneous frequency of a pulse can be an important concept when dealing with polychromatic (non - monochromatic) light sources. For an oscillatory signal such as a sinusoidal signal, the instantaneous frequency is constant and is equal to the oscillation frequency where as in the case of chirped pulses, the instantaneous frequency has a time dependence. This is simply because the oscillation frequency changes with time. Instantaneous frequency is defined mathematically by [14],

$$\nu = \frac{1}{2\pi} \frac{d\phi}{dt}. \tag{3.9}$$

The $\frac{1}{2\pi}$ factor is present in order to convert it from angular frequency to frequency. In order to be able to manipulate pulse duration an understanding of chirped pulses is needed. Figure 3.1 displays three pulses. The first plot depicts a pulse with zero chirp, in time, and is a representation of the pulse oscillation field. The next two plots display pulses with positive and negative chirp.

Ultra-short pulses have large frequency bandwidths. If manipulated correctly these pulses may be altered in time. In essence this is because each frequency component may experience a shift in its phase relative to the other frequency components present within the pulse. Taking advantage of this property and the large frequency bandwidths, allows ultra short pulses to be stretched

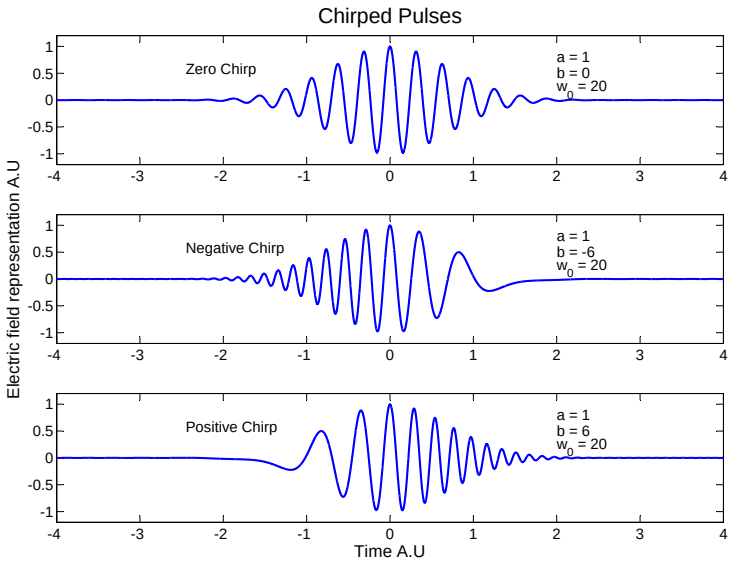

Figure 3.1: Auxiliary field representations of differently chirped pulses

to time durations orders of magnitude larger than its original time duration and also the possibility to be compressed back to a minimum transform limited pulse duration.

3.4 Linear pulse propagation theory in dispersive media

Consider a dispersive medium such as an atomic medium, prism or grating. A mathematical description of a dispersive medium would be any linear system which has a propagation factor [14], $\beta(\omega)$ which is frequency dependent and which has any other form besides $\beta = \omega/c$ which would mean no dispersion at all. If the signal has a narrow band, with frequency components situated primarily around some center frequency ω_0, then the propagation constant may now be expanded around its value at ω_0,

$$\beta(\omega) = \beta(\omega_0) + \beta' \times (\omega - \omega_0) + \frac{1}{2}\beta'' \times (\omega - \omega_0)^2 \tag{3.10}$$

where $\beta' \equiv d\beta/d\omega$ and $\beta'' \equiv d^2\beta/d\omega^2$ are both evaluated at $\omega = \omega_0$. Consider a Gaussian pulse of the form of equation 3.4 in time, propagating through such a dispersive system. The pulse spectrum can be obtained from the Fourier transform of equation 3.4 and thus the initial pulse spectrum is also of Gaussian form.

$$\tilde{E_0}(\omega) = exp[-\frac{(\omega-\omega_0)^2}{4\Gamma_0}], \Gamma_0 \equiv a_0 - ib_0 \tag{3.11}$$

It is now possible to calculate the output pulse spectrum after propagating a certain distance through a dispersive system. More precisely, this would be the input pulse spectrum multiplied by the frequency dependent propagation constant $\beta(\omega)$ over the dispersion length z.

$$\tilde{E}(z,\omega) = \tilde{E_0}(\omega) \times exp[-i\beta(\omega)] \tag{3.12}$$

The output pulse in time, after propagating through a dispersion length of z may now be calculated from the Fourier transform of the output spectrum, defined as,

$$\varepsilon(z,t) \equiv \int_{-\infty}^{+\infty} \tilde{E}(z,\omega)e^{i\omega t}d\omega. \tag{3.13}$$

The output pulse in time, after traveling a distance z through the dispersive system may now be written as,

$$\varepsilon(z,t) = exp[i(\omega_0 t - \beta(\omega_0)z)] \times exp[-\Gamma(z) \times (t-\beta' z)^2] \tag{3.14}$$

$$= exp[i\omega_0(t - \frac{z}{\upsilon_\phi(\omega_0)})] \times exp[-\Gamma(z)] \times (t - \frac{z}{\upsilon_g(\omega_0)})^2] \tag{3.15}$$

where

$$\upsilon_\phi(\omega_0) \equiv \omega_0/\beta(\omega_0) \tag{3.16}$$

and

$$\upsilon_g(\omega_0) \equiv 1/\beta'(\omega_0). \tag{3.17}$$

Equation 3.15 mathematically describes two physical attributes known as phase and group velocity.

3.4.1 Phase Velocity

This describes the velocity of the sinusoidal waves within pulse envelope. The first exponent in equation 3.15 implies a phase delay $\beta(\omega)z$ over a distance z, on the carrier frequency. This may also be thought of as phase delay t_ϕ in time which is simply related to phase velocity.

$$\text{Phase delay : } t_\phi = \frac{z}{\upsilon\phi(\omega_0)} = \frac{\beta(\omega_0)}{\omega_0}z$$

$$\text{Phase velocity: } \upsilon_\phi(\omega_0) = \frac{z}{t_\phi} = \frac{\omega_0}{\beta(\omega_0)}$$

3.4.2 Group Velocity Dispersion

The second exponent in equation 3.15 describes the group velocity at which the pulse envelope moves forward. The pulse envelope experiences a group delay time which is once again related to a group velocity.

$$\text{Group delay : } t_g = \frac{z}{\upsilon_g(\omega_0)} = \beta' z$$

$$\text{Group velocity} : v_g(\omega_0) = \frac{1}{(d\beta/d\omega)} \mid_{\omega_0}$$

If it were possible to monitor the propagation of the pulse through the dispersive system it would be observed that the pulse envelope moves forward at the group velocity and the individual cycles within the pulse move with the phase velocity. There are three physically possible cases. Firstly, for a pulse propagating in a non-dispersive medium (air is also a dispersive medium, so only in the case of vacuum propagation), the pulse would maintain its shape and pulse duration if the phase and group velocities were in phase. In the case where the group velocity is faster than the phase velocity $v_g(\omega_0) > v_\phi(\omega_0)$ it would be seen that the individual cycles would trail behind the pulse envelope. Similarly in the case of the phase velocity being greater than the group velocity, the individual cycles would run into the pulse envelope. In both these last two cases, the pulse would stretch and loose its original shape.

Controlling the rate at which this change between group velocity and phase velocity occurs, allows us to change the pulse duration and is known as Group Velocity Dispersion (GVD). GVD is described mathematically as [14],

$$\beta'' \equiv \frac{d^2\beta}{d\omega^2} \mid_{\omega_0} = \frac{d}{d\omega}.(\frac{1}{v_g(\omega_0)}) \tag{3.18}$$

In order to stretch or compress a pulse to a certain output duration, an amount of GVD needs to be imposed on the pulse. It is possible to calculate the amount of GVD imposed by monitoring the pulse parameters as the pulse propagates through a dispersive medium. Equation 3.4 and equation 3.6 describe the Gaussian pulse parameter a. Rewriting equation 3.6 yields,

$$a = \frac{2ln2}{\tau_p^2}. \tag{3.19}$$

As the pulse propagates through a dispersive medium, this Gaussian pulse parameter a changes. Equation 3.20 makes it possible to monitor this change. Taking a_0 as the input pulse parameter, then $a(z)$, the pulse parameter after a given dispersion distance z is written as,

$$a(z) = \frac{a_0}{(1 + 2\beta'' z b_0)^2 + (2\beta'' z a_0)^2}. \tag{3.20}$$

This equation describes the pulse parameter a after propagating a certain distance z in a dispersive medium, in terms of its original pulse parameter, the original pulse chirp and the amount of GVD introduced. It is from this equation that it is possible to calculate the GVD needed to produce a FWHM pulse duration implied from $a(z)$ from a FWHM pulse duration implied from the original pulse parameter a. It is also possible to monitor the chirp of the pulse in a similar manner from the following equation.

$$b(z) = \frac{b_0(1 + 2\beta'' z b_0) + 2\beta'' z a_0^2}{(1 + 2\beta'' z b_0)^2 + (2\beta'' z a_0)^2} \tag{3.21}$$

In equation 3.21, b is the original chirp of the input pulse, and $b(z)$ is the chirp of the pulse after propagating a certain distance in the dispersive medium.

The derivation of these equations are given explicitly and in two methods [14]. It is now possible to calculate how much GVD is needed in order to stretch or compress a pulse to a required pulse duration. An issue that arises in this process is that during compression, it is only possible to compress a pulse back to its transform limited duration if the optimum amount of GVD is introduced, or if the pulse propagates a specific dispersion distance. If this is not controlled, then the compression may overshoot and start stretching the pulse instead. A simple method is given [14] which calculates the optimum dispersion length needed to compress the pulse to its minimum value.

To do this, differentiate equation 3.20 with respect to the dispersion length $2\beta'' z$. This produces the maximum value of a(z) (which corresponds to a minimum FWHM pulse duration) which occurs for an optimum propagation distance.

Let $x = 2\beta'' z$, and set

$$\frac{d(a(z))}{dx} = 0 \Rightarrow \frac{d}{dx}\left(\frac{a_0}{(1+b_0 x)^2 + (a_0 x)^2}\right) \tag{3.22}$$

$$\Rightarrow (2\beta'')_{opt} = \frac{-b_0}{a_0^2 + b_0^2}. \tag{3.23}$$

The optimum pulse parameter may now be written as,

$$a_{opt} = a_0[1 + (b_0/a_0)^2]. \tag{3.24}$$

It is important to note that at the optimum pulse parameter a_{opt}, the pulse is transform limited and at its minimum pulse duration, implying that all chirp has been removed, leaving $b_{opt} = 0$. Large compression factors can only be achieved if the initial pulse chirp is great, or more specifically in the case where $b_0 >> a_0$.

The above discussions on GVD, stretching and compressing are general for ultra short pulses. The mathematical discussion was based on Gaussian pulses but the same ideas may be applied to different pulse shapes. The GVD discussed is referring to dispersion introduced by any optical system and is non-specific. There are a number of different optical elements and media that may produce this effect.

3.4.3 Pulse Broadening

Ultra-short pulses are usually used in applications that take full advantage of their spectral bandwidth. A transform limited pulse has a spectral width defined by the Fourier transform of its time-domain profile. Various laser types produce various pulses, in terms of time profiles. The 120 fs Gaussian shaped laser pulses produced by the oscillator used in this research, have spectral bandwidths of close to 10 nm. In contrast a 10 fs pulse has spectral width of 94 nm, which is considerably greater. As previously mentioned, this means that a pulse is constituted of spectral or frequency components, distributed about the

central frequency, each with different amplitude contributions. A transform limited pulse is said to be at its shortest pulse duration. This is due to the fact that the frequency components within the pulse are propagating simultaneously and at the same velocity.

As a short pulse propagates through any dispersive media, these frequency components become separated from each other in time. The components propagate with different velocities, and the pulse is stretched to a longer pulse duration.

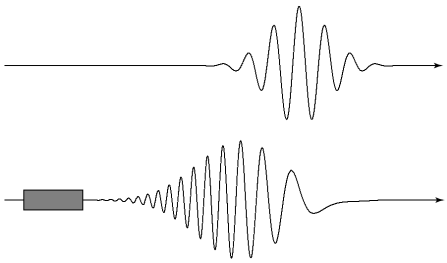

Figure 3.2: The top pulse propagates freely, where as the second pulse has passed through a dispersive medium.

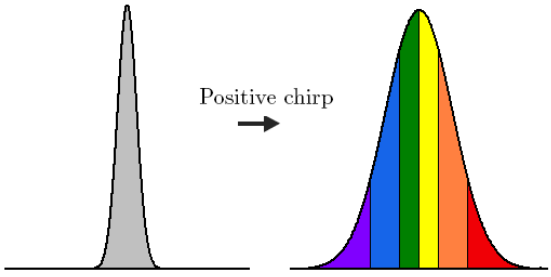

Figure 3.3: An illustration of positive chirp. The chirp free pulse is depicted in Grey, where the spectral components are in phase and indistinguishable from each other. The second pulse shows the blue components are delayed with respect to the red components.

Figure 3.2 depicts two pulses, one traveling freely, and the other has propagated through a dispersive medium, say a glass block for instance, and then allowed to continue unhindered. Two main issues are pointed out [15], the first one being that the center of the second pulse is delayed with respect to the

center of the first pulse. The speed that electromagnetic radiation propagates varies depending on the medium through which it travels. Light travels faster through air than it does in glass, yet slower than in vacuum. This is obviously not a broadening effect and is aptly called the group delay described in section 3.4.1. The second point to note is that dispersive media such as glass imposes a positive frequency 'chirp' on the pulse, delaying the blue components with respect to the red components, illustrated by figure 3.3. This is due to the individual spectral components traveling at slightly different speeds. The different propagation speeds for each spectral component arises from the wavelength dependent refractive index that the glass retains, see section 3.4.4

An alternate schematic representation of chirp is depicted in figure 3.4. Two gratings are arranged parallel to each other, separated by a distance d. The spectral components enter the system simultaneously with no delay between them. Each spectral component is reflected at different angles according to the grating equation 3.33, resulting in the obvious path length difference between a and b. It is this temporal delaying property of spectral components within USP's that accounts for pulses being stretched and compressed.

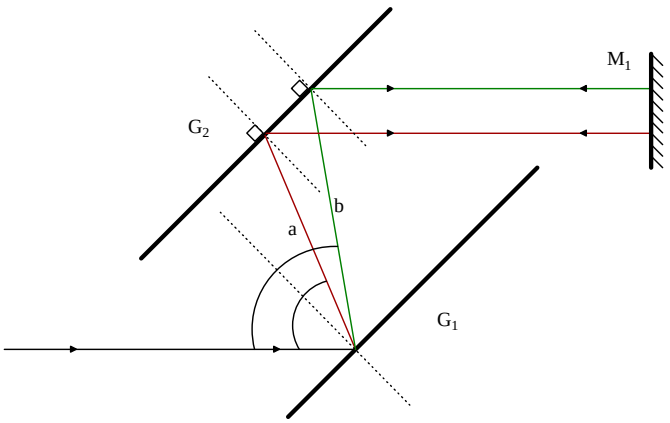

Figure 3.4: A simple illustration of how spectral components can become separated in time. Gratings G1 and G2 are parallel to each other and reflect different wavelengths along different paths. There is an obvious path length difference between a and b which results in the stretching or compressing of pulses.

Discussing a method presented in [15] due to its relevance in this work, consider a transform limited Gaussian pulse with a central frequency ω_0 and a pulse duration (FWHM) τ_{in}. Similarly to the propagation factor expansion in equation 3.10, it is common to consider the phase shift as a function of frequency ω. The phase can be expanded as a power series about the central frequency ω_0,

$$\varphi(\omega) = \varphi(\omega_0) + (\omega - \omega_0)\varphi'(\omega_0) + \frac{1}{2}(\omega - \omega_0)^2\varphi''(\omega_0) + \frac{1}{6}(\omega - \omega_0)^3\varphi'''(\omega_0) + \frac{1}{24}(\omega - \omega_0)^4\varphi''''(\omega_0) + ... \tag{3.25}$$

The electric field representing the pulse is given by,

$$E_{in}(t) = E_0\, exp\left[-\left(\frac{2ln2t^2}{\tau_{in}^2}\right) + i\omega_0 t\right]. \tag{3.26}$$

It is now possible to calculate the electric field of a pulse that has propagated through a dispersive medium by Fourier transforming E_{in} into the frequency domain and adding the dispersion components from the phase expansion in equation 3.25, before Fourier transforming back. The amount of GVD imposed on a pulse by a dispersive medium determines whether the pulse duration is stretched or compressed (in special cases, see section 3.5). This is the main phase parameter that is varied during the SHG maximization experiment. If the contribution from the GVD is considered on its own, which will be the case in the experiment, then the electric field E_{out} is given by,

$$E_{out}(t) = E_0\, exp\left[i(\omega_0 t - \varphi) - \Gamma(t - \varphi')^2\right] \tag{3.27}$$

where,

$$\Gamma = \left(\frac{\tau_{in}^2}{2ln2} + 2i\varphi''\right). \tag{3.28}$$

A linear frequency chirp is introduced and the output pulse duration is changed by the following ratio.

$$\frac{\tau_{out}}{\tau_{in}} = \sqrt{1 + \frac{\varphi_m''^2}{\tau_{in}^4}16(ln2)^2} \tag{3.29}$$

The GVD, φ_m'' in equation 3.29 caused by the material dispersion length l_m is related to the refractive index of the material $n(\lambda)$, through its second derivative with respect to wavelength about its central wavelength λ.

$$\varphi_m'' = \frac{\lambda_0^3 l_m}{2\pi c^2}\frac{d^2 n(\lambda)}{d\lambda^2} \tag{3.30}$$

3.4.4 Dispersive properties of glass

The speed of light experiences change when propagating through different media. In vacuum light propagates at a speed of 299 792 458 m/s. The velocity of light propagating through a medium is defined as,

$$\nu = \frac{c}{n} \tag{3.31}$$

where c is the speed of light in vacuum and n is the refractive index of the medium. The refractive index of vacuum is defined as $n_{vacuum} = 1$. As the

refractive indices increase when dealing with media such as air or glass, the speed of light in the medium slows in comparison to the speed of light in vacuum. The refractive index of a medium is characteristic of the material, and is not constant over all wavelengths. There is said to be a wavelength dependence in the refractive index. When dealing with glass, a popular or accurate method in determining the wavelength dependent refractive index, is given by the Sellmeier equations [16].

There are numerous forms of the Sellmeier equation and each may be rearranged and manipulated to produce the next. There are pro's and con's when dealing with different forms of the equation, but for the sake of a simple explanation consider a general form of the Sellmeier equation,

$$n^2(\lambda) = 1 + \frac{B_1\lambda^2}{\lambda^2 - C_1} + \frac{B_2\lambda^2}{\lambda^2 - C_2} + \frac{B3\lambda^2}{\lambda^2 - C_3}. \tag{3.32}$$

The constants in the above equation, known as the Sellmeier co-efficients, are measured characteristic values, and vary from glass to glass. Knowing these co-efficients, it is possible to plot the refractive index as a function of wavelength. The wavelength spectrum of the specific pulses in our lab fall within the plotted wavelength range in figure 3.5.

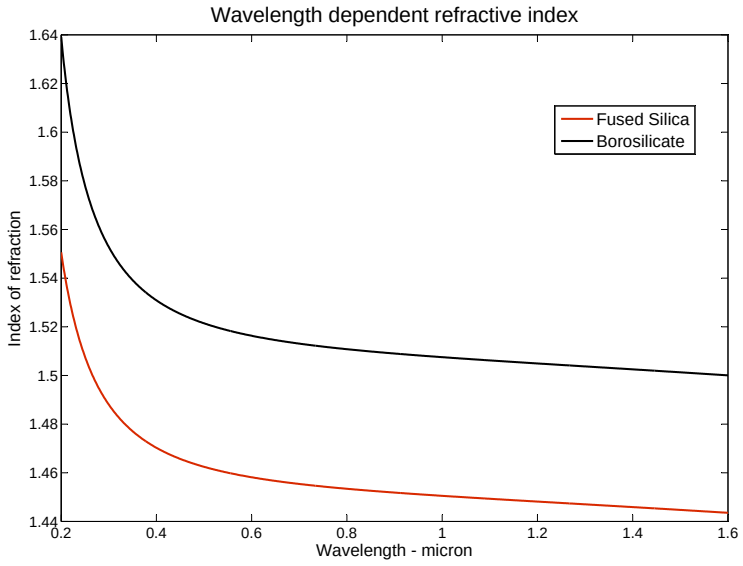

Figure 3.5: Wavelength dependent refractive indices for Borosilicate and Fused silica

To understand the dispersive effects of a particular type of glass, and to relate the amount of time stretching that occurs due to a length of dispersive medium, attention needs to paid to the second order refractive index with respect to wavelength as in equation 3.30. The equation of interest relates

	Borosilicate		Fused Silica
B_1	1.03961212	B_1	0.473115591
B_2	0.231792344	B_2	0.631038719
B_3	1.01046945	B_3	0.906404498
C_1	0.00600069867	C_1	0.0129957170
C_2	0.0200179144	C_2	0.00412809220
C_3	103.560653	C_3	98.7685322

Table 3.1: Sellmeier co-efficients for Borosilicate and fused silica

the input pulse length, the dispersion parameter and the length of dispersion medium, to an output pulse length, given by equation 3.29.

Calculating φ''_m (also known as GVD) is not a trivial process. Use of commercially available software called wxMaxima© was made, to algebraically find φ''_m. In figures 3.6 and 3.7, the first and second derivatives of the refractive index with respect to wavelength for Borosilicate and Fused silica are plotted.

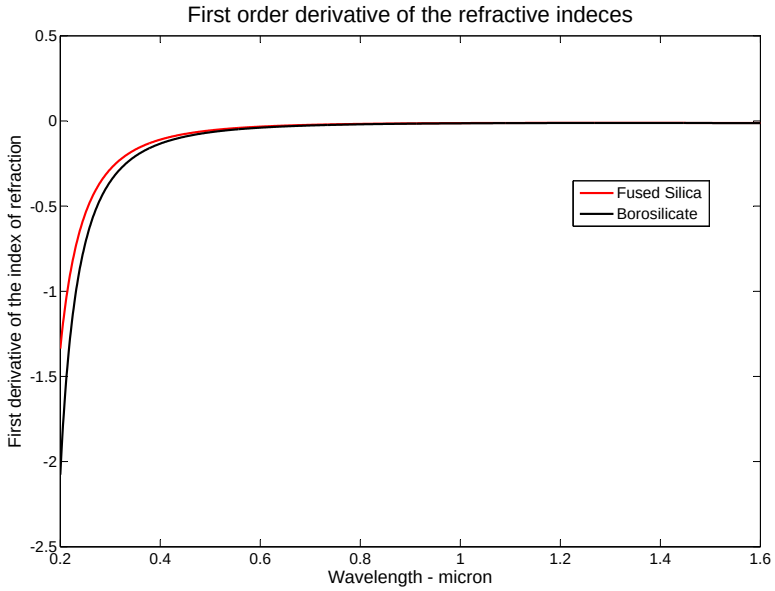

Figure 3.6: First derivative of the refractive index with respect to wavelength.

It is now possible to calculate the dispersion parameter φ''_m as a function of wavelength per unit length. This is plotted in figure 3.8. The exact values of φ''_m per unit length also known as K'' are given at 795 nm for Borosilicate and Fused silica. Equation 3.29 and the K'' values may be used to calculate the time stretching that would occur when a pulse of a certain time duration

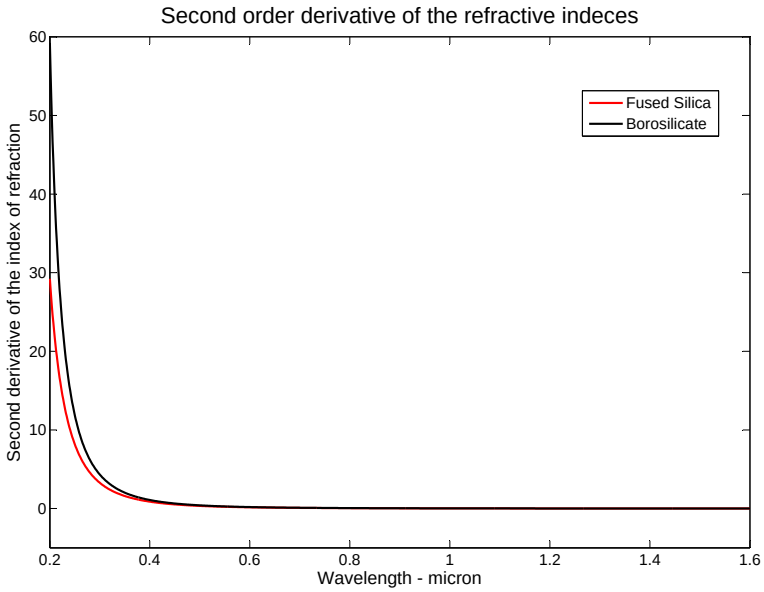

Figure 3.7: Second derivative of the refractive index with respect to wavelength.

is passed through a dispersive medium of length l_m.

Graph 3.9 shows the dispersive properties of borosilicate and fused silica. The plot is produced in the case of a 10mm dispersive rod of both glass types. The dispersive effect of such a glass rod becomes highly apparent at shorter input pulse durations. Dispersion control of USP's at durations below 40 fs becomes extremely important.

3.4.5 Dispersion compensation

Dispersion arises due to the optical components placed in the laser line as well as any medium through which the laser light propagates, including air itself. Dispersion leads to shape and time distortion and thus strict control is needed in order to maintain a pulse's original shape and duration. The main distortion is that pulses become stretched due to the introduction of group velocity dispersion in the fashion that was described in section 3.4.3. Dispersion compensation encompasses techniques which reduce this effect.

A number of designs which make use of prisms to compensate for dispersion in high powered lasers have been successfully implemented [18]. Just as dispersive media introduces a positive chirp, or frequency sweep, techniques such as prism or grating compressors introduce a negative chirp. An accurately controlled compressor is able to reduce a positively chirped input pulse to a transform limited state. The dispersion compensation technique in use on the

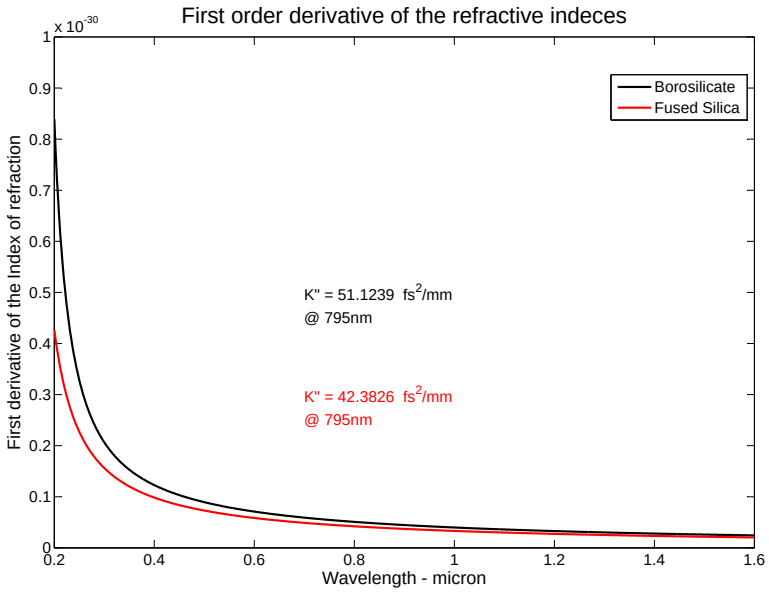

Figure 3.8: The dispersion parameter K'' (φ''_m) as a function of wavelength.

oscillator used in this work involves a pair of prisms described in [19]. Both techniques, prism pair and grating pair, are usually used in a double pass configuration as to eliminate spatial dispersion. Maintaining the balance between positive and negative dispersion is the key to retaining a specific pulse duration.

3.5 Pulse Stretching and compression

Controlling the amount of GVD that an optical systems adds or subtracts to a pulse effectively controls whether the pulse will be stretched or compressed. Consider a transform limited pulse. Such a pulse may not be compressed to a shorter pulse duration. It's transform limited pulse duration is actually defined by the manner in which it is created, see section 2.4. Most femtosecond pulses have been created in a gain medium with a large gain bandwidth and under a process known as mode-locking. This is not to say every ultra-short pulse is generated in this way, emphasis must just be put on the fact that pulse duration is defined in the pulse generating technique and cannot be reduced from its transform limited duration through introducing group velocity dispersion. Consider figure 3.10. Our input pulse is a transform limited pulse containing a continuum of frequencies over its frequency bandwidth. In the example only 5 frequency components have been included. As explained above, this input pulse cannot be compressed. To stretch this pulse, a positive amount of GVD needs to be introduced through a dispersive system. The pulse duration is thus stretched to a longer duration. Taking a chirped pulse as input, a negative amount of GVD needs to be introduced through a dispersive system to

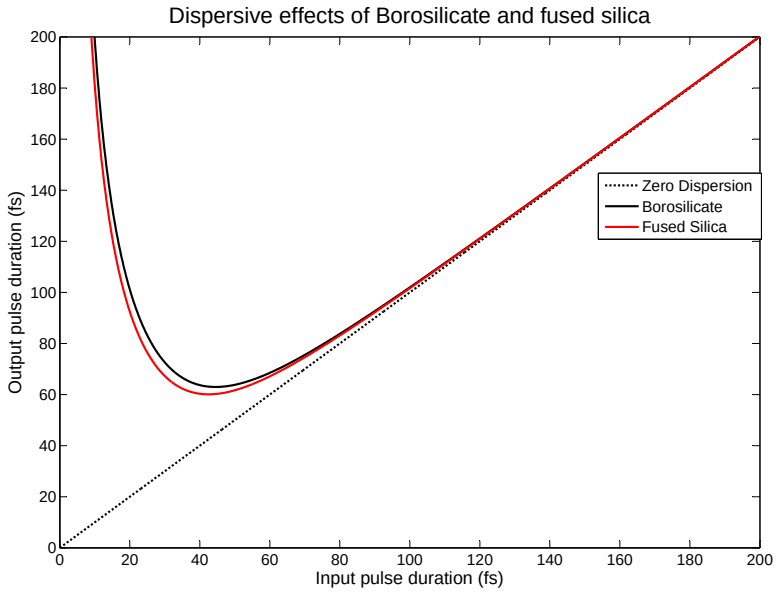

Figure 3.9: Dispersion of 10 mm borosilicate and fused silica glass rods. The stretching of pulses becomes greater at shorter input pulses.

compress the pulse. To compress the chirped pulse to its transform limited duration as in figure 3.10, requires accurate calculation.

3.6 Grating pair stretchers and compressors

One option as a dispersive element is the optical grating. These elements vary in nature from the way they are manufactured to whether they are blazed, reflective or transmissive. Gratings essentially are able to spatially separate light into its separate frequency components. It is an optical surface with finely ruled parallel lines cut into it. The distance between these parallel lines are a defining factor and are usually within the order of the wavelength of the incident light. Light that is transmitted or is reflected by a grating, is dispersed into its separate frequencies, and each frequency will follow its unique path along its unique angle. This dispersion angle is wavelength dependent. The angle is governed by the well known grating equation,

$$sin\theta_i + sin\theta_r = \frac{m\lambda}{d}. \tag{3.33}$$

It can be seen that the reflected angle is wavelength dependent where θ_i and θ_r are the incident and reflected angles respectively. Each individual frequency (represented by λ) will be reflected along a different angle. This angle also depends on the distance between each parallel line d. Finally, m represents the order of the reflection or transmission. Consider figure 3.11.

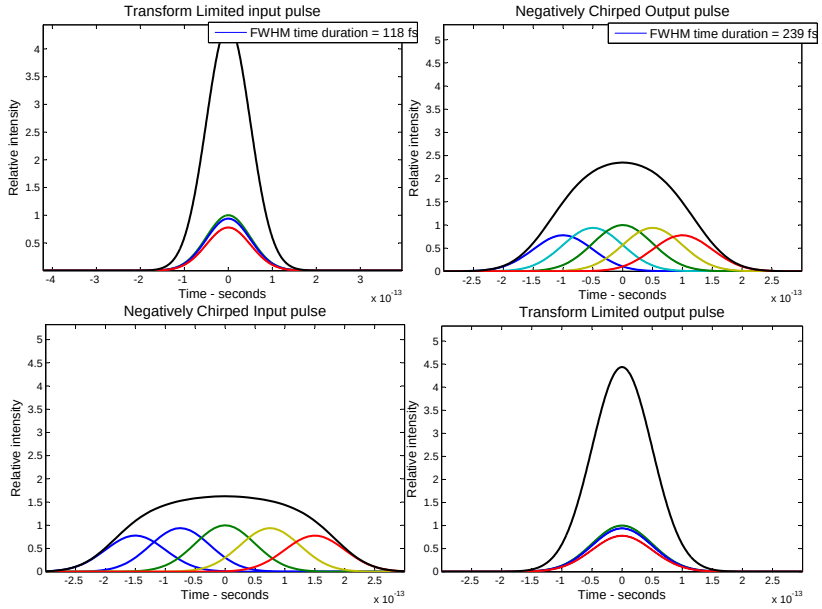

Figure 3.10: Top two graphs depict stretching a transform limited pulse whereas the bottom two graphs depict compressing a chirped pulse to its transform limited duration

The incident light (consisting of a continuum of frequencies) is reflected along a structured path. The angle between the incident light and the normal is equal to the angle between the normal and the reflected path. This is known as the zero order. An amount of light is dispersed into alternative orders numbered as the ...,-2nd,-1st,0th,1st,2nd.... orders. The percentage of light reflected along each specific order depends on the individual grating design, and so varies from grating to grating. A grating pair may be configured to stretch or compress a pulse. The amount of GVD imposed on the pulse depends on physical properties of the grating pairs themselves.

3.6.1 Compressor

Consider a pair of gratings aligned parallel to one another, depicted in figure 3.12, separated by a distance L. This configuration was first described soon after the advent of the first laser [20]. According to equation 3.33, the spectrum of a pulse will be spatially dispersed into separate orders, or for example into typically the most intense order, the -1^{st} grating order. The grating equation is now described as,

$$sin\gamma + sin\theta = \frac{2\pi c}{\omega d} \tag{3.34}$$

where γ is the incident angle with respect to the first grating normal vector, θ is the frequency dependent diffraction angle and d is the physical groove sep-

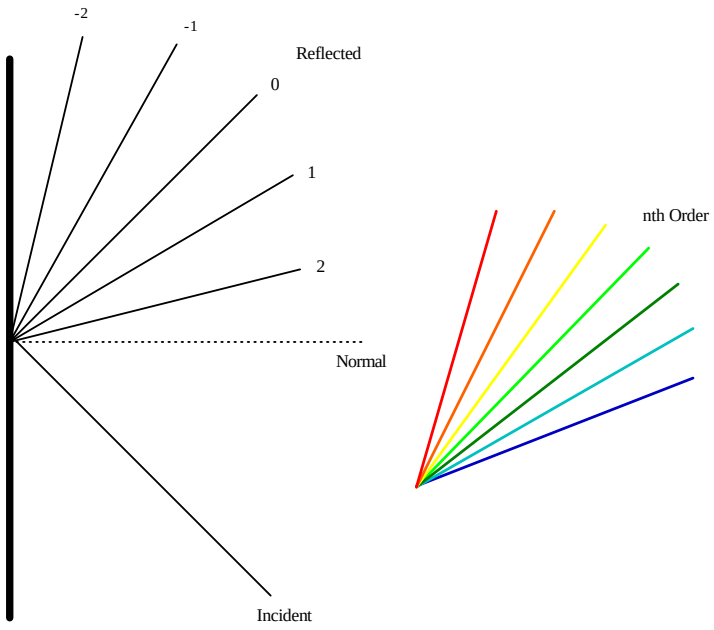

Figure 3.11: An illustration of grating dispersion. Incident light reflected along different orders, with frequency dispersion along each order except the 0th order.

aration of the gratings. The difference between the 1^{st} and -1^{st} orders would simply be a negative sign in the positive order's case. The optical path length P through the system is given by [21],

$$P = \frac{L}{cos\theta}[1 + cos(\gamma - \theta)]. \tag{3.35}$$

The total group velocity dispersion that is introduced by the grating pair is given by [21],

$$\frac{\partial^2\varphi}{\partial\omega^2} = -\frac{8\pi^2 c}{\omega^3 d^2}\frac{b}{cos^2\theta}. \tag{3.36}$$

Using equation 3.34 and the fact that $L = b/cos\theta$, equation 3.36 becomes,

$$\frac{\partial^2\varphi}{\partial\omega^2} = -\frac{8\pi^2 c}{\omega^3 d^2}\frac{L}{[1 - (\frac{\lambda}{d} - sin\gamma)^2]^{3/2}} \tag{3.37}$$

Equation 3.37 can take into account the double pass through the grating pair, simply by adding a factor of 2. The GVD introduced into the pulse is found to be negative, which means that this system acts as a compressor for input pulses which have been positively chirped. Material dispersion, such as glass dispersion in section 3.4.4 introduces a positive chirp. Mentioned is made in [21] that higher order dispersion terms become noticeable, and quadratic

dispersion is not the only contributing effect.

In summary, a grating pair, in a compressor configuration, may compress positively chirped pulses. The amount of negative GVD introduced may be controlled by the distance between gratings L, and the angle of incidence.

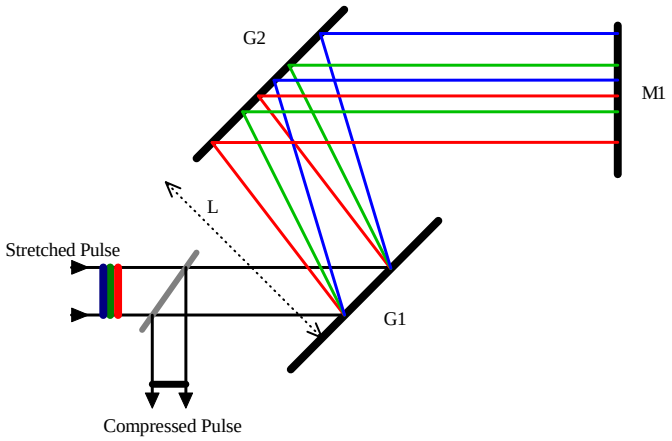

Figure 3.12: A grating pair, configured as a compressor, which imposes negative chirp on the pulse.

3.6.2 Stretcher

A stretcher configuration also consists of a grating pair, but they are not aligned parallel as in compressors. Instead the gratings are aligned as in figure 3.13. A telescope is inserted in between the gratings, which makes it possible to not only control the amount of GVD imposed by the angular dispersion of the gratings, but also the sign of the GVD by using the imaging properties as well as the angular magnification. This is the main difference between the stretcher and compressor configuration. The compressor can only impose a negative amount of GVD on the incident pulse. The stretcher can impose both negative and positive dispersion, but in the case where only compression is necessary, the compressor configuration is much simpler to implement. The total GVD of the stretcher system is now given as,

$$\frac{\partial^2 \varphi}{\partial \omega^2} = -\frac{8\pi^2 c}{\omega^3 d^2} \frac{1}{cos^3\theta} [L - 2(f_1 + f_2)] \left(\frac{f_1}{f_2}\right)^2 . \tag{3.38}$$

The stretcher may be simplified greatly by making sure the focal lengths of the lenses are equal, $f_1 = f_2$. The effective path $L - 4f$ does not have to be positive, and can be done by making the distance between the gratings and the lenses smaller than f. This makes the overall sign of the GVD positive. The positive GVD introduced is now given by,

$$\frac{\partial^2\varphi}{\partial\omega^2} = -\frac{8\pi^2 c}{\omega^3 d^2}\frac{2\Delta X}{cos^3\theta} \tag{3.39}$$

where ΔX is the distance between the position of the centre of the grating and the position of the focal point. A transform limited pulse with a considerable spectral width may now be stretched, by imposing an amount of positive GVD on the incident pulse, to longer pulse durations. The limitations on the stretching capabilities are determined by the physical setup.

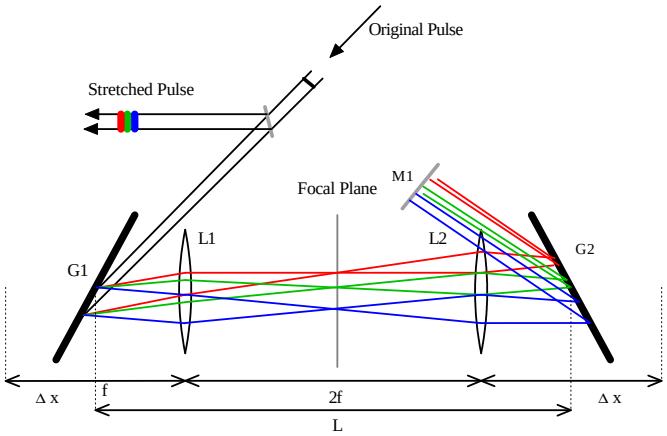

Figure 3.13: A grating pair, configured as a stretcher, which includes two focusing lenses. The setup imposes a positive chirp on the pulse.

In principle, one could match the amount and sign of GVD described by equations 3.37 and 3.39. This means that in theory that the dispersion could be matched such that a pulse may be stretched to a longer duration and then compressed back to its original pulse duration, without any residual phase deviation. Chirped pulse amplifiers use this technique by placing an amplification method between stretcher and compressor. To retain the initial pulse duration, calculation of how much GVD is introduced by the amplification method would be made. Compensation for this extra dispersion could be accommodated by stretcher and compressor configurations.

3.6.3 Pulse shaping in dispersion free $4f$ systems

Although the pulse shaping method involved in the SHG maximization experiment presented in this work makes use of an acousto-optic programmable dispersive filter, it is also possible and common to employ pulse shaping in a zero dispersion line with the use of a LCD as a spatial light modulator. The stretcher configuration described above can be setup in such a way that no dispersion is introduced. This is done by placing the gratings at the exact focal point of the focusing lenses, hence the name of a dispersion free $4f$ system. What this system does achieve, is the spatial separation of frequency

components in the Fourier plane of the setup. A spatial light modulator is then positioned there, and is then able to separately manipulate individual frequency components, which leads to pulse shaping without introducing any phase deviations from the shaping system itself.

The SLM acts as a phase and/or amplitude mask and modulates the spectral properties of the incident pulse directly in the frequency domain. An SLM consists of an $n \times n$ array of crystal cells, known as pixels. A specific voltage is supplied to each individual crystal cell and the orientation of the molecules in the crystal changes in such a way as to adjust the refractive index of the medium. As each individual pixel can be manipulated by varying applied voltages, the spatially separated frequency components will then experience a different phase shift, depending on the refractive index of the respective crystal through which it passes.

This is the basis for pulse shaping by LCD spatial light modulators. Figure 3.14 displays such a zero-dispersion 4 f system.

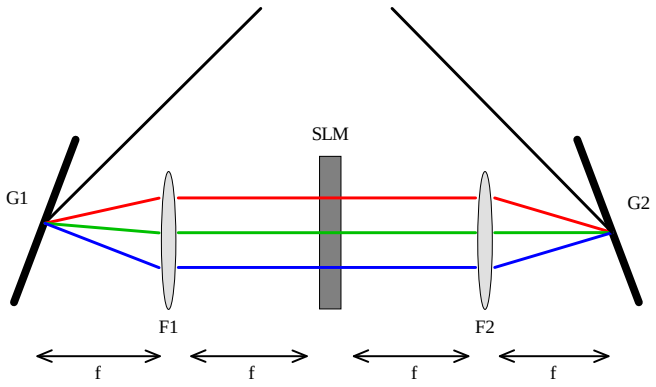

Figure 3.14: A stretching configuration with the gratings placed exactly at the focal points of the lenses. An SLM is placed in the Fourier plane and pulse shaping is now possible without any dispersion introduced by the system itself.

3.7 Pulse shaping

The description of pulse shaping presented here is based on time-invariant linear filtering and has been extensively covered [22]. Linear filtering can be thought of as an electrical signal processing method which includes frequency manipulation over a large bandwidth.

3.7.1 Linear filtering

Physical processes are commonly described mathematically in the time and frequency domain. Linear filtering can also be mathematically described in both domains. The shaping device responds to the optical input pulse as a filter would. In the time domain, the filter or response can be described as a function of time $h(t)$. This impulse response function convoluted with the optical input pulse $e_{in}(t)$ produces an optical output pulse $e_{out}(t)$ where,

$$e_{out}(t) = e_{in} * h(t) = \int dt' e_{in} h(t - t') \tag{3.40}$$

the convolution becomes a reasonably complicated integral function. Working in the time domain can thus become cumbersome and timely. It is mainly for this reason that it is preferred to shift into the frequency domain. The response of the medium or filter is now described as a frequency response $H(\omega)$. The output pulse $e_{out}(\omega)$ is now a product and not a convolution, which simplifies calculations. The relationship now looks as follows, where $e_{out}(\omega)$ is the product of $e_{in}(\omega)$ and $H(\omega)$,

$$e_{out}(\omega) = e_{in}(\omega) \times H(\omega). \tag{3.41}$$

Fourier transform analysis shows $e_{in}(t)$,$e_{out}(t)$ and $h(t)$ to be Fourier transform pairs with $e_{in}(\omega)$,$e_{out}(\omega)$ and $H(\omega)$ implying,

$$H(\omega) = \int dt\, h(t) e^{-i\omega t} \tag{3.42}$$

$$h(t) = \frac{1}{2\pi} \int d\omega\, H(\omega) e^{i\omega t}. \tag{3.43}$$

A pulse of simple or complex waveform may now be generated by creating a filter with the corresponding frequency response function. For instance, if the input pulse is known to be Gaussian in shape and a square pulse as an output in time is required, then the frequency response function needs to be calculated.

$$H(\omega) = \frac{e_{out}(\omega)}{e_{in}(\omega)} \tag{3.44}$$

If the expected output pulse is a square function in time, then the Fourier transform of a square pulse yields $e_{out}(\omega)$ and if the input pulse is a Gaussian pulse in time, then the Fourier transform of a Gaussian yields $e_{in}(\omega)$. Knowing that the Fourier transform of a square pulse is a sinc function and the Fourier transform of a Gaussian is a Gaussian, simplifies equation 3.44 to,

$$H(\omega) = \frac{FT(square)}{FT(Gaussian)}. \tag{3.45}$$

3.7.2 The DAZZLER® pulse shaping system

The DAZZLER pulse shaping system is a commercial device. It consists of three main components working in conjunction with one another. These components are namely, a radio frequency (RF) generator, a software package and

a tellurium dioxide (TeO_2) acousto-optic modulator (AOM).

The operation of this device is user controlled through the software package. Phase and amplitude modulation is easily achieved through two separate panels in the software itself. These panels allow five amplitude parameters to be varied along with four phase parameters. At any given stage the frequency response and impulse response functions are graphically displayed. These functions change when any of the phase or amplitude parameters are changed. The software package and control system also allows for the creation of multiple pulses and complex waveforms in general.

To understand exactly how the pulse shaping process takes place, an investigation into the interaction between the optical pulse and acoustic pulse needs to made. A physical discussion of the interaction follows along with a description of how an arbitrary variation of GVD can be added to an optical USP through a frequency modulated acoustic wave. These concepts have been fully covered [23, 24].

3.7.3 The acousto-optic interaction in the AOPDF.

The RF generator mentioned above is connected to a piezoelectric transducer. An acoustic wave is launched by the transducer which is excited by a specific temporal signal. The acoustic wave propagates at the speed of sound in the TeO_2 crystal, V, along the z-axis and spatially reproduces the temporal shape of the RF signal. The optical pulse propagates at the speed of light in the same crystal. As the velocity of the optical pulse is orders of magnitude greater than that of the acoustic wave, the incident optical pulses will see a fixed dielectric grating inside the device. This is also due to the ultra short pulse duration of the incident pulse. Figure 3.15 illustrates this interaction.

If the RF signal is chirped, where its instantaneous frequency is a continuously varying function of time, then the acoustic wave will reproduce this spatially, leaving its local spatial frequency chirped. It has been previously reported [25] that two optical modes can be coupled by acousto-optic interaction, specifically where certain phase matching criteria are met. In the case of only one local spatial frequency in the acoustic grating, then only one optical frequency can be diffracted at a specific position z. Figure 3.16 is an alternative illustration of the same interaction. The incident optical USP has a spectrum extending from ω_a to ω_b and is initially in mode 1. Every frequency ω travels a certain distance before it encounters a phase-matched spatial frequency in the acoustic grating. At this position $z(\omega)$, some of the energy in mode 1 is diffracted into mode 2 and then proceeds to propagate on that mode. The pulse leaving the device at mode 2 will be made of all the spectral components that have been diffracted at the various positions. In the case where the velocities of pulses propagating in mode 1 and in mode 2 differ, then each frequency will experience a different time delay.

The optical output E_{out} is a function of the optical input and of the electrical signal S(t). This is no different to the linear filtering concept introduced in section 3.7.1. The impulse response function $H(t)$ is now represented by $S(t)$.

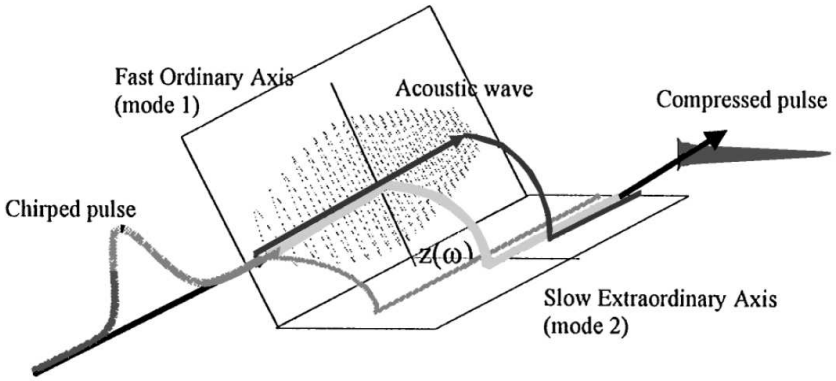

Figure 3.15: A schematic representation of the acousto-optic interaction in the AOPDF (Frederic Verluise, 2000).

E_{out} has been shown to be proportional to the convolution of the optical input and of the scaled electric signal [26].

$$E_{out}(t) \propto E_{in}(t) \otimes S(t/\alpha), \tag{3.46}$$

where α, the scaling factor

$$\alpha = \Delta n(V/c) \tag{3.47}$$

is the ratio of the speed of sound to the speed of light times the index of refraction difference between the ordinary and extraordinary waves. α is said to be the ratio of the acoustic frequency to the optical frequency. As previously discussed in section 3.7.1, when Fourier transforming into the frequency domain, relation 3.46 can be written in the frequency domain as,

$$E_{out}(\omega) \propto E_{in}(\omega) S(\alpha\,\omega). \tag{3.48}$$

In this formulation, $S(t/\alpha)$ is the impulse response function of a filter applied to the input optical pulse. Any arbitrary convolution with a temporal resolution given by the inverse of the available filter bandwidth can be achieved by generating a proper function $S(t)$. This function is then numerically computed [23] by defining its phase and amplitude. This proper function $S(t)$ is

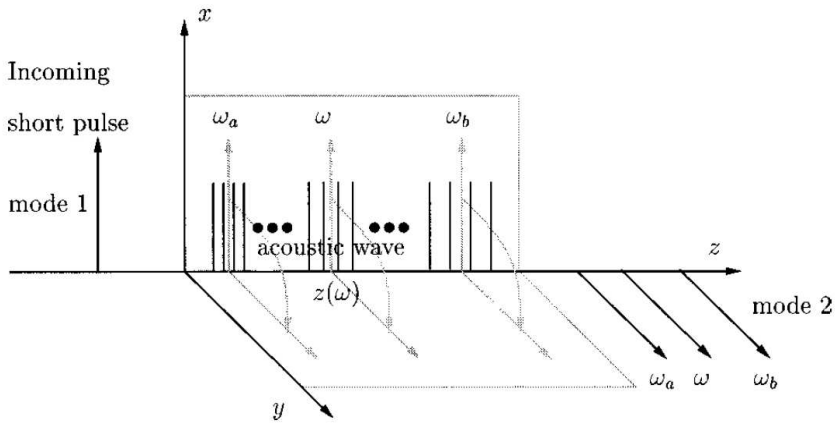

Figure 3.16: An alternative schematic representation of the acousto-optic interaction in the AOPDF (Frederic Verluise & Tournois, 2000). Notice the range of incident optical frequency components and how they are diffracted at points in the acoustic grating.

then sent to the crystal via the RF generator, and this signal is transduced onto the crystal forming an acoustic wave of time-varying instantaneous frequency which spatially represents $S(t)$.

The specifications of the Dazzler are as follows. The $Te0_2$ crystal is 2.5 cm long. The speed of sound V along the z-axis is close to 1000 m/s. The difference in index of refraction between the ordinary and extraordinary axes is found to be $\Delta n = 0.04$ and the maximum achievable group delay is 3 ps. A maximum diffraction efficiency of 30% is found. The transducer has a bandwidth larger than 20 MHz near the center frequency of $52,5$ MHz, which is the acoustic frequency that corresponds to diffraction at a wavelength of 800 nm. This 20 MHz bandwidth translates into an optical bandwidth of 150 THz near 375 THz. This leads to a associated temporal resolution of 6.7 fs.

Figure 3.17 shows the the direction of the diffracted beam with respect to the direct beam and the crystal orientation. As previously mentioned the diffraction efficiency is 30%. For an in depth discussion on an analytical expression relating the GVD at the output of the AOPDF to the input acoustic signal through the use of coupled wave theory see [23].

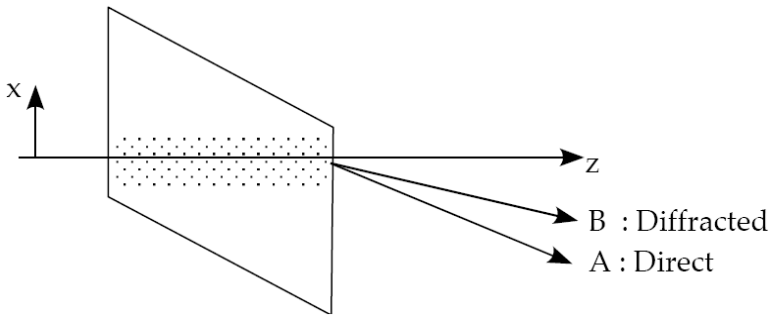

Figure 3.17: The orientation of the $Te0_2$ AOM crystal. The diffraction beam is displayed.

Chapter 4 - Optimization by pulse shaping through a Genetic Algorithm in a feedback loop

4.1 Optimization methods

Pulses are shaped for specific purposes, and these purposes may vary considerably. Whatever the purpose may be, in most cases the output from the effect of these shaped pulses can be measured. It is then possible to measure the difference in output produced from variations in the pulse shapes. This leads to output optimization through pulse shaping.

Optimization seeks to improve performance toward some common goal or output. Not only is it important that the search algorithm converges, but that the process or the performance of the algorithm is also to be taken into account. In actual fact it is not only the concern whether the search method converged to the optimum or not, but instead will compare the operation and performance to that of other techniques. Optimization is essentially about improvement, and different methods are able to compete in how efficient they are in achieving the optimum.

The three main types of optimization algorithms are known as calculus based, enumerative and random methods.

4.1.1 Calculus based methods

Calculus based methods can be subdivided into two groups, namely, indirect and direct search methods. Indirect searching looks for local extrema by solving sets of of non-linear equations. Direct searching involves looking for local extrema by following the path of the steepest gradient. Both methods starts at an initial point in the search space and then for example in the case of direct searching, takes gradients in all directions, and then moves in the direction of steepest gradient.

These search methods are limited by the fact that they are local in scope. If a function consists of numerous local extrema, it is possible that the global maximum or minimum will never be found. The other problem with this method class is that it depends on the existence of derivatives. Not all parameter spaces respect differentiation and generally lack the smoothness needed. It is common that parameter spaces are wrought with discontinuities and can be rather complex, making differentiation somewhat impossible. This search

method is confined to a limited number of problems.

4.1.2 Enumerative methods

This search group is somewhat broader than calculus based methods in terms of the plausible parameter spaces it may search. Including both finite and discretized infinite search spaces [27], the algorithm looks at objective function values at every point in the space, one at a time.

This search method is relatively simple and in fact is very effective but only where the search space is practically small. The issue is that many practical search spaces are simply too large. In large parameter spaces, it simply becomes inefficient to search the space point by point.

4.1.3 Random methods

This class of search method has recognized and improved on the shortcomings of the above two classes. Separation must be made between methods which are purely random methods such as random walks, and randomized techniques which make use of choice as a tool at specific junctions in their search method.

A purely random method may imply directionless search and will tend to perform similarly to enumerative and calculus based methods. Randomized techniques on the other hand, may use random choice as a highly exploitative search tool, that can turn out to be anything but directionless.

4.2 The Genetic Algorithm

4.2.1 Introduction

Evolutionary algorithms such as genetic algorithms use directed randomness as a searching technique. They do not perform in the same way that techniques such as random walks do, as these type of methods become highly inefficient as the sample space grows. Instead the genetic algorithm uses randomness as a tool to guide a highly exploitative search through a sample space [27]. The genetic algorithm is based on systems found in nature. It is mainly based on the idea of natural selection or survival of the fittest. This method encompasses the main idea behind evolution, including reproduction, crossover and mutation. To summarize, genetic algorithms differ from traditional search methods in the following ways.

1. The parameter set is first coded and genetic algorithms work with these coded parameter sets, not the parameter set itself.

2. At the start of the search, a population of points is chosen, and search does not start at a single point.

3. Objective functions are used instead of derivatives.

4. Probabilistic transition rules are used instead of deterministic ones.

Most optimization methods move from a single point to the next single point using a set of transition rules, in search of extrema. This often gives rise to false peaks in a multi-peak system. GA's on the other hand start off from a large set of points simultaneously, searching for peaks and dips in parallel. The next section explains the inner workings of the most simple genetic algorithm.

4.2.2 A Genetic algorithm in its simplest case

Consider the problem of searching a sample or parameter space, looking for a parameter value, that when tried in an objective function, produces the maximum value. For simplicity consider a simple quadratic function, which is plotted in figure 4.1. It will be easy to look for a maximum value from a sample set.

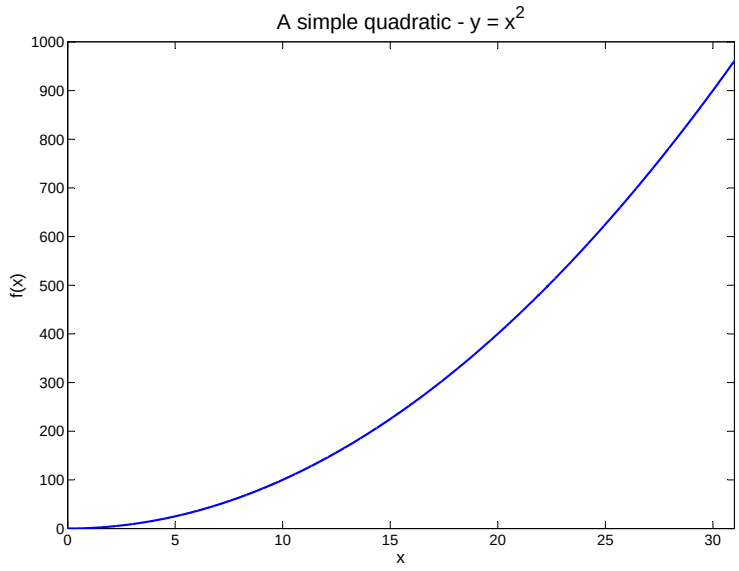

Figure 4.1: Simple function : $y = x^2$. Notice the parameter space [0...31].

The problem is defined as, which entry within the sample space would produce the maximum function value. An investigation into how a genetic algorithm solves this problem step by step will be followed. A genetic algorithm will then be used to solve complex function problems.

The GA solution to this problem involves numerous junctions where randomness is introduced. As part of the initialization process, a GA starts with a population of initial strings. The strings form the encoded search space. The

form of a string may vary, but in this solution, binary strings are used for simplicity. As an example consider a sample space that runs from [0...31]. Each entry can be mapped into a binary representation. For instance, 0 is represented as 0 in binary and 31 is represented as 11111. This means the minimum string length should be 5, ranging from [00000 ... 11111]. Thus the length of a string entry is actually defined by the size of the parameter space itself. Every string must be able to describe the largest or smallest parameter in the space. The size of the population, or the number of strings can be specified in the initialization process. The outcome or operation of the method may vary greatly depending on what population size is initially selected.

For example, a string population of 4 strings is chosen. The value of these strings will be chosen by a (random) coin flip where $heads = 1$ and $tails = 0$. Each string would need 5 coin flips and an initial population of 4 strings would thus need 20 coin flips in total. A binary string representation of the values in the sample space has been constructed.

String representation	Actual parameter
00110	6
11101	29
10001	17
11010	26

Table 4.1: Binary string representation of four random entries which are part of the parameter space.

From this initial population, successive generations will be developed through the help of three main operators.

4.2.3 Reproduction

This process does exactly what the name suggests. According to some selection probability based on the function value, or more specifically, its fitness value, the strings are copied to the base of the new generation. Strings with strong fitness values are likely to be selected for the new generation, some more than once, and those with low fitness values are likely not to be selected at all. This operator is similar to the natural selection theory of evolution.

Once a string has been selected it is replicated exactly and placed in the new, updated population. This is not the final population, but provides the bases population for the following two operators.

4.2.4 Crossover

Strings in the basis population are selected at random to 'mate' and produce offspring. This is represented by string crossover with each other. In essence this involves copying parts of each string and replacing existing parts. A second random number between 1 and $l - 1$ is selected, where l is the length of the string. This number effectively marks a position in each string where crossover

will take place. All characters beyond this cross site are swapped with the partner string.

In the example above, our strings are five characters in length. For instance a random number between 1 and $l-1$, where $l=5$, is found to be 3. This means the cross site between two example strings are found to be as follows,

$$S_1 = 1\,0\,1\,|\,1\,0$$
$$S_2 = 0\,1\,1\,|\,0\,1$$

The characters after the cross site are swapped from partner to partner. In terms of implementing this computationally, it is trivial, as the major role is simply a string copy, and is generally an inherent function in higher level programming languages. Once crossed these strings are then placed in a temporary population, until they are joined by each crossed over string. Each string is selected once only to be mated with another string, so to simplify the problem, ensure that the initial population is an even amount. The new pair looks as follows.

$$S_1 = 1\,0\,1\;\;0\,1$$
$$S_2 = 0\,1\,1\;\;1\,0$$

4.2.5 Mutation

A generation of strings has been constructed, which have been chosen on their fitness values and have subsequently been crossed with each other. The third operator is mutation and involves randomly changing a value in a string. In the case of binary strings, this involves flipping a one to a zero or vice versa. This mutation probability is generally a very low probability and generations may pass without a single mutation. The chance of mutation actually taking place will obviously increase with an increase in string population and string length.

The descriptions above only explain the physical processes behind each operator and do not explain the significant role they play in producing a robust effective search algorithm. The reasoning behind each operator (and the randomness introduced) will be explained and become apparent in the rest of this section.

Returning to the initial population of four strings and the objective function, the necessary operators to produce the next generation are available. In applying the reproduction operator four strings need to selected for the next generation. The population size remains the same in this example although growing populations sizes are not uncommon. In order to make an informed selection on the strings for the following generation a few calculations need to be completed.

The initial strings were constructed from a coin flipping process, and before their fitness can be tested their corresponding sample space value need to

No.	String	X-value	Fitness : $y = (x^2)$	P_{select} : $\frac{f_i}{\sum_{i=1}^{n} f_i}$	Count $\frac{f_i}{\overline{f}}$
1	00110	6	36	0.02	0.08
2	11101	29	841	0.46	1.82
3	10001	17	289	0.16	0.63
4	11010	26	676	0.36	1.46

Table 4.2: Fitness values of the initial population.

be known. A simple binary to decimal conversion needs to take place. Each string now has its 'x' value, and its fitness may be tested. Each of the following strings is displayed in table 4.2 along with the corresponding x-values and fitness values.

The fitness of a specific string is determined by the objective function. Knowing the fitness values with respect to the each other, and thus the sum of the fitness values of an entire generation, it is possible to assign selection probabilities to each string. The selection probability is simply defined as,

$$P_{select} = \frac{f_i}{\sum_{i=1}^{n} f_i} \tag{4.1}$$

where f_i is the fitness value of a specific string. The expected count is defined as,

$$C_{ex} = \frac{f_i}{\overline{f}} \tag{4.2}$$

where $\overline{f}$ is the average string fitness. A simple way to implement the selection process is to construct a biased roulette wheel where each strings selection probability defines the size of the strings segment on the roulette wheel, see figure 4.2. The strings with higher selection probabilities are set larger slots, thus allowing a larger possibility of selection at every 'spin of the wheel'. In order to select four strings for the next generation, four spins are needed. Strings with low selection probabilities, or more correctly, with low fitness values have a lower chance for being selected for the following generation. Four spins of the roulette wheel are made and the outcome is listed in the count column of table 4.2. It can be seen that the fittest string was selected twice and the least fit string was not selected at all.

A new pool of four strings has been selected, and mates from this pool are randomly paired with each other. The next operator, crossover, takes place. As previously described, a cross site is randomly selected, and the characters after the cross site are swapped between pairs. This ends the crossover stage and leaves only mutation as the final operator. Mutation takes place by moving through every bit in the string population, flipping characters according to a mutation probability. The probability is small, say for example, $P_{mutation} = 0.002$ which means that if twenty bits are present in the generation,

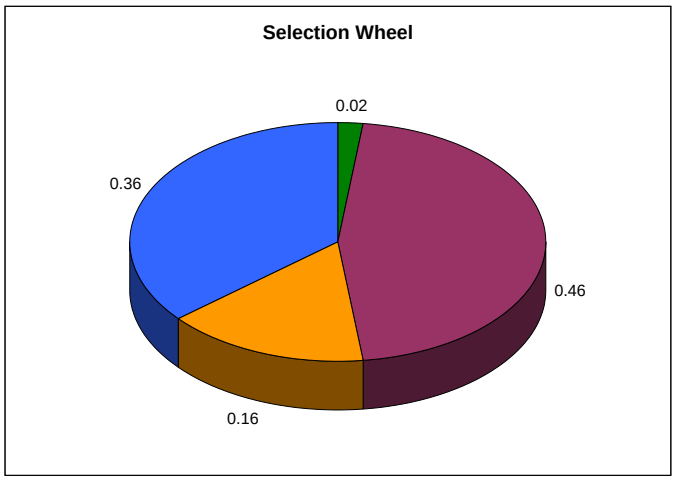

Figure 4.2: A biased roulette wheel of string fitness values.

then the number of bits expected to mutate is $20 \times 0.002 = 0.04$. This implies that in one generation the chance of a single bit being flipped is relatively low. Chances of mutation increase as the generations progress further and further, along with larger numbers of strings present in the generation. After the three operators have completed their tasks, a new generation has formed, which can be seen in table 4.3.

Selected string	Mating pool and cross site	New Population	x	$y(x)$
4	110|10	11001	25	625
2	111|01	11110	30	900
2	11|101	11001	25	625
3	10|001	10101	21	441

Table 4.3: The new generation, and the improved fitness values

The new average fitness value is 647.45. Comparing this to the previous generations average fitness value of 460.5, an increase of 186.95 after one generation is observed. As the generations progress, this average fitness value is what needs to be monitored to determine if the search algorithm reaches the goal, and in what manner it does. This is a generation slice, or the initial generation slice and after further iterations, the evolutionary track may be analysed. In the following section results from more complicated functions with vast string population and large sample space are interpreted.

4.3 Genetic algorithm simulations

4.3.1 Simple quadratic

The objective function was plotted in figure 4.1 which is simply the positive side of $y = x^2$. From our initial example of a generation slice, a simulation of the fixed population of four strings over twenty generations was produced. The result is plotted in figure 4.3. A second simulation is plotted, where the population size has been doubled to 8 strings. The average sample space x-value of the entire population is plotted against the generation progression. The optimum, $x_{optimum} = 31$ (theoretical) is plotted as to confirm whether the simulation reaches its goal. The difference between simulations is noticeable where the second simulation reaches the optimum after only 7 generations and the first simulation takes 18 generations to get there.

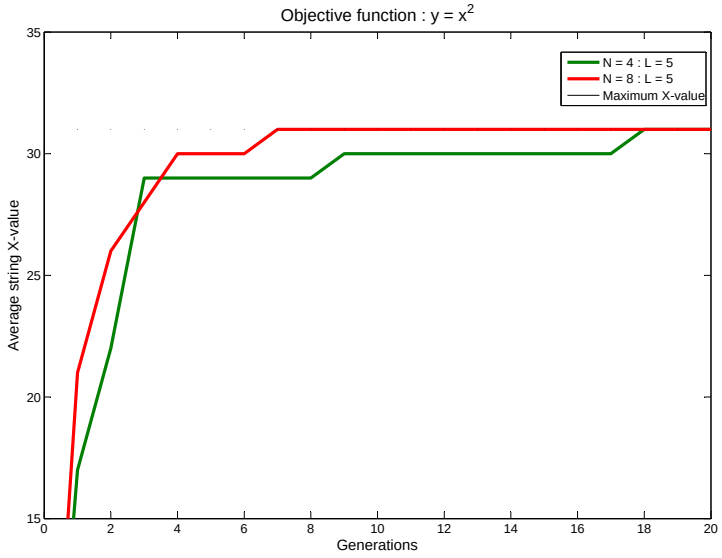

Figure 4.3: Optimization results for $y = x^2$ with population sizes of 4 and 8 strings and a string length of 5. The optimum result is plotted at an x-value of 31.

A more appropriate simulation is plotted in figure 4.4. In this simulation the string length has been increased to 10 characters, which means the maximum value of x, or the value within the sample space which produces the greatest fitness, according to the the objective function $y = x^2$ is now $x_{optimum} = 1023$. A string population of $N = 20$ and $N = 100$ are plotted over 200 generations.

The simulation with a higher string population seems to strive towards the optimum in a stable fashion where as the smaller population approaches the optimum erratically. This increase in smoothness obviously has to do with

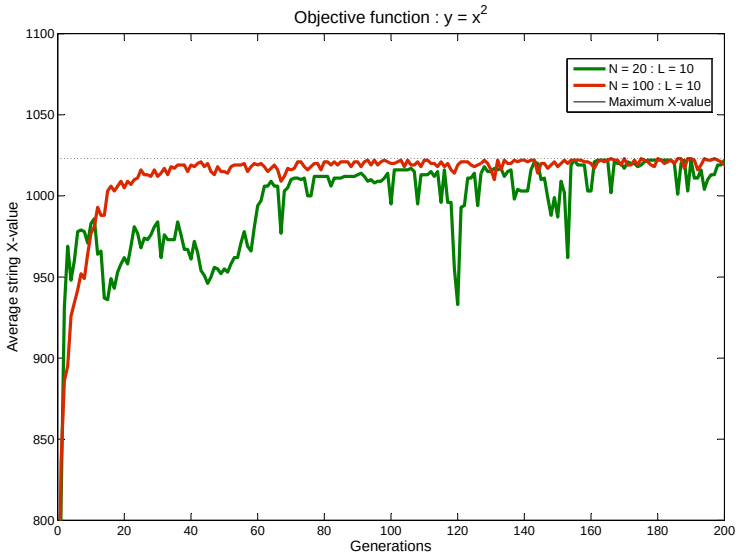

Figure 4.4: Optimization results for $y = x^2$ with population sizes of 20 and 100 strings and a string length of 10. The optimum x-value is displayed as 1023.

the average smoothing out over more strings. A smaller population size is less robust and the populations average fitness is sensitive in comparison. A weak string being selected has a dramatic effect as well the mutation operator causing large dips and spikes in average fitness. A string has a (significant) bit switched at random. This mutation causes the fitness value of the string to drop, and thus has an affect on the entire populations average fitness (if the population is small). This mutated string, can be handled in two ways. Either the process of natural selection excludes the string from the next population, or the string is mated with a highly fit string to produce a fitter string. The time taken to reach the optimum is perhaps the most important factor.

The code from this specific simulation is given in appendix B.

4.3.2 Gaussian

A more realistic test on the performance of the algorithm would be a multipeaked objective function. Figure 4.5 depicts a combination of Gaussian functions, described as,

$$f(x) = Ae^{-((t-t_0)^2)/10000)} + Be^{-((t-t_1)^2)/10000)} \tag{4.3}$$

In order to test the 'locality' issue, a GA is used to search for the x-value within the sample space which produces the maximum fitness value. This is obviously the second and highest peak in figure 4.5. Figure 4.6a, shows the

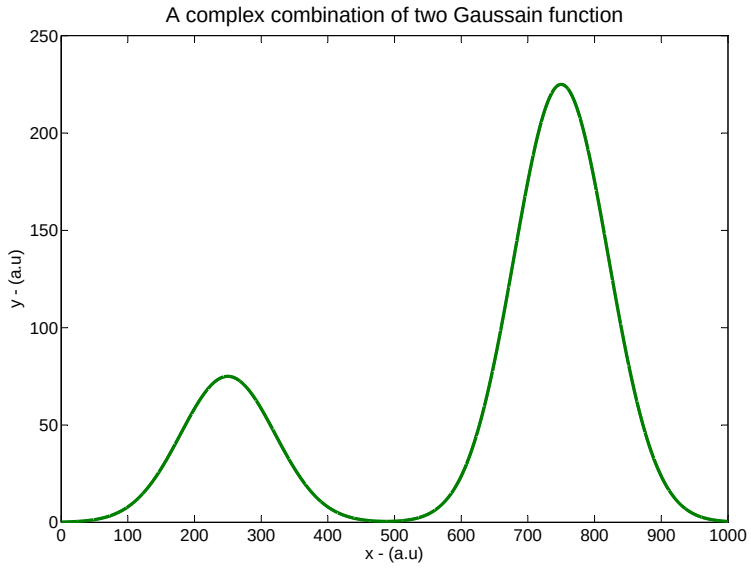

Figure 4.5: A combination of two Gaussian functions to produce a double peak.

search for the optimum along four different string populations. It can be seen, that the strings strive to the highest fitness value, and thus the corresponding x-value at the highest peak is produced as the result of the simulation. The population with the highest number of strings reaches the optimum fastest. The lowest string population reaches the optimum relatively fast but can be seen to change dramatically under mutation. The differences seem only subtle and this has to do with the long string length. Each of these simulations started with an initial population with an evenly spread distribution of x-values.

To test a severe case, the x-values of the initial string population were forced to have values below the start of the first peak. If the GA can move past this small local peak and find the maximum peak then it has turned out to be somewhat robust. Three cases were produced and the simulation progression is plotted in figure 4.6b. The initial string populations x-value were distributed between 127 - 0, 63 - 0 and 31 - 0. This can be seen by the average x-values of each population in the first generation. Emphasis is not to be placed on the differences in initial distributions, as the main reason three simulations were run was to illustrate the non-deterministic nature of the progression of this genetic algorithm. The optimum was found in each case, thus showing that the algorithm moved past the initial local peak and progressed to show the second peak as the maximum. This search process, as previously explained, contains a large random influence, and it is impossible to produce the same simulation over again. This makes it difficult to analyze. In the second case, the initial population was restricted between 0 and 64. It seems as though the average x-value stabilizes around the value of 250, which corresponds to the x-value of

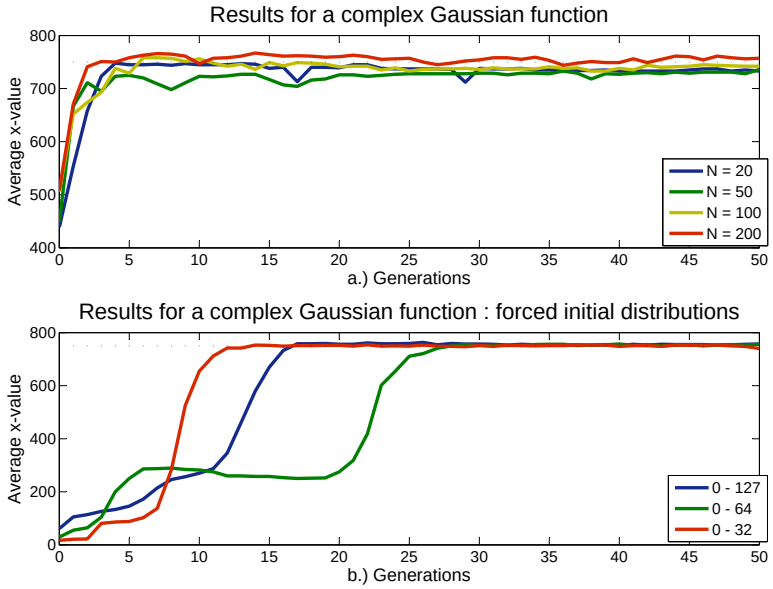

Figure 4.6: a.) The GA strives to the optimum fitness value and thus, the corresponding x-value. b.) The same algorithm parameters are used, where the initial distribution is varied.

the first peak. After a few successive generations the average x-value starts to climb and then strives towards the second, higher peak.

4.3.3 The Sinc function

A more complicated function, such as the sinc function, which consists of a large number of peaks and local extrema, with a global maximum positioned in the middle, seen in figure 4.7 was similarly tested. The specific sinc function was described by,

$$f(x) = 10 \times sin(0.1x - 500)/(x - 500). \tag{4.4}$$

In the initial population, as previously mentioned, the strings are assigned random values between 0 and 1000. For larger numbers of strings, the closer the average x-value of the initial population will have a value of $x = 500$, depending on the random number generator. Seeing that the global maximum is situated at $x = 500$, the initial population was forced to have random x-values between 0 and 127 assigned to them instead. In this way, it can be seen that the algorithm works past the local extrema and then finds the global maximum.

In figure 4.8 it can be seen that larger string populations find the maximum in the shortest number of generations. The smallest population seems to

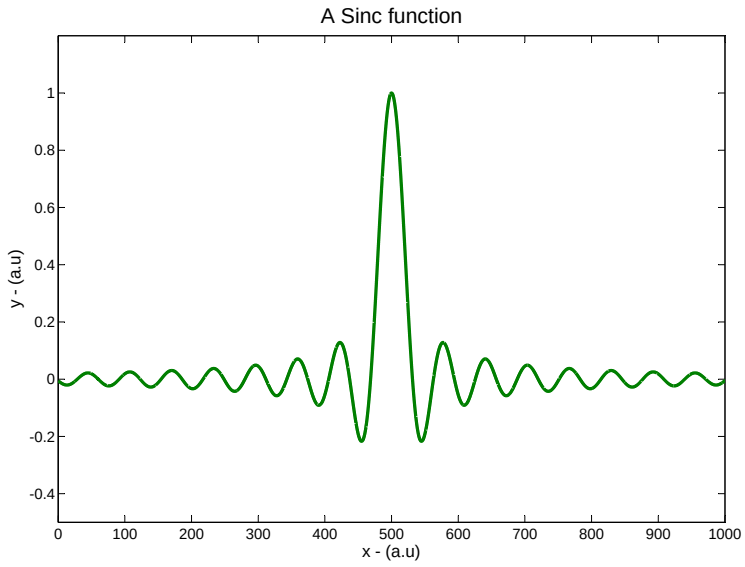

Figure 4.7: A sinc function, consisting of local extrema and a global maximum situated at $x = 500$.

stumble over local extrema and is subject to considerable changes under the mutation operation.

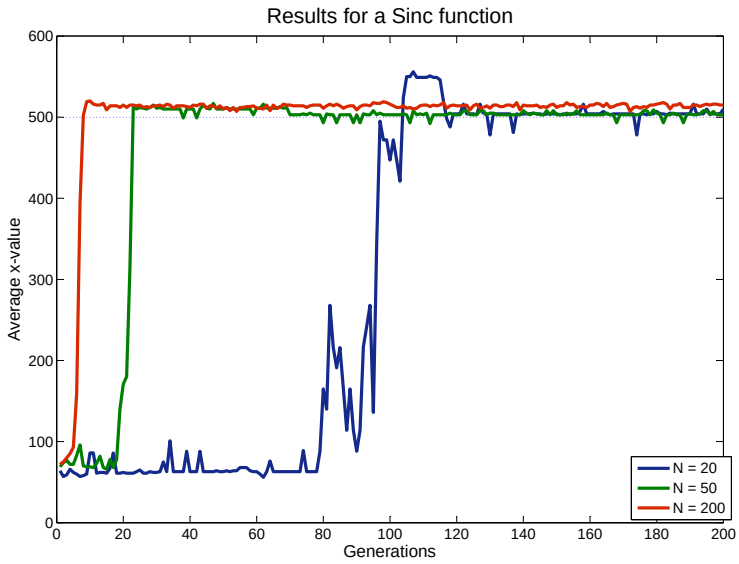

Figure 4.8: Results from the simulation showing that the maximum fitness function value is found at $x = 500$.

Chapter 5 - Experimental setup

5.1 Introduction

The aim of this experiment is to ensure the shortest possible pulse duration as an outcome. SHG signal is used as a proxy for pulse duration. The SHG signal is related to the input pulse intensity. The greater the pulse intensity the greater the SHG signal will be. The peak intensity of USP's can be manipulated by changing the pulse duration.

It is possible to manipulate the pulse duration of a pulse of a certain energy, in the manner that has been discussed throughout this work. Although the laser pulse as an energy packet remains constant, the peak intensity may rise and fall with the respective compressing and stretching cases. An USP's highest peak intensity is found in its transform limited state. To search for the highest SHG signal implies searching for the highest peak intensity, which in turn implies the search for the shortest possible time duration.

5.2 Experimental configuration

The SHG maximization experiment involved the following apparatus.

1. Femtosecond Oscillator
2. DAZZLER pulse shaping system
3. Amplifier
4. Non-linear BBO crystal
5. Photo-detector
6. Oscilloscope
7. Personal computer with genetic algorithm and feedback control.

Figure 5.1 depicts the experimental setup. Pulses of 120 fs in duration are passed onto the pulse shaping system. The optical pulse becomes convoluted in time with the acoustic pulse present in the DAZZLER. This interaction shapes the pulse which is then passed into the amplifier. A non-linear BBO crystal is placed in the optical line, and the pulses pass through, causing SHG. This signal is collected by the slow photo-detector and subsequently relayed to an

oscilloscope which reads this value and passes a voltage value to a computer. This voltage is proportional to the integrated SHG signal and hence proportional to the SHG yield converted from the fundamental.

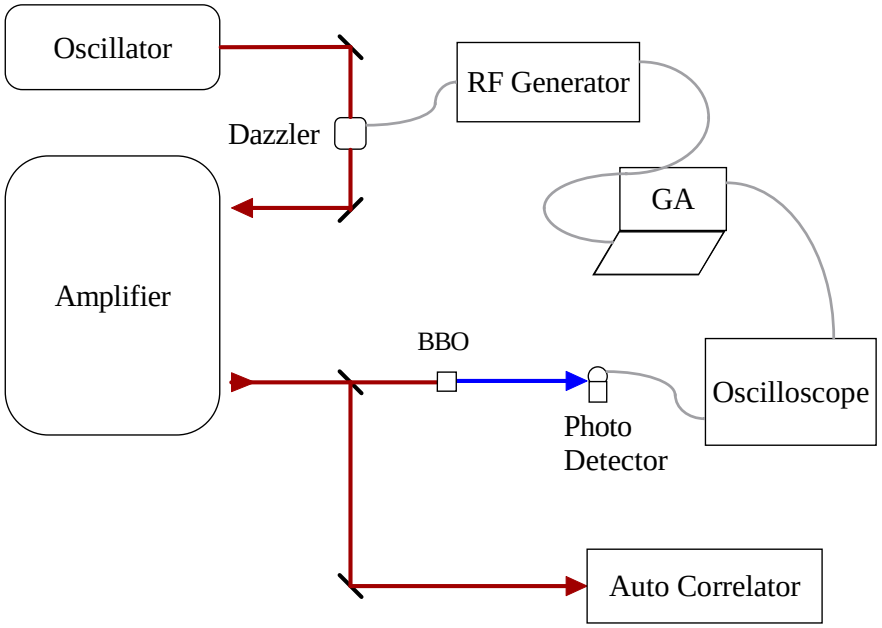

Figure 5.1: A schematic diagram of the second harmonic optimization experiment.

This voltage value is used as input into the genetic algorithm which then in turn produces output which affects the shape of the next pulse in consideration. This completes the feedback loop.

5.2.1 The femtosecond laser oscillator characterization

Once it has been confirmed that the oscillator is mode locked and is producing pulses, it is simple enough to measure the average output power. This is done by recording pulse trains with an oscilloscope and ensuring that the pulses are being produced at the required repetition rate. A power meter is then placed in the optical line and a reading of, average (continuous) power is taken, as an example,

$$P_{ave} = 0,691 \text{ W} = 691 \text{ mW} \tag{5.1}$$

This reading was taken at a repetition rate of 75,75 MHz. Thus implying each pulse to have an energy of, energy per pulse,E_{pp},

$$E_{pp} = \frac{0,691 \text{ W}}{75,75 \text{ MHz}} = 9,122 \times 10^{-9} J \approx 9,1 \text{nJ} \tag{5.2}$$

A major part of pulse characterization is spectral analysis. This provides information on frequency bandwidth, center frequency as well as phase information. The spectrum was recorded by an Ocean Optics $USB4000$ spectrometer. The following spectrum was recorded when the oscillator was operating in a mode locked state at 75,75 MHz.

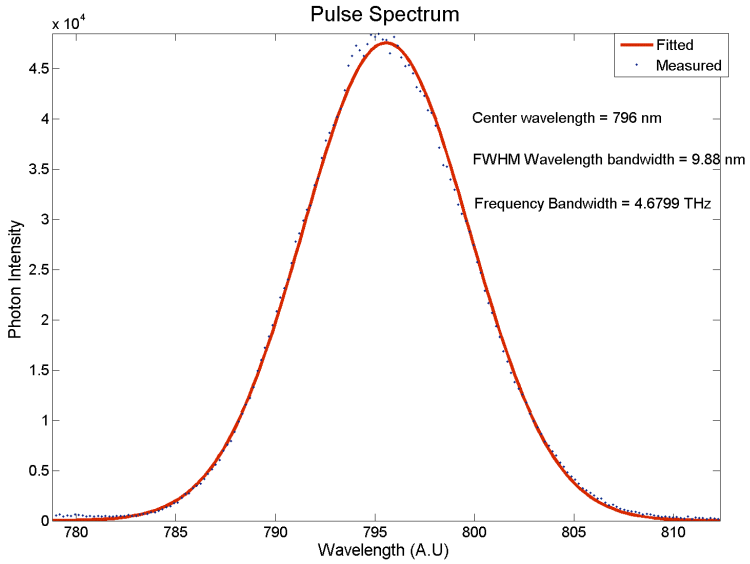

Figure 5.2: A Gaussian fit of the spectrum at 75,75 MHz:

The center wavelength can be measured by reading the corresponding wavelength of the maximum data point, or more accurately by taking the mean of the fit. The wavelength bandwidth can be found by simply taking the FWHM of the data fit. The frequency bandwidth is calculated in the following manner.

For a single frequency, it is related to a single wavelength by,

$$\nu = \frac{c}{\lambda}. \tag{5.3}$$

The frequency bandwidth can be written as,

$$d\nu = |\frac{-c}{\lambda^2}|d\lambda. \tag{5.4}$$

Now that the frequency bandwidth has been calculated, the implied pulse duration may be calculated (based on a few assumptions). The assumption that the pulse is without chirp and that the pulse is of Gaussian form in the time domain must be made. To come up with a theoretical value for the pulse

duration (assuming there is no chirp present in the pulse) the time-bandwidth product discussed in section 3.3.1 can be used. This provides us with an estimate of the pulse duration. This is only only a theoretical value based on measured spectral data. Measuring the actual pulse duration requires an optical method. Figure 5.2 is a fit to spectral data taken at a repetition rate of 75,75 MHz.

An autocorrelation technique was used to measure the pulse duration and is described in section 5.3. Once an autocorrelation trace has been captured, a value for the auto-correlation pulse duration, Δt_A, can be produced. For a Gaussian pulse shape specifically, the actual pulse duration maybe related to the autocorrelation pulse duration by,

$$\Delta t = \frac{1}{\sqrt{2}} \Delta t_A. \tag{5.5}$$

The autocorrelation trace FWHM is calculated from the measured FWHM value in figure 5.3. In order to calculate Δt_A the scanning speed and the scanning time which is the FWHM of the trace, must be known. It is now possible to calculate the correlation length,

$$d = AC_{fwhm} \times scanspeed. \tag{5.6}$$

Once this distance is known, the time taken to travel this distance is simply the distance covered divided by the speed of light in air. As the pulse actually travels up and down the arm depicted in figure 5.8, a factor of 2 forms part of equation 5.7,

$$\Delta t_A = \frac{2 \times d}{c} \tag{5.7}$$

where c is the speed of light in air. Equation 5.5 may be used to calculate the actual pulse duration as seen in figures 5.3 and 5.4. Figure 5.4 is another auto correlation trace but at a different scanning speed. There are slight differences in the corresponding pulse durations which may be attributed to the fitting procedure as well as inaccuracies in the auto-correlation technique.

Each Gaussian fit was made by a least squares approximation explained in appendix A. The laser oscillator used in this work may be characterized by the attributes displayed in table 5.1.

5.2.2 The DAZZLER pulse shaping system

The shaping system is fully described in section 3.7.2. As a component to this experiment, its main function is to receive a GVD setting as an input and consequently update the acoustic waveform. This step is part of the feedback loop and the GVD values it receives are output values from the genetic algorithm.

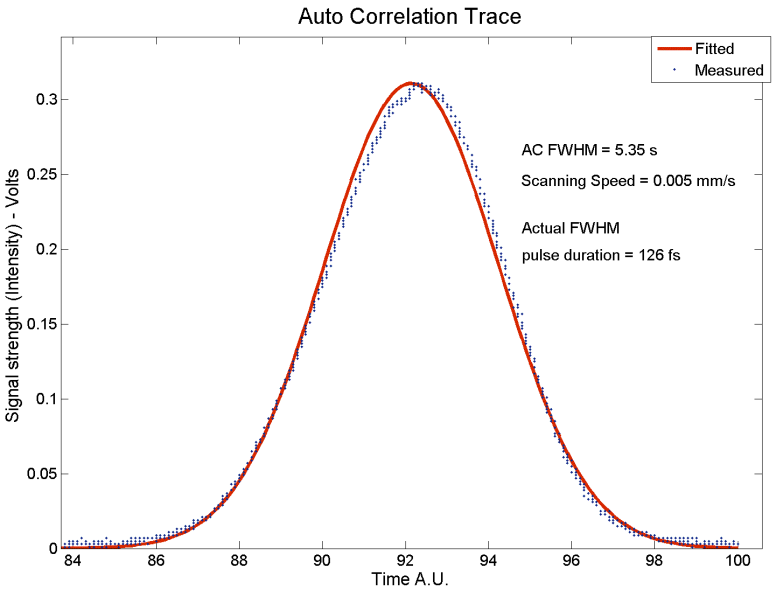

Figure 5.3: An auto-correlation trace with a Gaussian fit.

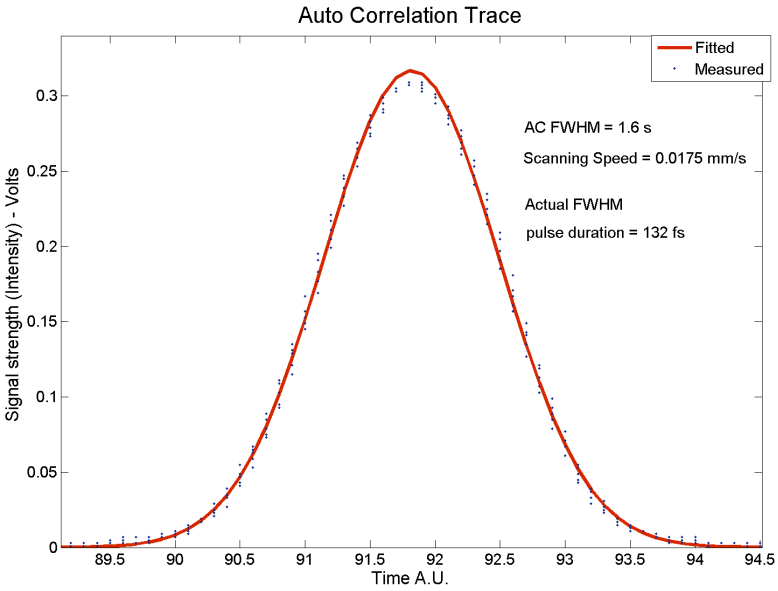

Figure 5.4: An auto-correlation trace with a Gaussian fit.

Repition Rate	75,75 MHz
Average Power	691 mW
Energy per pulse	9,1 nJ
Peak power	79,1 kW
Center wavelength	795 nm
Center frequency	377,3 THz
Wavelength bandwidth	9.88 nm
Frequency bandwidth	4,6799 THz
Pulse duration	125 fs

Table 5.1: The femtosecond laser oscillator characteristics used in this work.

A small acousto-optic crystal is all that is placed in the optical line and can be seen in figure 5.5. The new waveforms are sent via radio-frequency and then transduced onto the crystal.

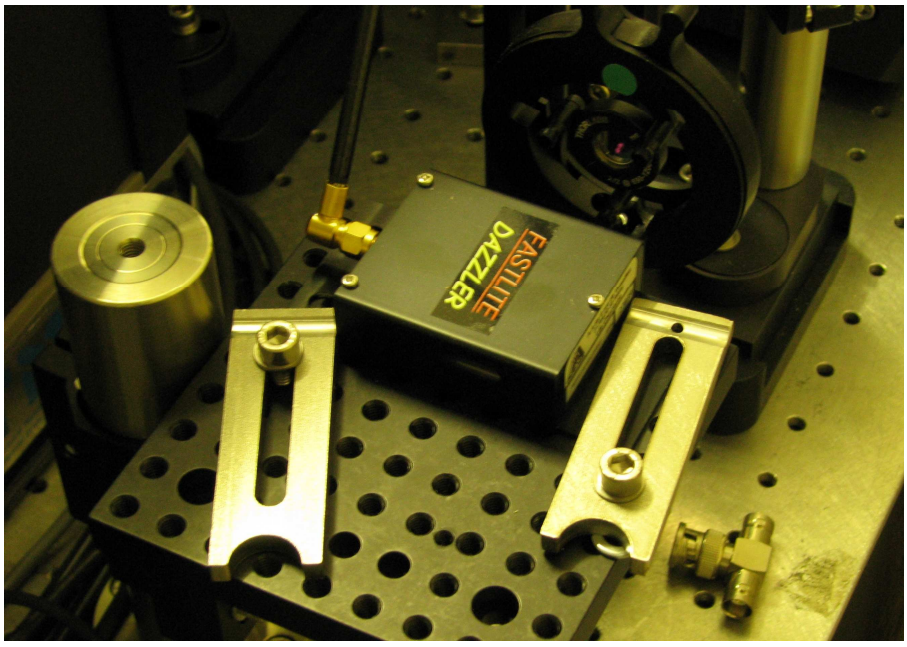

Figure 5.5: The DAZZLER shaping device placed in the optical line. The AOM crystal resides within the black box shown. The wire connecting the RF generator and the transducer can be seen on the left of the box.

5.2.3 The amplifier

This is a regenerative multipass amplifier. The oscillator produces femtosecond laser pulses at a repetition rate of 76 MHz. The pulses are shaped at a

repetition rate of 1 KHz and it is at this rate the amplifier operates.

This amplifier is used in conjunction with a technique known as Chirped Pulse Amplification (CPA). The pulses from the oscillator are stretched reducing the peak intensity. This pulse is then safely amplified and the re-compressed to from a high intensity USP. Figure 5.6 illustrates this technique.

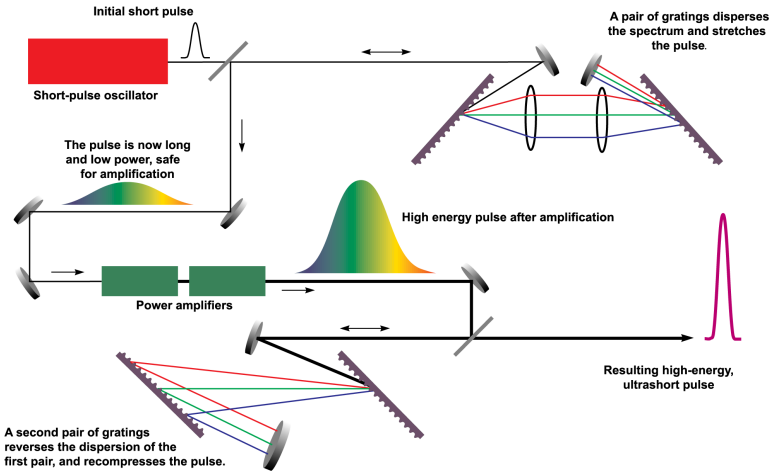

Figure 5.6: A schematic representation of chirped pulse amplification.

5.2.4 The Non-linear BBO crystal and the Photo-detector

A 5 mm thick BB0 crystal is placed in the optical line, and amplified pulses are passed onto it. Second harmonic generation is produced, converting the input center wavelength of 800 nm to a wavelength of 400 nm. A comprehensive discussion on the physical process of second harmonic generation has been reported [28].

A $GaP\ PDA25k - EC$ amplified detector from Thorlabs was used. This detector has a high detection efficiency around 400 nm. Figure 5.7 displays the setup. The SHG signal is collected and the signal is then relayed to an oscilloscope.

5.2.5 The Oscilloscope

A Tektronix DPO 4034 digital phosphor oscilloscope was used in this experiment. The scope is set to read data in a specific way by the main program. The

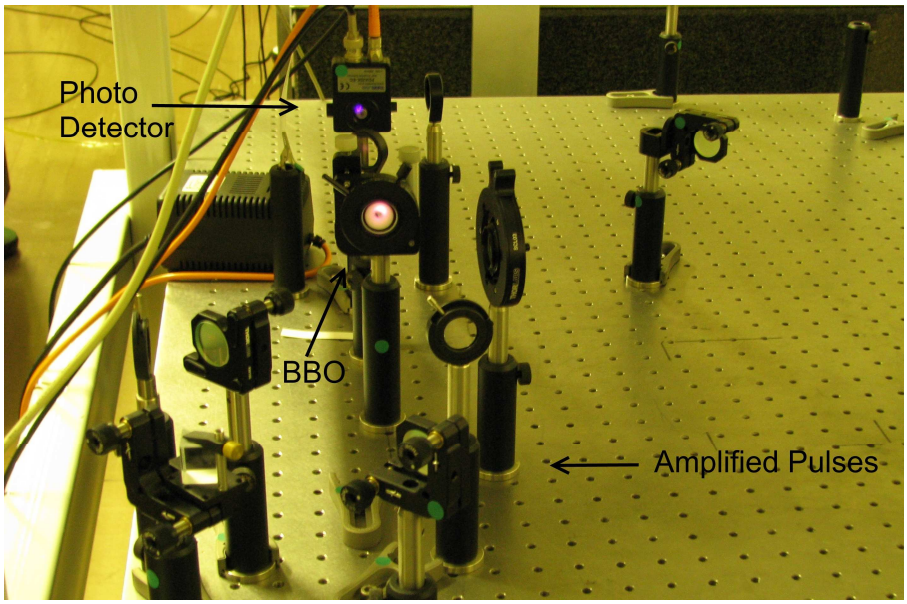

Figure 5.7: A picture of the BB0 crystal and photo-detector setup.

scope reads a maximum voltage and then passes this onto the main program which then proceeds to enter this as a fitness value in the string population within the genetic algorithm. The signal from the photo-detector is relayed via a co-axial cable and the oscilloscope communicates with the personal computer via a USB cable and a set of drivers supplied from Tektronix.

5.2.6 Personal computer with genetic algorithm and feedback control.

To automate the search for the shortest pulse duration, a feedback control loop was constructed. The main program controlling the feedback loop is written in LABVIEW. The genetic algorithm is written in C++ but is compiled as a Dynamic Link Library(DLL). This allows other programs and languages such as LABVIEW, to call functions in this library.

When the main program executes, the initial user-entered parameters are taken as input, including the population size and string length. These inputs are passed to a function which is called from a DLL. The DLL consists of all the genetic algorithm functions written in C++. The function initializes the genetic algorithm by randomly assigning GVD values to each individual string in the population. Each string now has a randomly assigned GVD value. The main program returns from the library function and then iteratively steps through the string population in the following manner.

In turn, each string's GVD value is updated to the DAZZLER. The DAZ-

ZLER is given a short space of time to update, the newly shaped pulse is amplified and passed onto the BBO crystal and then the corresponding SHG signal is measured. This measured value is now entered as that specific string's fitness value. Once this has been completed an initial population, with random GVD values and now with measured fitness values, has been formed. The genetic algorithm is now in a position to begin its search process.

The main program then calls the genetic algorithm's 'next generation' function, and repeats this process until the user specified number of generations has been reached. Reproduction, crossover and mutation is performed on each generation. This produces new GVD values which need fitness values assigned to them. So the pulse shapes are updated and SHG signals measured for every string in the population. After each generation has been completed, the average fitness and average GVD values are stored in a file. After the final generation this file is used to monitor the progress of the populations fitness and GVD values. From this, it can be determined if the algorithm converged to the optimum or not.

The results of this SHG maximization experiment are presented in chapter 6.

5.3 Background free Auto-correlation

Intensity auto-correlation is essentially an indirect technique, where the pulse is referenced against itself. A pulse is replicated and given a time delay with respect to the initial pulse and then sampled against each other. It is from the resulting integrated signal that the pulse duration is obtained. Figure 5.8 depicts the setup of a background free auto-correlator. As the optical path is non-collinear, the background signal is removed and only the integrated signal is collected.

As the pulse enters the setup, it is immediately replicated, or split by a 50/50 beam splitter. These two equal pulses pass into the two arms of the interferometer. The shorter arm follows a path such that when the second pulse is propagated in the same direction, there is a spatial separation between pulses. The second arm of the interferometer, has two mirrors placed on a translation stage. The translation stage allows for the change in optical path length and thus the replicated pulse may be given a time delay τ. If d is the delay line or the extended optical path length, then,

$$\tau = \frac{2d}{c}. \tag{5.8}$$

The two non-collinear pulse's are focused into a non-linear crystal, a BBO crystal. The BBO frequency doubles the incoming light, converting the incoming 800 nm light to a wavelength of 400 nm. Visibly this changes the red color of the light to a bright blue color. This frequency doubling occurs in the same

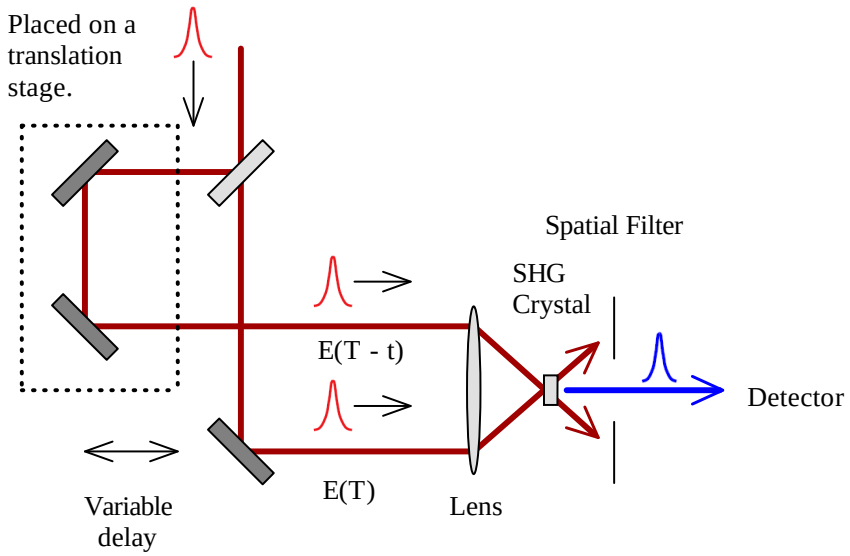

Figure 5.8: A schematic of background free auto-correlation.

direction as the incoming light, but if a specific phase matching condition is fulfilled, then frequency doubled light is generated along the bisector of the incoming pulses. The fundamental light as well as the corresponding SHG it generates, propagates along the incoming direction, and is finally blocked out by a spatial filter. Only the SHG in the forward direction is allowed to pass onto the detector.

The electromagnetic field of the SHG signal is expressed as,

$$E_{SHG}(t,\tau) \propto E(t) \cdot E(t-\tau). \tag{5.9}$$

As $I \propto |E^2|$, this proportionality also applies to the intensity. The auto correlation signal is given as,

$$A_{int}(\tau) = \int_{-\infty}^{+\infty} I(t) \cdot I(t-\tau)dt \tag{5.10}$$

which is a well known auto-correlation function in mathematics. If one of the time dependent intensity functions in equation 5.10 is known or given, then the measurement of $A_{int}(\tau)$ will directly produce the other function. As none of the time dependent intensity functions are known, an assumption of the initial pulse is made. If the assumption is invalid then major discrepancies

of the actual pulse duration and shape will be deduced. The normalized auto-correlation signal, detected with a slow detector is given by,

$$A_{int}(\tau) = \frac{\int I(t) \cdot I(t-\tau)}{\int I^2(t)} dt. \tag{5.11}$$

The maximum signal is obtained when the pulses overlap exactly in time and the minimum signal is produced when there is no overlap at all. Figure 5.9 depicts an auto-correlation signal for a specific τ. As the translation stage moves, the time delay changes, and a different auto-correlation signal is produced. Once an entire scan has been completed, then calculations may be made to produce the time duration.

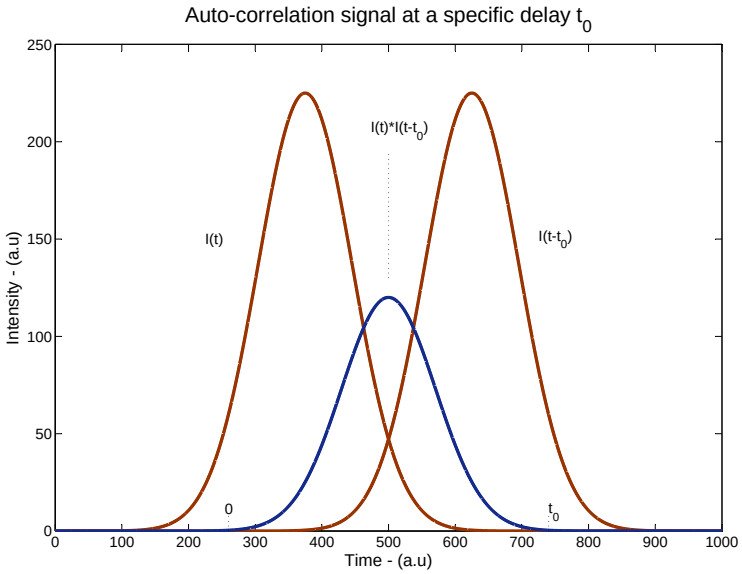

Figure 5.9: An auto-correlation signal for a specific time delay between the pulses. In this graph t_0 represents τ.

An auto-correlation trace is a plot of auto-correlation signal vs scanning delay. This data set is collected by an oscilloscope and transfered to a personal computer where a curve fitting process determines the FWHM of the trace. From this auto-correlation trace FWHM, an actual FWHM pulse time duration may be determined. Auto-correlation is used in this work to provide a reasonably accurate measurement of USP's. This information is not used directly, but only as to confirm results. This description of background auto-correlation was taken from [29].

There are many possible improvements to be made to this auto-correlation system. Due to the assumptions and inaccuracies of auto-correlation as an optical diagnostic technique [30], Frequency Resolved Optical Gating (FROG) is currently being implemented as a substitute. Pulse duration measurements are now expected to be highly accurate.

Chapter 6 - SHG maximization experiment - results and discussion

6.1 Introduction

The SHG maximization results are presented here. In section 4.3 optimization problems were solved with genetic algorithms. In figure 6.1 the measured objective function is displayed and it is this relationship which is optimized.

6.2 SHG maximization results and discussion

The optimization problem is worded as a SHG signal maximization experiment. Only one parameter, the imposed GVD by the acoustic pulse, is varied. Figure 6.1 is aptly named a 'chirpscan'. As the DAZZLER software is able to directly manipulate the imposed acoustic GVD, a chirpscan is made whereby the SHG signal is measured for different GVD values.

In the graph, it can be seen that the maximum SHG signal is measured close to 550 mV. The corresponding GVD value is close to -14000 fs^2/mm. As the pulse shaping system includes an AOM crystal which is naturally a dispersive medium, some dispersion compensation would need to be included in order to leave the input pulse duration unaltered. As previously discussed, any positively dispersed pulse can be negatively dispersed to retrieve the initial pulse duration. Thus the shaping software system has a 'self-compensate' mode which automatically sets the imposed GVD to a value of -12500 fs^2/mm, which automatically compensates for the dispersive effect of the pulse shaping system. The maximum SHG signal is found to be in the region of -14000 fs^2/mm.

The maximization of the SHG signal implies a pulse duration minimization. This is confirmed by figure 6.2. The GVD value in these two graphs may not correspond exactly, but note must be made that there are errors in the data capture procedure. The sensitivity of the photo-detector and oscilloscope combination needs to be characterized before any serious conclusion to the discrepancy in SHG maximum and pulse duration minimum can be made.

The reason for the 'gap' in the measured data sets is due to the fact that the response filter function present at those specific GVD values are extremely sharp functions. The convolution of the input pulse and a sharp 'delta-like' function has an extremely small interaction area. The output power of the dazzler shaping system depends on this interaction area. For small areas, very

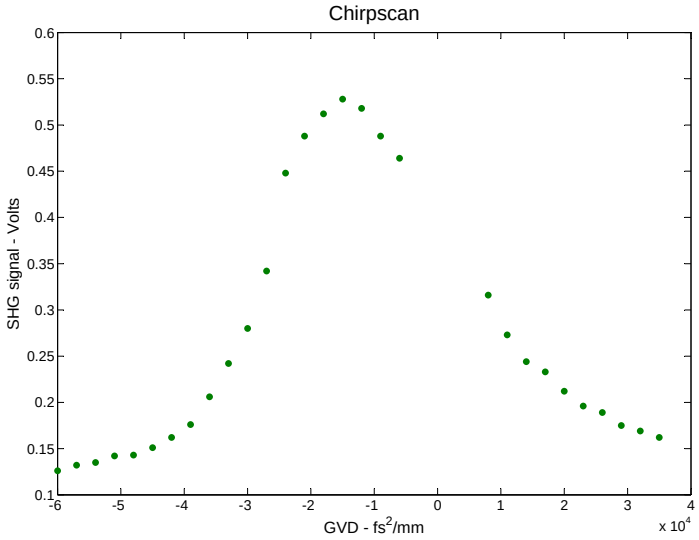

Figure 6.1: This is the relationship between the GVD imposed by the optical pulse and the SHG produced by the shaped output pulse.

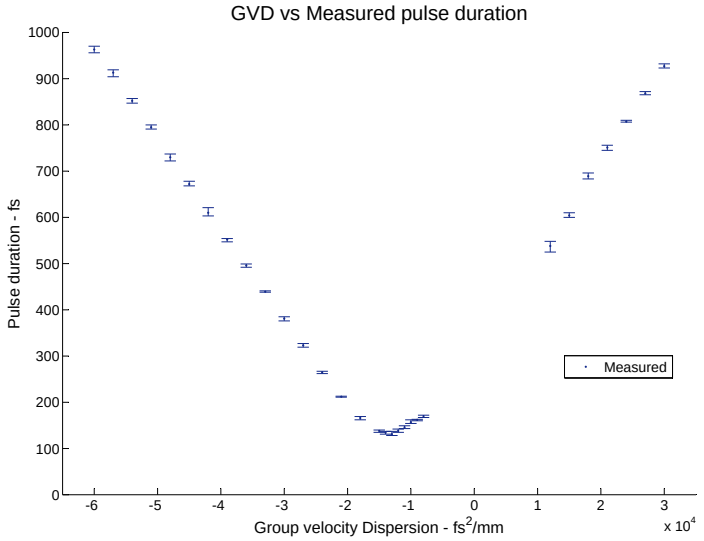

Figure 6.2: The relationship between the input pulse duration and the imposed GVD. The minimum pulse duration should correspond with the maximum SHG signal in figure 6.1.

low powers are produced hence the missing data points. This occurs where the GVD value is close to zero. Figure 6.3 illustrates this point. A full explanation

has been reported [31].

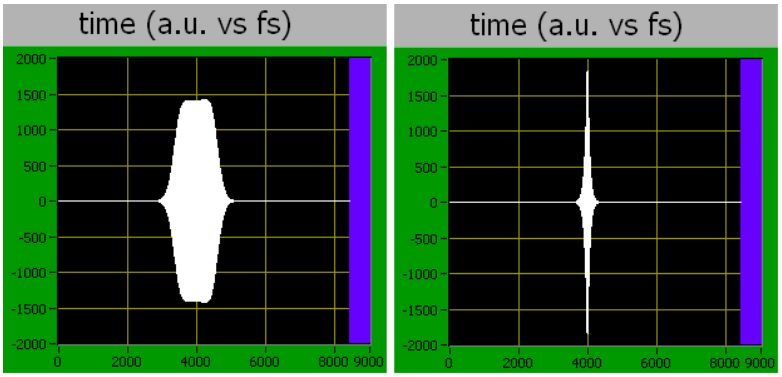

Figure 6.3: The response function depicted on the right is the above mentioned sharp function. The interaction area formed between the input pulse and this function produces extremely low powers. The second function depicted on the left will form a larger interaction area when convoluted with the input pulse. The main difference is the amount of GVD these response functions impose on the input pulse.

The genetic algorithm looks for the maximum peak in the relationship displayed in figure 6.1. A genetic algorithm is not needed to find the peak of this function and any optimization method could have found the maximum with comparable results. The idea behind using the genetic algorithm is that when pulse shaping is conducted over a larger number of parameters and complicated objective functions, or even unknown objective functions, then the genetic algorithm will perform where others will fail.

Figure 6.4 shows three separate experimental results. The highest string population finds the optimum in fewer generations than the others, although the population size of ten strings seems to perform reasonably. The string population of 6 does not find the maximum at all. At around generation number 48, the population takes on a constant value. My conclusion is that due to the small number of strings, it is highly likely that eventually the population will consist of 6 identical strings. If two identical strings are crossed with each other, the outcome is two of exactly the same strings. There will be no change in GVD setting and thus no change in SHG. The only way this can change over subsequent generations is under the mutation operator. The chance of mutation is low as the population size is small.

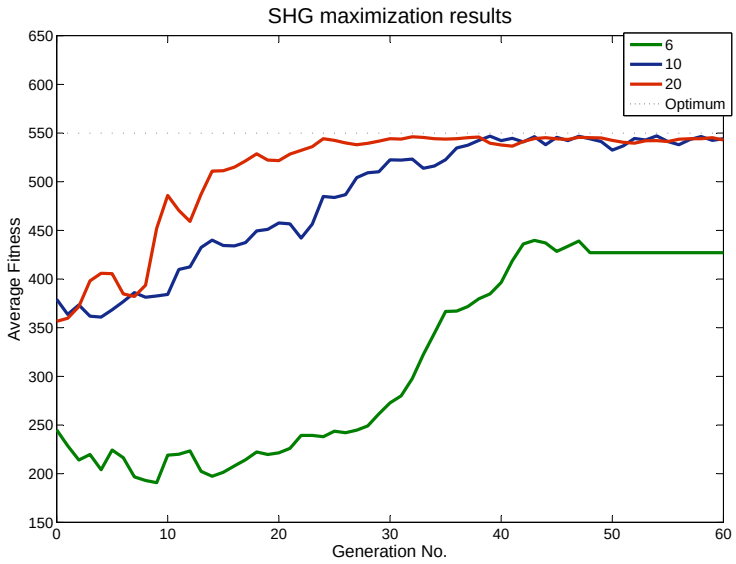

Figure 6.4: Three separate experimental results are displayed. The difference being the string population size, displayed at 6,10 and 20 strings. The string population of 20 finds the optimum relatively quickly and the string population of 6 seems not to find the optimum at all. This may be crudely associated with the effect of 'inbreeding' found in nature.

The number of strings in a population has a decisive effect on the outcome of such an optimization problem. In figure 6.5, a population size of 30 strings was chosen and the average fitness monitored over 40 generations. The optimum is found at a shorter number of generations than was found by the experiment with 20 strings.

The reason for not choosing a much higher population is simple. Firstly, the complexity of this specific problem does not require a large population and secondly, the feedback loop takes a considerable amount of time to step through a single generation and obtain fitness values. The experiment plotted in figure 6.5 took close to 40 minutes to complete. The larger the population size, the longer time period taken to complete. That having been said, a higher population size would almost certainly take fewer generations to converge.

6.3 Improvements

In order to improve convergence times the following points should be considered.

An investigation into the exact amount of time needed to allow the AOM to refresh and then an accurate measurement to be made, should be completed.

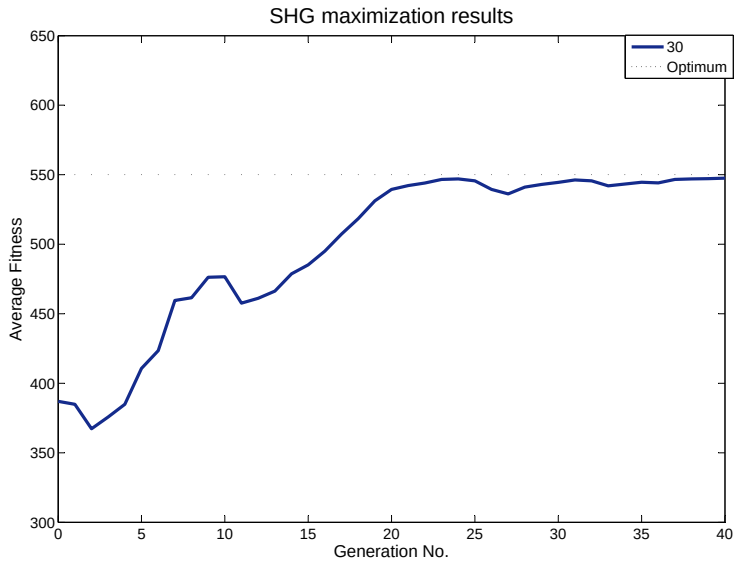

Figure 6.5: An optimization result for a population size of 30 strings.

Currently the delay period is set at $2s$. This means that for every string in the population, a delay of $2s$ is taken, at every generation in the experiment. Knowing exactly how much time is needed will greatly reduce this time delay.

In reference to figure 6.1 it can be seen that strings with high fitness, are assigned values close to 550 mV where as weaker, less fit strings are assigned values of anywhere between $150 - 300$ mV. In terms of GA progression, this means that it is not so easy to select fitter strings as the selection probability is only so much higher than the weaker strings.

A solution to this which is easily implemented is scaling the fitness values linearly. If y is the final fitness value and x the measured fitness value, then a scale of the form,

$$y = Ax + B \tag{6.1}$$

can be applied, where A and B are constants. This will improve selection of fitter strings in the reproduction operator and thus improve convergence times. Setting $A = 100$ and $B = 1000$, the following SHG maximization evolutionary plots were produced. In figure 6.6 the relationship between the imposed GVD and the SHG signal is plotted. It is considerably different to figure 6.1. The reason for the difference in intensity is that a different combination of attenuators is used in the separate scans. The reason for the shift in the central peak is somewhat more complicated. In between these chirpscans, the oscillator experienced a serious problem. One of the optical components within the cavity itself was damaged. An optical coating had been stripped and the power of the

laser had dropped dramatically. The laser was repaired but adjustments were made within the resonator. These adjustments included changing the length of dispersive medium and hence changed the chirp of the output pulse. The effect can be seen in the shift of the central peak, which is now situated at around -10000 fs^2/mm. All this means is that to produce an output pulse of shortest pulse duration (a transform limited state) a different amount of GVD needs to be imposed.

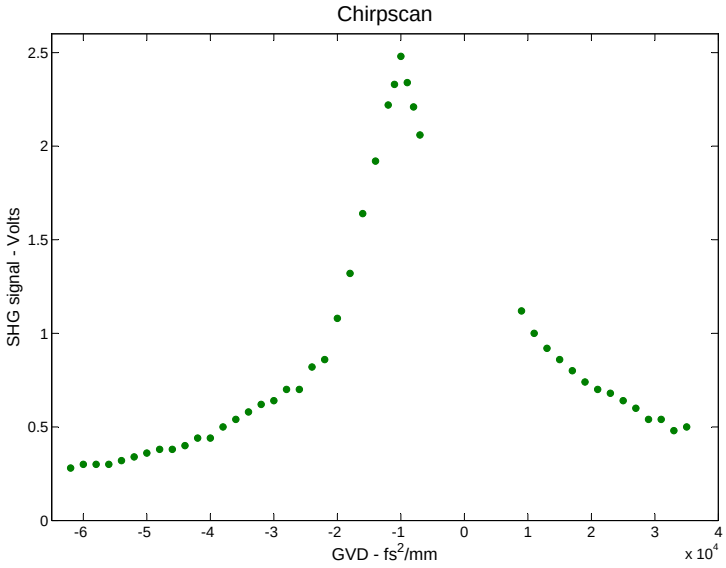

Figure 6.6: A chirpscan taken after adjustments were made to the oscillator cavity. The central peak height is seen to be close to 2500 mV.

Once the linear scale of the fitness function was implemented, two experiments were conducted to confirm the improved operation of the genetic algorithm. The experiments were run at two different population sizes of 20 and 30 strings. There is not much difference between the two, confirming the fact that for simplistic optimization problems, the population size need not be that great. Figures 6.7 and 6.8 show the evolutionary tracks of these experiments. It is noticeable that the convergence time is quicker than those in figures 6.4 and 6.5, and are now similar to the convergence paths produced by simulation in figures 4.4 and 4.8.

6.4 Conclusion

This project serves as a proof of principle experiment. The idea behind this work was to investigate time domain pulse shaping and to attempt to create the first automated time-domain pulse shaper (within the femtosecond time domain) within Africa. This has been completed and the results from initial

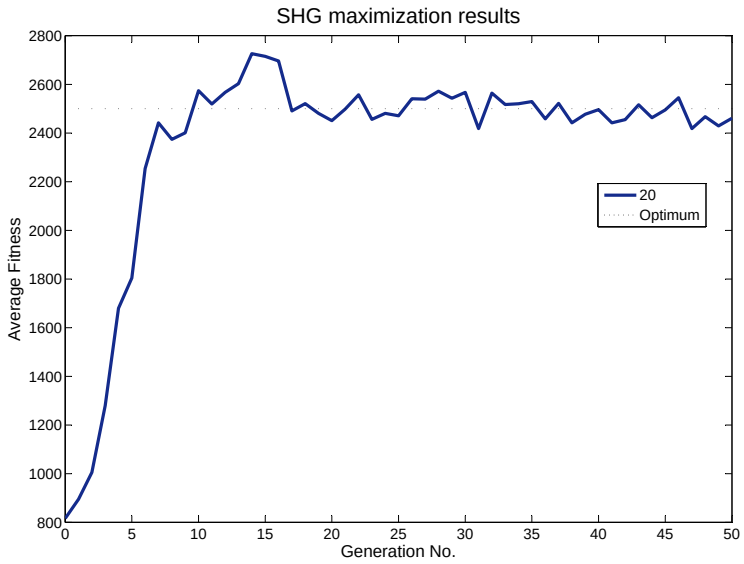

Figure 6.7: Evolutionary path of a population of 20 strings.

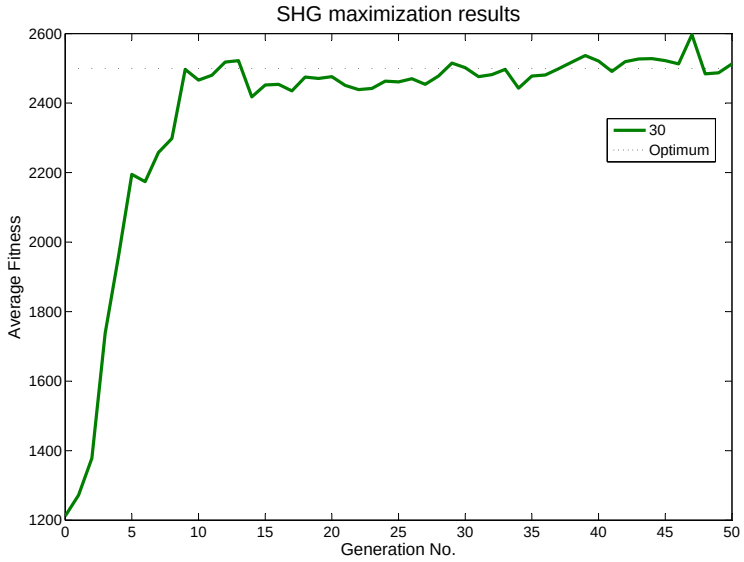

Figure 6.8: Evolutionary path of a population of 30 strings.

experiments have been reported here.

A simple shaping experiment was successfully conducted and the prospect

for complicated shaping experiments now seems plausible. The work presented here forms the basis for future shaping experiments and the feedback loop constructed will be adapted accordingly for different shaping expectations. Although it would be convenient if the change from this simple experiment to a complicated one would be a smooth transition, there may be some effort required to do so, as the written algorithm is not completely generic.

It has been shown that laser pulse shaping by means of an AOPDF (DAZZLER) in uncomplicated setups has been achieved. Implementation of a genetic algorithm as an optimization method produced results which allows for the re-use of the algorithm, adjusted to search numerous parameter spaces simultaneously. Successful construction of a feedback control loop will allow for future processes to be fully automated.

Dispersion compensation was successfully automated to the point where the duration of pulses after the shaping system have been found to be consistent and sufficiently short. Creating or searching for the shortest possible pulse duration is now possible. Regardless of what dispersive optical components are inserted before the DAZZLER (up to limitations on the shaping device), it can now be ensured that pulse durations match the original pulse durations from the output coupler of the oscillator. This is all done through an automatic process and simply requires the user to start a program.

In conclusion, I am happy to present this work as a successful piece of research with a definitive outcome. I am proud to allow this work to be used as a basis for future shaping experiments.

Bibliography

[1] T. Brixner, A. Oehrlein, M. Strehle and G. Gerber.,'Feedback-controlled femtosecond pulse shaping,' *Applied Physics B*, pp. 110-124, 2000.

[2] Oleg J. Korovyanko, Robert Rey-de-Castro, Christopher G.Elles and Robert A. Crowell.,'Optimization of a femtosecond Ti:Sapphire amplifier using an acoust-optic programmable dispersive filter and a genetic algorithm,', volume 6100, 2006.

[3] T. Baumert, T. Brixner, V. Seyfried and G. Gerber.,'Femtosecond pulse shaping by an evolutionary algorithmwith feedback.' *Applied Physics B*, volume 65, pp. 779-782, 1997.

[4] C. Rulliere,'Femtosecond Laser Pulses, Principles and experiments.' , Springer, 1998.

[5] D. J. Kuizenga,'Generation of Short Pulses for Laser Fusion in an Actively Mode-Locked Nd:YAG Laser.' *IEEE JOURNAL OF QUANTUM ELECTRONICS*, volume 13, no. 9, pp. 878-879, 1977.

[6] S.L. Huang, J.W. Lin, C.C. Kang and H.Z. Cheng,'Diode-laser pumped high-power and actively mode-locked Nd:YLF laser with fast switching.' *Optical and Quantum Electronics*, volume 32, no. 4, pp. 609-617, 2000.

[7] C. Loth and D Bruneau,'Single-frequency active-passive mode-locked Nd:YLF oscillator at 1.053 m.' *Applied Optics*, volume 21, no. 12, pp. 2091-2092, 1982.

[8] R. Paschotta and U. Keller,'Passive mode locking with slow saturable absorbers.' *Applied Physics B*, volume 73, no. 7, pp. 653, 2001.

[9] Martin E. Fermann,'Ultrafast Lasers' , Marcel Dekker, 2003.

[10] R. Bartels, S. Backus, E. Zeek and H. C. Kapteyn,'Shaped-pulse optimization of coherent emission of high-harmonic soft X-rays.' *Nature*, volume 406, 2000.

[11] H. P. Sardesai, C.-C. Chang and A. M. Weiner,'A Femtosecond Code-Division Multiple-Access Communication System Test Bed.' *Journal Of Lightwave Technology*, volume 16, no. 11, 1998.

[12] C.-C. Chang, H. P. Sardesai and A. M. Weiner,'Dispersion-free fiber transmission for femtosecond pulses by use of a dispersion-compensating fiber and a programmable pulse shaper.' *Optics Letters*, volume 23, no. 4, 1998.

[13] S. A. Kovalenko , A. L. Dobryakov , J. Ruthmann and N. P. Ernsting,'Femtosecond spectroscopy of condensed phases with chirped supercontinuum probing.' *Physical Review A*, volume 59, pp. 2369 - 2384, 1999.

[14] Anthony E. Siegmann,'Lasers.' *University Science Books*, 1986.

[15] Gavin D. Reid and Klaas Wynne,'Ultrafast Laser Technology and Spectroscopy.' *Encyclopedia of Analytical Chemistry*, volume 35, no. 4, 2000.

[16] G. Ghosh, M. Endo and T. Iwasaki,'Temperature-dependent Sellmeier coefficients and chromaticdispersions for some optical fiber glasses.' *Journal of Lightwave Technology*, volume 12, no. 8, pp. 1338 - 1342, 1994.

[17] Debabrata Goswami,'Optical pulse shaping approaches to coherent control.' *Physics Reports*, volume 374, 2003.

[18] S. Backus, C.G. Durfee, M.M. Murnane and H.C. Kapteyn,'High Power Ultrafast Lasers.' *Review of scientific instruments*, volume 69, no. 3, 1998.

[19] R.L. Fork, O.E. Martinez and J.P. Gordon,'Prism Compression.' *Optics Letters*, volume 9, no. 3, pp. 150-152, 1984.

[20] Edmond B Treacy,'Optical pulse compression with diffraction gratings.' *IEEE Journal of Quantum electronics*, volume qe-5, no. 9, 1969.

[21] Stefan Witte,'Terawatt-intensity few-cycle laser pulses.' *Vrije universiteit, amsterdam*, 2007.

[22] A.M Weiner,'Femtosecond pulse shaping using SLM's.' *Review of scientific instruments*, volume 71, no. 5, 2000.

[23] Frederic Verluise, Vincent Laude, Jean-Pierre Huignard and Pierre Tournois,'Arbitrary dispersion control of ultrashort optical pulses with acoustic waves.' *J. Opt. Soc. Am. B*, volume 17, no. 1, 2000.

[24] Frederic Verluise, Vincent Laude, P. Tournois et al,'Amplitude and phase control of ultrashort pulses by use of an acousto-optic programmable dispersive filter: pulse compression and shaping.' *Optics Letters*, volume 25, no. 8, 2000.

[25] A. Yariv and P. Yeh,'Optical waves in crystals.' *Optical waves in Crystals*, volume 1, 1984.

[26] Pierre Tournois,'Acousto-optic programmable dispersive filter for adaptive compensation of group delay time dispersion in laser systems.' *Optics communications*, volume 140, no. 8, pp. 245-249, 1997.

[27] David E Goldberg,'Genetic Algorithms.' *Addison - Wesley Publishing Company*, 1989.

[28] Robert W. Boyd,'Nonlinear Optics.' *Academic Press*, 1st Edition, 1992.

[29] Florian Kienle,'Temporal Measurements on ps and fs-Laser systems:Autocorrelation Techniques.' *Diploma Thesis: Aalen University of Applied Sciences*, 2007.

[30] D. J. Kane and R. Trebino,'Characterization of arbitrary femtosecond pulses using frequency-resolved optical gating.' *IEEE J. Quantum Electron*, volume 29, 1993.

[31] D Kaplan and P Tournois,'Theory and performance of the acousto optic programmable dispersive filter used for femtosecond laser pulse shaping.' *J. Phys. IV France*, volume 12, pp. 69-75, 2002.

Appendix A - Least squares data fitting

This appendix is a description of the least squares curve fitting program I used to fit Gaussian curves to measured data. The fitting function is somewhat different than the usual least squares fit, but only in slight variation.

We want to fit a Gaussian curve of the following form to captured data from a spectrometer for example,

$$y(x) = \frac{a}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{A.1}$$

where a is the amplitude of the Gaussian, σ is the standard deviation and μ is the mean. Firstly, in order to write this equation in a matrix form for a least squares fit, we linearize equation A.1 in the following way. By rearranging and taking logs both sides of equation A.1 yields

$$\frac{y\sigma\sqrt{2\pi}}{a} = e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{A.2}$$

$$log\left(\frac{y\sigma\sqrt{2\pi}}{a}\right) = log\left(e^{-\frac{(x-\mu)^2}{2\sigma^2}}\right) \tag{A.3}$$

$$log\left(y\sigma\sqrt{2\pi}\right) - log(a) = -\frac{x^2 + 2x\mu + \mu^2}{2\sigma^2} \tag{A.4}$$

$$log\ y(x) = log(a) - log\left(y\sigma\sqrt{2\pi}\right) - \frac{1}{2\sigma^2}x^2 + \frac{\mu}{\sigma^2}x + \frac{\mu^2}{2\sigma^2} \tag{A.5}$$

This equation is now in the form of a straight line and we are able to represent this relationship in matrix form,

$$Y = Ax^2 + Bx + C \tag{A.6}$$

$$Y = \begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n^2 & x_n & 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} \tag{A.7}$$

where

$$Y = log\ Y \tag{A.8}$$

$$A = -\frac{1}{2\sigma^2} \tag{A.9}$$

$$B = \frac{\mu}{\sigma^2} \tag{A.10}$$

$$C = -\frac{\mu^2}{2\sigma^2} + log\left(\frac{a}{\sigma\sqrt{2}}\right) \tag{A.11}$$

This leaves us in the position to calculate the mean, standard deviation and the amplitude of the fitted curve. The following code insert was written in MATLAB and contains the least squares fit as well as the rest of the program. The least squares fitting section is commented.

```
clear all
clc;
format long e;
c = 299792458; % m/s

% Data file consists of 2 Columns : 1st Column has the wavelength
% and the second column has the corresponding photon count at that
% specific wavelength.
data = dlmread('oscillator1.txt','\t');
wavelength = data(:,1);
photon_count = abs(data(:,2));

counter=1;
max_t = max(photon_count);

% storing the two columns into seperate data structures and
% eliminating data below 10% of the maximum data point.
for i=1:size(wavelength,1)
    if photon_count(i,1)>max_t*0.10
        photon(counter,1)=photon_count(i,1);
        wave(counter,1)=wavelength(i,1);
        counter=counter+1;
    end
end

% linearizing the system in order to apply least squares fit.
%logging both sides
log_photon_count = log (photon);
%creating the matrix
A=[wave.^2 wave ones(size(wave,1),1)];
% calculating the co-efficient matrix
x=A\log_photon_count;

% calculating the standard deviation and mean
sigma = sqrt(-1/(2*x(1,1)));
mu = x(2,1)*sigma^2;
% Amplitude
amp=sigma*sqrt(2*pi)*exp(x(3,1)+(mu^2)/(2*sigma^2));

% Final form of equation with fitted parameters
y = amp./(sigma.*sqrt(2.*pi)).*exp( -((wavelength - mu).^2) /(2*sigma.^2) );
y_temp = amp./(sigma.*sqrt(2.*pi)).*exp( -((wave - mu).^2) /(2*sigma.^2) );
```

```
distance = abs(y_temp - photon);
Standard_dev_of_error = std(distance);

figure;
plot(wavelength,y,'r','linewidth',3)
hold on
plot(wavelength,photon_count,'.')
axis([mu-4*sigma mu+4*sigma 0 max(photon_count)]);
xlabel('Wavelength (nm)');
ylabel('Intensity (A.U.)');
title('Oscillator Spectrum at 75,75 MHz')
legend('Fitted','Measured')

FWHM = fwhm(wavelength,y);
temp = round(num2str(FWHM));
FWHM = FWHM*1e-9;

mu_temp = mu*1e-9;
frequency = (c/mu_temp^2)*FWHM;
frequency_temp = frequency*1e-12;
pulse_duration = 0.44/frequency;
temp_pd = pulse_duration*1e15;

text(mu+1.5*sigma,max(photon_count)*0.63,['Center wavelength = ',num2str(round(mu)),' nm']);
text(mu+1.5*sigma,max(photon_count)*0.6,['FWHM Wavelength bandwidth = ',temp,' nm']);
text(mu+1.5*sigma,max(photon_count)*0.57,['Frequency Bandwidth = ',num2str(frequency_temp),' THz']);
text(mu-3*sigma,max(photon_count)*0.57,['Theoretically implied']);
text(mu-3*sigma,max(photon_count)*0.54,['pulse duration = ',num2str(temp_pd),' fs']);
text(mu-3*sigma,max(photon_count)*0.45,['Standard deviation = ',num2str(Standard_dev_of_error)]);
```

Appendix B - A simple genetic algorithm written in C code.

```
#include<stdio.h>
#include<stdlib.h>
#include <time.h>
#include <unistd.h>
#include <math.h>

#define N 50
#define LENGTH 8
#define MAXGEN 100
#define TRUE 1
#define FALSE 0

// #note NB: Length defines the variable space

int sumfitness;
int sumvalue;
float avefitness;
int min_fit;
int max_fit;
int min_val;
int max_val;
int nmutate;
int ncross;
int temp_mutate;
int parammax;
int parammin;
int xglobal;

struct ELEMENT{
int number;
int string[LENGTH];
int xval;
int  fitness;
int parent1[LENGTH];
int parent2[LENGTH];
int cross;
};

typedef struct ELEMENT pop;

int convert2val(int string[])
{
```

```
int j = 0;
int total = 0;
int temp = 0;
for(j=LENGTH-1;j>=0;j--)
{
temp = (LENGTH - 1) - j;
total = total + string[j]*pow(2,temp);
}
return total;
}

//calculate the decimal values from binary
int convert(pop population[], int index)
{
int j = 0;
int total = 0;
int temp = 0;
for(j=LENGTH-1;j>=0;j--)
{
temp = (LENGTH - 1) - j;
total = total + population[index].string[j]*pow(2,temp);
}
return total;
}

//specify the fitness function here: this is the actual function to be optimized
int fitnessfunction(int value)
{
int fit = 0;
fit = value*value; //f(x) = x^2;
return fit;
}

void initclear(pop population[])
{
int i,j;
int temp = 0;
int total = 0;
for(i=0;i<=N-1;i++)
{
population[i].number = 0;
population[i].cross = 0;

for(j=0;j<=LENGTH-1;j++){
population[i].string[j] = 0;
population[i].parent1[j] = 0;
population[i].parent2[j] = 0;
}
population[i].xval = 0;
population[i].fitness = 0;
}
}

void init(pop population[])
```

```
{
int i,j;
int temp = 0;
int total = 0;
int tempmin[LENGTH];
int tempmax[LENGTH];
for(j=0;j<=LENGTH-1;j++)
{
tempmin[j] = 0;
tempmax[j] = 1;
}

parammin = convert2val(tempmin);
parammax = convert2val(tempmax);
for(i=0;i<=N-1;i++)
{
population[i].number = i;
population[i].cross = 0;
for(j=0;j<=LENGTH-1;j++){
population[i].string[j] = rand()%2;
population[i].parent1[j] = 0;
population[i].parent2[j] = 0;
}
population[i].xval = convert(population,i);
population[i].fitness = fitnessfunction(population[i].xval);
}
}

void display(pop generation[])
{
int i,j;
for(i=0;i<=N-1;i++){
printf("-------------------------\n");
printf("- Index: %d              -\n", generation[i].number);
printf("- Value: %d             -\n", generation[i].xval);
printf("- Fitness: %d          -\n", generation[i].fitness);
printf("- Bit string: ");
for(j=0;j<=LENGTH-1;j++){
printf("%d", generation[i].string[j]);
}
printf("     -\n");
printf("- Parent1: ");
for(j=0;j<=LENGTH-1;j++){
printf("%d", generation[i].parent1[j]);
}
printf("        -\n");
printf("- Parent2: ");
for(j=0;j<=LENGTH-1;j++){
printf("%d", generation[i].parent2[j]);
}
printf("        -\n");
printf("- Crossite: %d           -\n",generation[i].cross);
printf("-------------------------\n");
}
```

```
}

int selection(pop generation[])
{
int partsum;
int selected = 0;
float r = 0; //random number between 0 and 1
float wheel_point = 0;
int i;

r = (float)(random()%100);
r = (r/100);
wheel_point = r * sumfitness;

i = 0;
partsum = 0;
partsum = partsum + generation[i].fitness;
while((partsum < wheel_point)&&(i < N-1)) {
i = i + 1;
partsum = partsum + generation[i].fitness;
}
selected = i;
return selected; //must return the selected index
}

int headstails(float probability)
{
float temp = 0;
temp = random()%100;
temp = temp/100;
if(temp<probability)
{
return TRUE;
}
else
{
return FALSE;
}
}

int mutation(int bit,float pmutate)
{
int mutate = 0;
int mtemp = 0;
mutate = headstails(pmutate);
if(mutate)
{
nmutate = nmutate + 1;
temp_mutate = temp_mutate+1;
mtemp = temp_mutate%5;
if(mtemp==0)
{
temp_mutate = 0;
if(bit == TRUE)
```

```
{
bit = FALSE;
}
else
{
bit = TRUE;
}
}
return bit;
}
else
{
return bit;
}
}

void crossover(pop population[], int mate1,int mate2,float pcross,float pmutate, pop newpop[],
{
int i=0;
int crossite = 0;
int parent1[LENGTH],parent2[LENGTH];
int child1[LENGTH],child2[LENGTH];

for(i=0;i<=LENGTH-1;i++)
{
parent1[i] = population[mate1].string[i];
parent2[i] = population[mate2].string[i];
}

for(i=0;i<=LENGTH-1;i++)
{
newpop[j].parent1[i] = parent1[i];
newpop[j].parent2[i] = parent2[i];

newpop[j+1].parent1[i] = parent1[i];
newpop[j+1].parent2[i] = parent2[i];
}

if(headstails(pcross))
{
crossite = random()%(LENGTH-1);
ncross = ncross + 1;
}
else
{
crossite = LENGTH-1;
}

for(i=0;i<=crossite;i++)
{
child1[i] = mutation(parent1[i],pmutate);
child2[i] = mutation(parent2[i],pmutate);
}
if(crossite!=LENGTH-1)
```

```
{
for(i = crossite+1;i<=LENGTH-1;i++)
{
child1[i] = mutation(parent2[i],pmutate);
child2[i] = mutation(parent1[i],pmutate);
}
}
//fill in the new info in the new population
newpop[j].number = j;
newpop[j+1].number = j+1;
for(i=0;i<=LENGTH-1;i++)
{
newpop[j].string[i] = child1[i];
newpop[j+1].string[i] = child2[i];
}
newpop[j].xval = convert(newpop,j);
newpop[j+1].xval = convert(newpop,j+1);
newpop[j].cross = crossite;
newpop[j+1].cross = crossite;

newpop[j].fitness = fitnessfunction(newpop[j].xval);
newpop[j+1].fitness = fitnessfunction(newpop[j+1].xval);
}

void fullcopy(pop oldpop[],pop newpop[])
{
int i,j = 0;
for(i=0;i<=N-1;i++)
{
oldpop[i].number = newpop[i].number;
oldpop[i].xval = newpop[i].xval;
oldpop[i].fitness = newpop[i].fitness;
oldpop[i].cross = newpop[i].cross;
for(j=0;j<=LENGTH-1;j++)
{
oldpop[i].string[j] = newpop[i].string[j];
oldpop[i].parent1[j] = newpop[i].parent1[j];
oldpop[i].parent2[j] = newpop[i].parent2[j];
}
}
}

void stats(pop population[],int t)
{
int i;
int temp = 0;
float tempx = 0;
xglobal = 0;
sumfitness = 0;
avefitness = 0;
sumvalue = 0;
min_fit = 0;
max_fit = 0;
min_val = 0;
```

```
max_val = 0;
sumfitness  = population[0].fitness;
min_fit = population[0].fitness;
max_fit = population[0].fitness;
min_val = population[0].xval;
max_val = population[0].xval;
for(i=1;i<=N-1;i++)
{
temp = population[i].fitness;
sumfitness = sumfitness + temp;
if(temp > max_fit)
{
max_fit = temp;//a new maximum fitness value
max_val = population[i].xval;
}
if(temp<min_fit)
{
min_fit = temp;//a new minimum fitness value
min_val = population[i].xval;
}
}
avefitness = (float)sumfitness/N;
tempx = sqrt(avefitness);
xglobal = roundf(tempx);

printf("-----------------------------------------------------------\n");
printf("- No. of chromosomes :  [%d]     Generation no. :  [%d]     -\n",N,t);
printf("-                                                          -\n");
printf("- Sum Fitness :  [%d]                                  -\n",sumfitness);
printf("- Ave Fitness :  [%f]                                -\n",avefitness);
printf("- Min Fitness :  [%d]                                  -\n",min_fit);
printf("- Max Fitness :  [%d]                                 -\n",max_fit);
printf("- Min Value   :  [%d]                                 -\n",min_val);
printf("- Max Value   :  [%d]                                 -\n",max_val);
printf("-----------------------------------------------------------\n");

}
void report()
{
printf("*****************************************************************\n");
printf("*                            Final Report                       *\n");
printf("*                                                               *\n");
printf("* No. of chromosomes (strings) : [%d]                          *\n",N);
printf("* Length of strings            : [%d]                            *\n",LENGTH);
printf("* No. of generations           : [%d]                          *\n",MAXGEN);
printf("* Parameter space              : [%d ... %d]                     *\n",parammin
printf("* No. of crossovers            : [%d]                         *\n",ncross);
printf("* No. of mutations             : [%d]                          *\n",nmutate);
printf("*                                                               *\n");
printf("* Average Fitness value        : [%f]                      *\n",avefitness);
printf("* Corresponding x-value        : [%d]                             *\n",xglobal);
printf("*****************************************************************\n");
}
int main()
```

```
{
printf("A simple GA\n");
FILE *file;
file = fopen("out.txt","w");
int gen_t = 0;
int index = 0;
int i,j = 0;
int temp_popsize = 0;
int mate1,mate2 = 0;
float pcross = 0;
float pmutate = 0;
avefitness = 0;

pcross = 0.9;
pmutate = 0.001;
pop oldpop[N],newpop[N];
//clearing data sets
initclear(oldpop);
initclear(newpop);
ncross = 0;
nmutate=0;
temp_mutate = 0;

//randomizing the seed in terms of the internal system clock
srand(time(NULL) + getpid());

//initiate the initial population
init(oldpop);

//printout of the population stats
stats(oldpop,gen_t);
printf("[%d] : [%d]\n",gen_t,xglobal);
fprintf (file, "%d \t %d\n",gen_t,xglobal);
//main loop
while(gen_t <= MAXGEN)
{
//population loop
temp_popsize=0;
while(temp_popsize < N-1)
{
mate1 = selection(oldpop);
mate2 = selection(oldpop);
crossover(oldpop,mate1,mate2,pcross,pmutate,newpop,temp_popsize);
temp_popsize = temp_popsize + 2;
}
//copy the new generation over the old generation
//i.e move the generation forward
fullcopy(oldpop,newpop);
gen_t = gen_t + 1;
stats(oldpop,gen_t);
printf("[%d] : [%d]\n",gen_t,xglobal);
fprintf (file, "%d \t %d\n",gen_t,xglobal);
}
report();
```

```
fclose(file);
return 0;
}
```