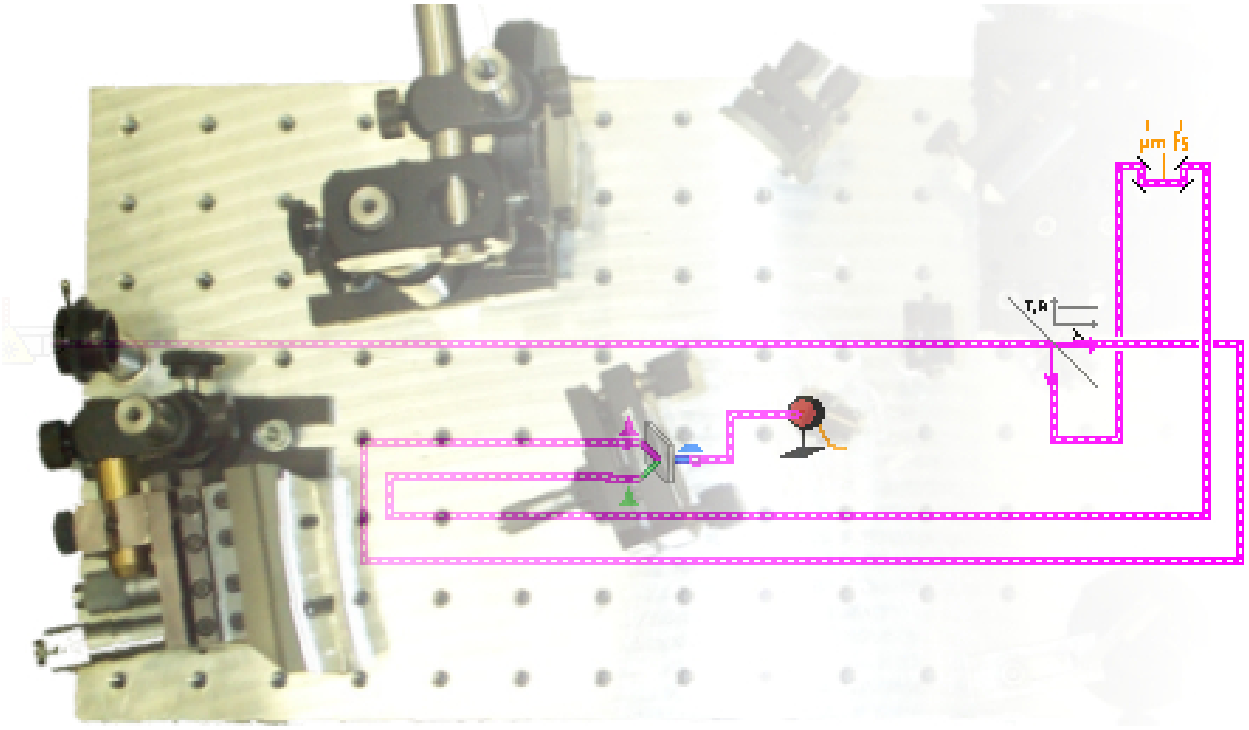
A Virtual Femtosecond Laser Laboratory



version 5.0

by: Thomas Feurer

Published in Bern, Switzerland. (version February 19, 2009)

http://www.lab2.de

Contents

Chapter 1

Introduction

Modern science without lasers is like a lake without water, boring and desert-like. Over the past decades lasers made it possible to observe motions in nature with unprecedented temporal resolution. Let us illustrate this by one example. Everybody who has taken pictures with dad's analog camera knows that the little son playing hockey may only be captured in a sharp image if the shutter time is very very short. A fraction of a millisecond is what state of the art cameras are able to do. It is fast enough for most hockey players but it is not fast enough to capture much more tiny hockey players in nature, such as single molecules or atoms. Clever people have thought of an alternative way to capture images of fast moving objects. Darken the room, leave the camera shutter open all the time, and illuminate the object you want to image with a short flash of light. People who like clubbing know what we mean, it's nothing but a strobe light. Many images like this in a row make a nice movie. The shortest strobe currently available is a (sub-)femtosecond laser. A single pulse is about 0.000000000000005 seconds long.

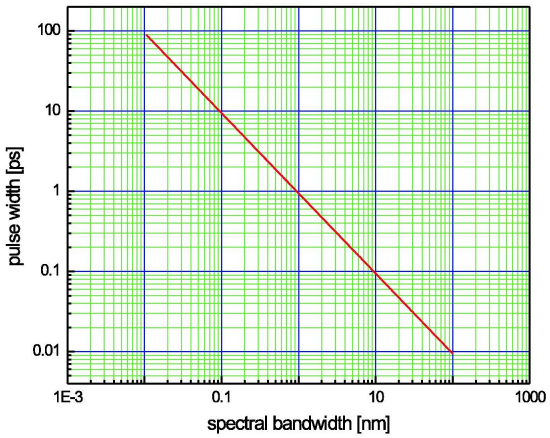


Figure 1.1: Temporal pulse width (full width at half maximum) versus spectral bandwidth for a Ti:Sapphire laser.

Because a short burst of light must be supported by a lot of spectral components, the

spectrum of a femtosecond laser pulse is usually some tens or hundreds of nanometers broad. In fig. 1.1 we show how broad the spectrum needs to be to support a laser pulse of a given temporal duration. The lasers center wavelength is 800 nm. Obviously, a pulse width of 100 ps requires a spectral bandwidth of 0.01 nm, whereas a pulse width of 10 fs already needs about 100 nm spectral bandwidth! Now, every optical component, such as a lens, a window, a dielectric mirror etc., are dispersive, meaning that they influence the different spectral components of a short laser in a different way. That, as one might guess, sometimes has dramatic effects on the laser pulse. In other cases, well controlled dispersion is used on purpose for intentional pulse manipulations. Another striking feature of ultrashort laser pulses is that the peak intensity can be enormously high although the energy is at a ridiculously low level. All sorts of nonlinear material responses show up and need to be considered when doing experiments with pulses so short. A familiar nonlinear effect is breaking the optics whenever the laser intensity is larger than the damage threshold intensity. To circumvent damaging of optics, especially in laser amplifiers, lead to the introduction of what is now known as chirped pulse amplification (CPA). To do such things as stretching or compressing pulses special optical designs are required.

In the next few chapters we introduce a modular programming tool — we called it $\mathcal{LAB}$ II because it was in fact in lab 2 of our institute where we spent so many nights trying to figure out why the heck this or that stupid thing happened — that allows to simulate most of the experimental setups in a rather intuitive way. It is almost like a virtual laboratory. Elements are placed on a white board and connected by laser beams just as one would set up an experiment in the lab. We have chosen to embed our package in `LabView` since the graphic programming language `G` allows in a unique way to program something that actually also looks like a virtual experiment.

Chapter 2 summarizes the basic math we have used for all of the programming. Chapter 3 describes the different laser sources, chapter 4 all the available linear elements, chapter 5 all the available nonlinear elements, chapter 6 all the detectors that have been implemented, and chapter 7 a selection of modules simulating laser matter interaction. The very last chapter 8 introduces optimization algorithms which are commonly used in combination with pulse shaping to reach specific goals. A couple of examples will be given.

So, enjoy reading and more importantly working with $\mathcal{LAB}^{\text{II}}$,

Thomas

For further reading we recommend the following textbooks:

▷ G.P. Agrawal; "Nonlinear Fiber Optics", Academic Press, San Diego (2001)

▷ S.A. Akhmanov, V.A. Vysloukh, A.S. Chirkin; "Optics of Femtosecond Laser Pulses", AIP, New York (1992)

▷ R.W. Boyd; "Nonlinear Optics", Academic Press, San Diego (1992)

▷ C.C. Davis; "Lasers and electro-optics", Cambridge University Press, Cambridge (1996)

▷ J.C. Diels, W. Rudolph; "Ultrashort Laser Pulse Phenomena", Academic Press, San Diego (1996)

▷ V.G. Dmitriev, G.G. Gurzadyan, D.N. Nikogosyan; "Handbook of Nonlinear Optical Crystals", Springer, Berlin (1997)

▷ S. Huard; "Polarization of Light", John Wiley & Sons, Chichester (1997)

▷ S. Mukamel; "Nonlinear Optical Spectroscopy", Oxford University Press, New York (1995)

▷ B.E.A. Saleh, M.C. Teich; "Fundamentals of Photonics", John Wiley & Sons, New York (1991)

▷ A. Yariv; "Quantum Electronics", John Wiley & Sons, Chichester (1989)

Chapter 2

Ultrashort Laser Pulses

2.1 General

First of all we will restrict ourselves to the time dependence of the electric field and omit the spatial part. This is equivalent to the viewpoint that a detector is located at a fixed position and the field only varies with time. Since the electric field of the laser pulse is in principle a measurable quantity it must be real. For most calculations, however, it is much more convenient to use the analytic signal — a complex quantity — and in a moment we will see why. So we start with an electric field described by

$$\begin{aligned} E(t) &= A(t)\ \cos\left[\Phi(t)\right] = A(t)\ \frac{1}{2}\ \left[e^{i\Phi(t)} + e^{-i\Phi(t)}\right] \\ &= \frac{1}{2}\ A(t)\ e^{i\Phi(t)} + c.c. \end{aligned} \tag{2.1}$$

where $A(t)$ is time dependent amplitude and $\Phi(t)$ the time dependent phase. The Fourier transform of the electric field is

$$E(\omega) = \mathcal{F}\{E(t)\} = \int_{-\infty}^{\infty} dt\ E(t)\ e^{-i\omega t} = A(\omega)\ e^{i\Phi(\omega)} \tag{2.2}$$

and the inverse Fourier transform is equivalently

$$E(t) = \mathcal{F}^{-1}\{E(\omega)\} = \frac{1}{2\pi}\ \int_{-\infty}^{\infty} d\omega\ E(\omega)\ e^{i\omega t}. \tag{2.3}$$

One might ask, why do we consider the Fourier transformation of a time varying laser pulse? There are many reasons to do this. The spectral analysis yields information on the wavelengths contained in the pulse and also provides a hint whether a pulse may be compressed in time or is as short as it can possibly get. The propagation of ultrashort laser

pulses through dispersive media depends strongly on the frequency and, finally, the concept of Fourier transformation is a powerful tool for solving equations.

Since the electric field is real, i.e. $E(t) = E^*(t)$, we see immediately that

$$E^*(-\omega) = \left[\int_{-\infty}^{\infty} dt\ E(t)\ e^{i\omega t} \right]^* = \int_{-\infty}^{\infty} dt\ E(t)\ e^{-i\omega t} = E(\omega). \tag{2.4}$$

This is a useful identity for a number of the following calculations. Inspecting equation 2.1 shows that the Fourier transform has two distinct distributions which are symmetric with respect to the origin of the frequency axis. Now, have you ever seen a spectrometer that can measure at negative frequencies or wavelength? Certainly not, so it is sometimes more convenient to start in the frequency domain by defining a field having only positive frequencies. The consequence is that the electric field in the time domain becomes complex. Again we end up with an unrealistic situation since we use complex instead of real Fourier transformation. None of these problems would occur, would we use a real Fourier transformation, for example the cos Fourier transformation, but the mathematical efforts would increase substantially and so nobody uses it.

$$E^+(t) = \frac{1}{2\pi} \int_0^{\infty} d\omega\ E(\omega)\ e^{i\omega t} = \frac{1}{2\pi} \int_{-\infty}^{\infty} d\omega\ E^+(\omega)\ e^{i\omega t} \tag{2.5}$$

with

$$E^+(\omega) = \begin{cases} E(\omega) & : \quad \omega \geq 0 \\ 0 & : \quad \omega < 0 \end{cases} \tag{2.6}$$

The same holds true for the negative frequency components. They give rise to a field $E^-(t)$. It is easy to show that the original field is reconstructed by

$$E(t) = E^-(t) + E^+(t) \quad \text{and} \quad E(\omega) = E^-(\omega) + E^+(\omega) \tag{2.7}$$

Now we want to become somewhat more specific and adapt the formalism to laser pulses we have to deal with in the laboratory. What are the specific features of say a Ti:sapphire laser pulse. It's 'central' wavelength is around 800 nm, the pulse width is typically around 100 fs with some exceptions where pulses do get as short as 5 fs. Because the oscillation period of the field at this wavelength is about 2.7 fs the pulses are usually longer than one period. If we record a spectrum we see that there is a broad distribution around the center wavelength with a width of some tens of nanometers. These experimentally observed characteristics influence the mathematical description in the following way. Identifying a 'central' wavelength ω_0 and a width $\Delta\omega$ in the spectrum satisfying the condition $\Delta\omega/\omega_0 < 1$ is equivalent to $\Delta t/T > 1$. This means that the electric field of the laser pulse can be split into a product of a slowly varying envelope Δt and a rapid oscillation with a period $T = 2\pi/\omega_0$. The electric field in the time domain is then written

$$E^+(t) = \frac{1}{2}\, A(t)\, e^{i\varphi(t)}\, e^{i\omega_0 t} = \frac{1}{2}\, \varepsilon(t)\, e^{i\omega_0 t} \tag{2.8}$$

where $A(t)$ is the slowly varying amplitude, $\varphi(t)$ the slowly varying phase, and $\varepsilon(t)$ a complex quantity combining the slowly varying parts. The definition of ω_0 is not unique and there are various ways to define it. A convenient one is to use the 'central' frequency, i.e. the frequency where the spectral amplitude has a global maximum. If the spectrum is structured and the maximum is not easily identified it is more convenient to use the first frequency moment. The definition of moments will appear a few times throughout the text. The n$^{\text{th}}$ order moment of x is defined

$$x^{(n)} = \frac{\int dx\; x^n\; f(x)}{\left[\int dx\; f(x)\right]^n} \tag{2.9}$$

where $f(x)$ is a distribution function. Now we have most of the basic tools and it is time to apply the formalism developed so far to Gaussian pulses. A Gaussian pulse has a Gaussian envelope function and we restrict ourselves to phases quadratic in time and we omit the constant phase. If we do this many interesting effects can be calculated analytically. The reason for choosing a Gaussian shaped envelope is simply that it allows to calculate most things analytically with very little mathematical effort.

$$E(t) = \frac{\Delta\omega}{2\sqrt{\pi}(1+A^2)^{1/4}}\; \exp\left[-\frac{\Delta\omega^2}{4(1+A^2)}t^2\right]\; \cos\left[\omega_0 t + \frac{A\Delta\omega^2}{4(1+A^2)}t^2\right] \tag{2.10}$$

Figure 2.1 shows the electric field $E(t)$ for a pulse with $A = 0$. The rapid oscillations increase with time, reach a maximum, and decrease again. This behavior is determined by the slowly varying envelope, i.e. the Gaussian.

The reason to choose this form is that in most cases we start with a pulse having a fixed spectral width $\Delta\omega$ and treat effects that do not change this width. The constant factor makes the Fourier transform of this pulse relatively simple.

$$E(\omega) = \exp\left[-\frac{(\omega-\omega_0)^2(1+iA)}{\Delta\omega^2}\right] + \exp\left[-\frac{(\omega+\omega_0)^2(1-iA)}{\Delta\omega^2}\right] \tag{2.11}$$

Clearly, the Fourier transform has two distinct frequency distributions around $-\omega_0$ and ω_0 (see fig. 2.2). The slowly varying envelope approximation breaks down when the two frequency distribution start to overlap at the origin of the frequency axis. In addition, it may be seen that a quadratic phase modulation in time leads to a quadratic phase modulation in the frequency domain. Splitting the frequency distribution in the negative and the positive component leads to

$$E^-(\omega) = \exp\left[-\frac{(\omega+\omega_0)^2(1-iA)}{\Delta\omega^2}\right] \tag{2.12}$$

$$E^+(\omega) = \exp\left[-\frac{(\omega-\omega_0)^2(1+iA)}{\Delta\omega^2}\right] \tag{2.13}$$

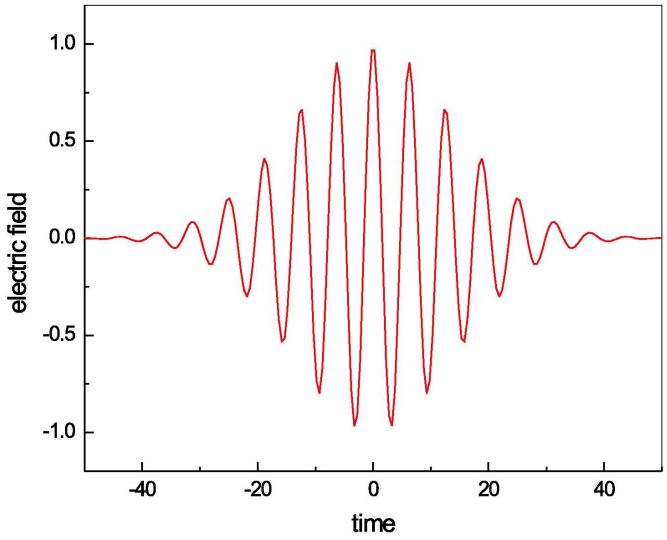


Figure 2.1: The slowly varying envelope and the rapid oscillations of a Gaussian pulse $E(t)$ are shown.

In the time domain we obtain

$$E^{-}(t) = \frac{\Delta\omega}{2\sqrt{\pi}(1+A^2)^{1/4}} \exp\left[-\frac{\Delta\omega^2}{4(1+A^2)}(1+iA)t^2 - i\omega_0 t\right] \tag{2.14}$$

$$E^{+}(t) = \frac{\Delta\omega}{2\sqrt{\pi}(1+A^2)^{1/4}} \exp\left[-\frac{\Delta\omega^2}{4(1+A^2)}(1-iA)t^2 + i\omega_0 t\right]. \tag{2.15}$$

After having defined the fields we have to answer the question what do we measure? Here is where the problems start. What do we measure? First of all this question has to be stated in a different way. If we use this specific detector what do we measure? Let's start for example with a pyro-electric detector. The laser pulse is absorbed in a ideally non-reflecting layer which subsequently gets warmer and changes its electrical resistivity. This change then is monitored on an analog display. The whole device has a time constant of a few tens of milliseconds. So what do we measure? Not surprisingly the pulse energy. Or, what happens if we use a photodiode with a time constant of ten femtoseconds? With this diode we will not resolve the rapid oscillations of the laser field but we will be able to measure the temporal evolution of the slowly varying envelope. So what we measure is mostly dependent on the type of detector which we are using. A detector with a time constant large compared to the rapid oscillations but fast compared to the temporal envelope will measure a quantity called the instantaneous intensity $I(t)$.

$$I(t) = \epsilon_0 c_0 n \frac{1}{T} \int_{t-T/2}^{t+T/2} dt' \; E^2(t') \tag{2.16}$$

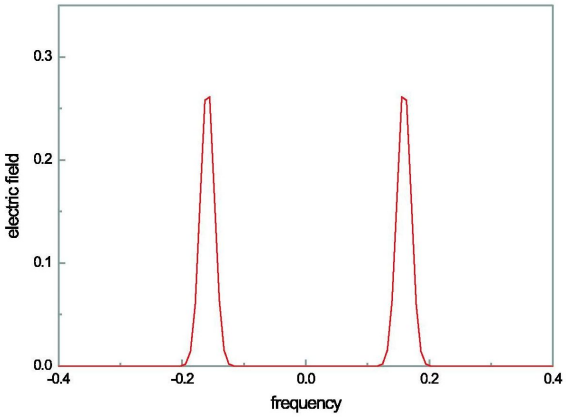


Figure 2.2: The electric field of a real Gaussian pulse $E(\omega)$ shows two distinct contributions.

For a Gaussian pulse the instantaneous intensity is

$$I(t) = \frac{1}{2}\epsilon_0 c_0 n \, \frac{\Delta\omega^2}{\pi\sqrt{1+a^2}} \, \exp\left[-\frac{\Delta\omega^2}{2(1+A^2)}t^2\right] \tag{2.17}$$

where ϵ_0 is the dielectric constant of vacuum, c_0 the speed of light in vacuum, and n the index of refraction of the transparent material the pulse is propagating in. Obviously, the integration over one period reflects the fact that we have a detector not able to resolve the field oscillations. The instantaneous intensity is measured in energy per unit time and per unit area. If the finite area A of the detector is considered we obtain

$$P(t) = \int_A d\sigma' \; I(t) \tag{2.18}$$

If the response time of the detector is even longer the pulse energy W is measured

$$W = \int_{-\infty}^{\infty} dt \; P(t) \tag{2.19}$$

which for a Gaussian is

$$W = \frac{\epsilon_0 c_0 n \Delta\omega}{\sqrt{2\pi}} \tag{2.20}$$

Making use of the slowly varying wave approximation it is relatively easy to show that the following equalities hold

$$I(t) = \frac{1}{2}\epsilon_0 c_0 n \; \varepsilon(t)\varepsilon^*(t) = 2\epsilon_0 c_0 n \; E^+(t)E^-(t) \tag{2.21}$$

A further useful quantity is the instantaneous frequency $\omega(t)$. Suppose you are able to measure a spectrum of the laser, say every femtosecond, and you plot the center frequency

of each of these spectra as a function of time, you have a good idea of what the instantaneous frequency is. A more sophisticated definition is given by two-dimensional distribution functions. Knowing the phase $\Phi(t)$ the instantaneous frequency is derived in the following way

$$\omega(t) = \partial_t \Phi(t) = \partial_t \varphi(t) + \omega_0 \tag{2.22}$$

For a Gaussian pulse we find

$$\omega(t) = \omega_0 + \frac{\Delta\omega^2 A}{2(1+A^2)} t \tag{2.23}$$

Obviously, a quadratic phase leads to a linear change of the instantaneous frequency. This is called a linear chirp. A quadratic chirp corresponds to a cubic phase and so on.

Now, suppose an ideal spectrometer is used to measure the spectrum of the pulse. Because we operate in the frequency domain, there is no such thing as time anymore. Each infinitely narrow frequency component corresponds to an infinitely long wave. Later we will see what influence the finite resolution of a real spectrometer has on the measurements. We have already noted that no spectrometer is able to measure negative frequencies, therefore, the spectrum is

$$S(\omega) = \eta(\omega)^2 |E^+(\omega)|^2 \tag{2.24}$$

where $\eta(\omega)$ contains all specific information on the spectrometer and the detection unit used, i.e. how the response of the spectrometer (mirrors, gratings, prisms, photomultiplier, CCD, etc.) depends on the frequency. For an ideal spectrometer η is a constant and can be determined using Parseval's theorem

$$\int_{-\infty}^{\infty} dt\ |E^+(t)|^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} d\omega\ |E^+(\omega)|^2 \quad \Rightarrow \quad \eta^2 = \frac{\epsilon_0 c_0 n}{\pi} \tag{2.25}$$

$$S(\omega) = \frac{\epsilon_0 c_0 n}{\pi}\ \exp\left[-2\frac{(\omega-\omega_0)^2}{\Delta\omega^2}\right] \tag{2.26}$$

Integrating the spectral intensity $S(\omega)$ over ω must again yield the total pulse energy W

$$W = \frac{\epsilon_0 c_0 n \Delta\omega}{\sqrt{2\pi}} \tag{2.27}$$

There is one thing you should never forget. Once the instantaneous intensity $I(t)$ is known, never perform a Fourier transform in order to calculate the spectrum $S(\omega)$. Calculate the electric field first, Fourier transform the field, and then calculate the spectral intensity. This is the only way to obtain the correct spectrum. Try both ways and you will see that a direct Fourier transformation leads to an erroneous result.

The last thing in this chapter is the definition of the full width at half maximum (FWHM). For the Gaussian pulse the intensity FWHM is

$$\mathrm{FWHM}_t = \frac{\sqrt{8\ln 2(1+A^2)}}{\Delta\omega} \tag{2.28}$$

and the spectral FWHM

$$\mathrm{FWHM}_\omega = \Delta\omega\sqrt{2\ln 2} \tag{2.29}$$

It is easy to see that the product of both is independent of any specific pulse parameters except the linear chirp. Note, that we only allowed for a linear chirp in the first place. Usually the time bandwidth product is different for different pulse shapes (Gaussian, sech^2, etc.) and is written in terms of frequency rather than angular frequency. Therefore, a measurement of the temporal and the spectral FWHM gives already some information on the phase modulation of the pulse. If the product of both numbers is equal to the minimum value the pulse has no phase modulation. In our analytic Gaussian model this requires $A = 0$.

2.2 Short Pulse Propagation in the Linear Regime

Linear elements are those which influence the spectral amplitude of the pulse or its spectral phase in a linear fashion. This is almost always the case if the pulse passes through optical elements and the intensity is moderate. Linear pulse manipulations in the spectral domain may be expressed as follows

$$E_{out}(\omega) = E_{in}(\omega)\cdot H(\omega)\ \mathrm{e}^{-i\Phi(\omega)}. \tag{2.30}$$

The spectral amplitude as well as the phase of the input field $E_{in}(\omega)$ may be changed by $H(\omega)$ and $\Phi(\omega)$, respectively. The generated output field is $E_{out}(\omega)$.

Almost all elements in the lab, which happen to be in the laser beam, are dispersive and modify the spectral phase of the pulse. Dispersion, in most cases, is an unwanted side effect that one would rather avoid in the first place but sometimes it is used on purpose, for example in CPA (chirped pulse amplification). The phase is usually expressed in terms of the coefficients of a Taylor series expansion.

$$\begin{aligned}\Phi(\omega) &= \sum_{k=0}^{\infty}\frac{1}{k!}\ \partial_\omega^k\Phi(\omega)\big|_{\omega=\omega_0}\ (\omega-\omega_0)^k \\ &= \sum_{k=0}^{\infty}\frac{1}{k!}\ \Phi_k\ (\omega-\omega_0)^k \\ &\quad \text{with} \quad \Phi_k := \partial_\omega^k\Phi(\omega)\big|_{\omega=\omega_0}\end{aligned} \tag{2.31}$$

Let us inspect the first two terms a little more careful. The one corresponding to $k = 0$ represents the absolute phase of the pulse. It is this term that determines the relative phase between the slowly varying amplitude and the rapid field oscillations. The next term,

corresponding to $k = 1$, consists of a constant factor multiplied by $(\omega - \omega_0)$. Recalling the properties of the Fourier transformation shows that a phase term linear in frequency, shifts the whole pulse in the time domain by a constant time delay on the time axis. Therefore, the first derivative with respect to frequency corresponds to the time a pulse needs to travel through the dispersive medium and may be used to determinate the group velocity. All terms of higher order generally change the pulse envelope in some way. It may be shown that all even order Taylor coefficients change the slowly varying envelope in a symmetric fashion whereas all odd order Taylor coefficients cause an asymmetric change.

In many textbooks the phase is expressed as follows

$$\begin{aligned} \Phi(\omega) &= \beta(\omega) z \\ &= \sum_{k=0}^{\infty} \frac{1}{k!} \; \partial_\omega^k [\beta(\omega) z]\big|_{\omega=\omega_0} \; (\omega - \omega_0)^k \\ &= \sum_{k=0}^{\infty} \frac{1}{k!} \, \beta_k z \, (\omega - \omega_0)^k \\ &\quad \text{with} \quad \beta_k := \partial_\omega^k \beta(\omega)\big|_{\omega=\omega_0} \end{aligned} \tag{2.32}$$

The number β per definition is a wave vector, or a phase per unit length. Suppose the pulse traverses a transparent but dispersive medium with thickness d, and assume the medium's dispersion is fully characterized by β_1 only, then we immediately see that

$$E_{out}(t) = \mathcal{F}^{-1}\left\{E_{in}(\omega) e^{-i\beta_1 d(\omega-\omega_0)}\right\} = E_{in}(t - \beta_1 d). \tag{2.33}$$

Since the pulse propagates with the group velocity we find that $\beta_1 = 1/v_g$ or $v_g = 1/\beta_1 = d/\Phi_1$.

2.3 Short Pulse Propagation in the Nonlinear Regime

Nonlinear effects involve products of electric fields in the time domain and since this is equivalent to a convolution in the frequency domain, nonlinear processes usually lead to frequency mixing. Depending on the order of the process there are two, three, or even more pulses involved. So far we have implemented the most commonly encountered nonlinear effects, such as three wave frequency mixing and self-phase modulation.

2.3.1 Self-phase Modulation

Self-phase modulation is in principle a third order ($\chi^{(3)}$) effect and leads to a time varying phase. The three waves involved are three replica of the original pulse. Just as a spectral phase leaves the spectrum of the pulse unchanged but has dramatic effects on the temporal intensity, a temporal phase causes the opposite, it leaves the temporal intensity the same but

changes the spectrum. An easy but somewhat sloppy way to express self-phase modulation is by introducing an intensity dependent index of refraction

$$n(I) = n_0 + n_2 I \tag{2.34}$$

which in turn causes an extra phase in the time domain

$$\begin{aligned} \Phi(t) &= \frac{\omega_0}{c_0}\, n(I(t))\, z = \frac{\omega_0}{c_0}\, (n_0 + n_2 I(t))\, z \\ \Phi_{nl}(t) &= \frac{\omega_0}{c_0}\, n_2\, I(t)\, z \end{aligned}$$

so it is essentially the product of intensity I times the propagation distance z that determines the nonlinear phase contribution. In principle one would expect the same result as one changes the material thickness z but keeps the product Iz constant by adjusting the intensity I. This is not quite true, especially for large z, since dispersion effects tend to broaden the pulse as it propagates which leads to a decreasing intensity I, and, consequently, a decrease in the nonlinear phase contribution Φ_{nl}.

A frequently used expression in this context is the so called B-integral. The B-integral is a measure of the accumulated nonlinear phase and serves as a criterion whether nonlinear effects play a role or not. It takes into account the fact that the intensity might change as the pulse propagates along the z-coordinate. It is defined by the following relation

$$B = \frac{\omega_0}{c_0} \int_0^d dz\; n_2 I(z) = \frac{2\pi}{\lambda_L} \int_0^d dz\; n_2 I(z) \tag{2.35}$$

As a rule of thumb, self-phase modulation becomes a serious issue when the B-integral exceeds 3 or 4, although effects of self-phase modulation may be seen much earlier.

In order to account for both dispersion and self-phase modulation, one often uses an algorithm called 'split step Fourier transform'. This algorithm splits the material in many slices. Each slice is again subdivided into three regions. Now, the first region takes into account one half of the linear dispersion. Then one performs a Fourier transformation to the time domain, accounts for the temporal phase due to self-phase modulation of the full slice in the second region, and Fourier transforms back. The third region then takes care of the missing half of the linear dispersion. This is done for all slices and eventually, if the number of slices is high enough, the algorithm will converge and lead to the true output pulse.

2.3.2 Three Wave Mixing Processes

The equations that describe three wave mixing processes are coupled nonlinear differential equations, one differential equation for each of the three fields. These equations need to be integrated along the direction of propagation in order to simulate three wave mixing processes in a nonlinear crystal, such as second harmonic generation, sum frequency mixing,

or difference frequency mixing. Energy conservation requires that $\omega_1 + \omega_2 = \omega_3$. The interplay between the three field $\tilde{E}_{j=1,2,3}$ is described by

$$\partial_z \tilde{E}_1(\omega_1) = -i\kappa_1 \int d\omega_2 \; \tilde{E}_2^*(\omega_2) \; \tilde{E}_3(\omega_1 + \omega_2) \; \mathrm{e}^{i\Delta k(\omega_1,\omega_2)z} \tag{2.36}$$

$$\partial_z \tilde{E}_2(\omega_2) = -i\kappa_2 \int d\omega_1 \; \tilde{E}_1^*(\omega_1) \; \tilde{E}_3(\omega_1 + \omega_2) \; \mathrm{e}^{i\Delta k(\omega_1,\omega_2)z} \tag{2.37}$$

$$\partial_z \tilde{E}_3(\omega_3) = -i\kappa_3 \int d\omega_1 \; \tilde{E}_1(\omega_1) \; \tilde{E}_2(\omega_3 - \omega_1) \; \mathrm{e}^{-i\Delta k(\omega_1,\omega_3-\omega_1)z} \tag{2.38}$$

with

$$\kappa_j = \frac{\omega_j^2 d_{\mathrm{eff}}}{c_0^2 k_j} \tag{2.39}$$

$$d_{\mathrm{eff}} = \frac{1}{2}\,\chi^{(2)} \tag{2.40}$$

$$\Delta k(\omega_1,\omega_2) = \frac{(\omega_1+\omega_2)n_3(\omega_1+\omega_2) - \omega_2 n_2(\omega_2) - \omega_1 n_1(\omega_1)}{c_0} \tag{2.41}$$

where ω_j is a frequency component corresponding to spectrum E_j and $n_j(\omega)$ is the wavelength dependent index of refraction. The coupled set of three differential equations is integrated along the direction of propagation z. The final spectral fields are obtained by propagating the solution of the differential equations with the corresponding wave vector k_j, which incorporates the linear dispersive effects

$$E_j = \tilde{E}_j \mathrm{e}^{-i\cdot k_j L} \tag{2.42}$$

In case the conversion efficiency is small the above set of differential equations may be substantially simplified. This is usually the case for thin crystals and/or rather low intensities. If we assume that both fundamental input beams are not depleted by the conversion process $\partial_z E_{1,2} \equiv 0$, only one differential equation is left to solve

$$E_3(\omega_3) = -\kappa \int d\omega_1 \; E_1(\omega_1) \; E_2(\omega_3 - \omega_1) \; \eta(\omega_1, \omega_3 - \omega_1) \tag{2.43}$$

$$\eta(\omega_1,\omega_2) = \frac{\mathrm{e}^{i\;\Delta k(\omega_1,\omega_2)\;L} - 1}{\Delta k(\omega_1,\omega_2)\;L} \tag{2.44}$$

$$\kappa = \frac{\omega_3^2 d_{\mathrm{eff}}}{c_0^2 k_3} \tag{2.45}$$

Here, E_1 and E_2 are the spectral fields of the incoming pulses — which are constant — and E_3 is the spectral field of the generated pulse. The constant factor κ is determined by the center frequency ω_3, the effective nonlinearity d_{eff} (units [pm/V]) of the specified nonlinear interaction process (which depends on the crystal orientation), the vacuum phase velocity c_0, and the wave vector at the center frequency k_3. The phase mismatch is $\Delta k = k_3 - k_2 - k_1$.

2.3.3 Optical fiber propagation

Another important nonlinear element is the optical fiber which is frequently used for a process called super-continuum generation. To describe pulse propagation along an optical fiber we need to include two more nonlinear effects, i.e. self steepening and the Raman effect, and find

$$\partial_\xi \tilde{E}^+(\xi;\tau) = -i \sum_{m=2}^{\infty} \frac{k_m}{i^m\, m!} \partial_\tau^m \tilde{E}^+(\xi;\tau) \\ - i\gamma \left[1 - \frac{2i}{\omega_0}\, \partial_\tau\right]\, \tilde{E}^+(\xi;\tau) \int d\tau_1\, R(\tau_1) |\tilde{E}^+(\xi;\tau-\tau_1)|^2, \quad (2.46)$$

where we have combined the instantaneous electronic response and the Raman response of the medium to

$$R(t) = (1-f)\, \delta(t) + f R_R(t). \quad (2.47)$$

The relative weight is governed by the constant f which for fused silica is approximately 0.18.

Chapter 3

Lasers

First we introduce the numerical structure of a pulse. A virtual laser pulse is a cluster of different data types. (A cluster in `G` is equivalent to a `structure` in `C` or a `record` in `Pascal`). It consists of the following elements.

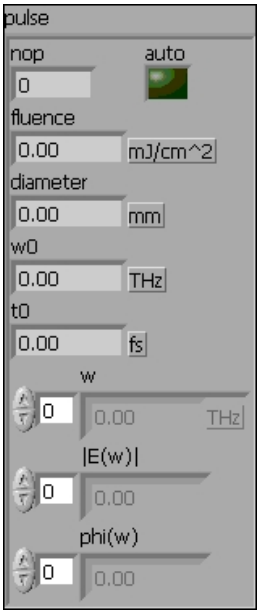


Figure 3.1: Laser pulse cluster.

The first item is the number of sample points and determines how many points are used to represent a pulse in time or frequency space. The value is always a power of 2, which allows to use the FFT-algorithm (fast Fourier transformation). The pulse is encoded in the frequency domain and we only consider positive frequencies. The frequency vector is stored relative to the center frequency ω_0 of the laser pulse. The vector `w` contains the relative frequency vector and the corresponding spectral amplitudes and phases are stored in the vector `|E(`ω`)|` and `phi(`ω`)`. Because the energy per unit area, i.e. the `fluence`, is stored separately the spectral amplitude is usually normalized to a value appropriate for numerical

calculations. The beam diameter is stored in `diameter`. First order phases (time delays) are accumulated in $\mathtt{t}_0$. Using such an encoding allows to consider large delay times with relatively modest sampling rates which would otherwise violate the Nyquist limit.

An important flag is `auto`. It determines whether all simulations are done with a variable or a fixed number of samples. Variable means that at the input of every element the program tries to estimate the ideal number of sampling points needed to propagate the pulse through that element, and resamples the pulse to that ideal sampling grid. Auto mode is switched on or off by appropriate choice of parameters in all source or laser vis.

Note, $\mathcal{LAB}^{\mathrm{II}}$ assumes a uniform intensity profile across the specified beam diameter. That is, the fluence is determined through

$$F_0 = \frac{W}{\pi r^2}, \tag{3.1}$$

with the pulse energy W and the beam radius $r = d/2$. If you assume a Gaussian beam profile

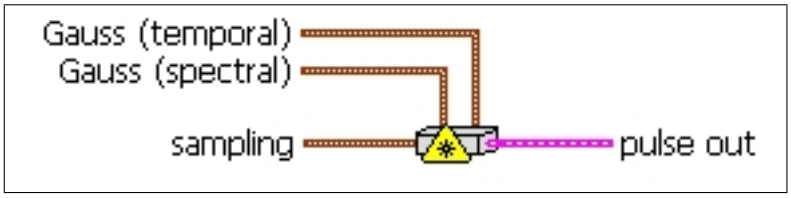
$$F(r) = F_0 \exp\left(-\frac{2r^2}{w_0^2}\right), \tag{3.2}$$

with a beam waist of $w_0 = d/2$ (measured at 13.5%, i.e. $1/\mathrm{e}^2$), the fluence at the beam center ($r = 0$) is

$$F_0 = \frac{2W}{\pi r^2}, \tag{3.3}$$

which is twice as high as for a uniform beam profile. That is to say, if you'd like to perform the simulations at the peak intensity of a pulse with a Gaussian spatial beam profile having a waist of $d/2$, then you have to artificially increase the energy by a factor of two.

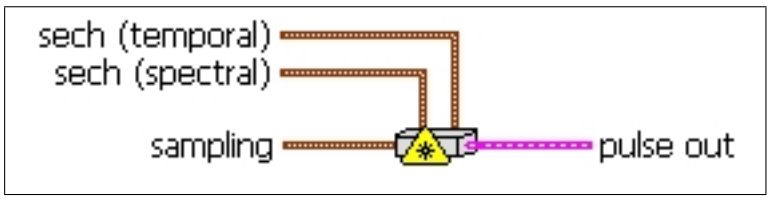
Gaussian pulse



Input		Output	
energy:	energy	pulse out:	laser pulse
beam diameter:	beam diameter		
λ_0:	center wavelength		
$\Delta\lambda$:	spectral FWHM		
$\Delta\tau$:	temporal FWHM		
sampling:	number of samples		
resolution:	sampling mode		

`Gaussian pulse` produces a laser pulse with a Gaussian shaped slowly varying amplitude. The pulse characteristics are either specified in frequency or in time domain (two different clusters). Only one input is required. In both cases one needs to specify the pulse energy, the diameter of the transverse mode profile, and the center wavelength λ_0. In addition, in frequency domain the FWHM of the laser spectrum $\Delta\lambda$, and in time domain the FWHM of the temporal intensity $\Delta\tau$ are mandatory inputs. If nothing else is done, all subsequent calculations run in auto mode, that is, fields are automatically resampled if necessary and iterations are adjusted such that outputs converge to the percent level. In case the sampling cluster is connected all simulations run in manual mode and sampling is determined by the values in this cluster. The number of samples is always a power of two and there are four options for the type of resolution. High spectral: The spectral window is such that the laser spectrum just fits in and the spectral resolution is highest. High temporal: Here, sampling is adjusted so that the temporal resolution is high and the full time window is only slightly larger than the temporal FWHM of the pulse. Of course spectral resolution is very poor here. Balanced: In this mode the sampling is such that both temporal and spectral amplitude are sampled with a somewhat balanced resolution. Time window: Here, you can adjust the time window to a fixed value.

Sech Square Pulse

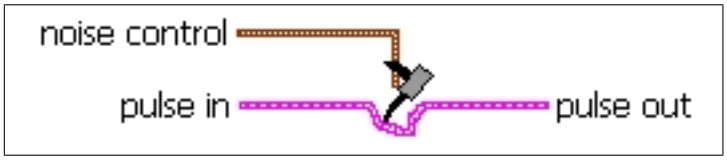


©TF

Input		Output	
energy:	energy	pulse out:	laser pulse
beam diameter:	beam diameter		
λ_0:	center wavelength		
$\Delta\lambda$:	spectral FWHM		
$\Delta\tau$:	temporal FWHM		
sampling:	number of samples		
resolution:	sampling mode		

`Sech Square Pulse` is similar to `Gaussian Pulse`. The only difference is the slowly varying envelope is a sech^2 rather than a Gaussian. The pulse characteristics are either specified in frequency or in time domain (two different clusters). Only one input is required. In both cases one needs to specify the pulse energy, the diameter of the transverse mode profile, and the center wavelength λ_0. In addition, in frequency domain the FWHM of the laser spectrum $\Delta\lambda$, and in time domain the FWHM of the temporal intensity $\Delta\tau$ are mandatory inputs. If nothing else is done, all subsequent calculations run in auto mode, that is, fields are automatically resampled if necessary and iterations are adjusted such that outputs converge to the percent level. In case the sampling cluster is connected all simulations run in manual mode and sampling is determined by the values in this cluster. The number of samples is always a power of two and there are three options for the type of resolution. High spectral: The spectral window is such that the laser spectrum just fits in and the spectral resolution is highest. High temporal: Here, sampling is adjusted so that the temporal resolution is high and the full time window is only slightly larger than the temporal FWHM of the pulse. Of course spectral resolution is very poor here. Balanced: In this mode the sampling is such that both temporal and spectral amplitude are sampled with a somewhat balanced resolution. Time window: Here, you can adjust the time window to a fixed value.

Add noise

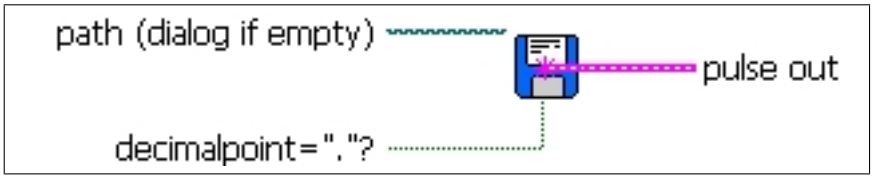


©TF

Input		Output	
pulse in:	input pulse	pulse out:	output pulse
noise control:	type of noise		

`Add noise` adds a certain type of noise to the input laser pulse. Presently the options are `multiply/add amplitude noise` and `multiply/add phase noise`. Either the spectral amplitude or the spectral phase is multiplied with a white noise of specified amplitude. An amplitude of 0.1 for example will add ten percent noise or multiply the existing amplitude by $(1 + 0.1 \cdot \text{noise})$ depending on whether `add ...` or `multiply ...` is selected.

Read Pulse from File



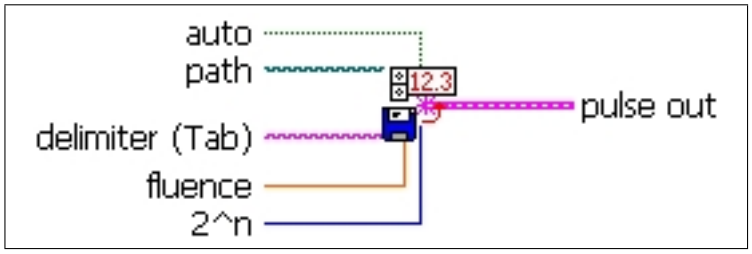
©GS

Input		Output	
path:	path to file	pulse out:	laser pulse
decimalpoint='.'?:	set to false for ',' instead of '.'		

`Read Pulse` reads a pulse from a file. The format is as follows:

`nop=...`	number of samples
`auto=...`	simulation mode
`w0[THz]=...`	center frequency
`fluence[mJ/cm`^{2}`]=...`	energy per unit area
`diameter[mm`^{2}`]=...`	beam diameter
`t0[fs]=...`	first order phase
`frq E(amp) phi(w)`	frequency, amplitude, residual phase
... ...	

Read Pulse from Spreadsheet



©GS

Input		Output	
path:	path to file	pulse out:	laser pulse
auto:	simulation mode		
delimiter:	tab (default), space, etc.		
fluence:	fluence value		
2^n:	number of samples		

`Read Pulse from Spreadsheet` reads a pulse from a spreadsheet file. The format is equivalent to that used by the FROG program (©Femtosoft). The first column gives the wavelength in nm, the second the spectral intensity in units of J/(m^2 nm), the third column the phase, the fourth column the real part, and the last column the imaginary part of the spectral field in units of V/(m nm). Obviously, there is too much information here, but we decided to simply use the format of the FROG software in order to facilitate the import of retrieved pulses. Pulses are also correctly loaded if the last two columns are missing. If the third column is missing the phase is assumed to be zero for all wavelengths.

If `auto` is true the the spectrum is automatically sampled, if false, then the number of samples must be specified and the loaded spectrum is interpolated accordingly. If the fluence is specified then the amplitude of the electric field is recalculated such the the fluence is equal to the value specified.

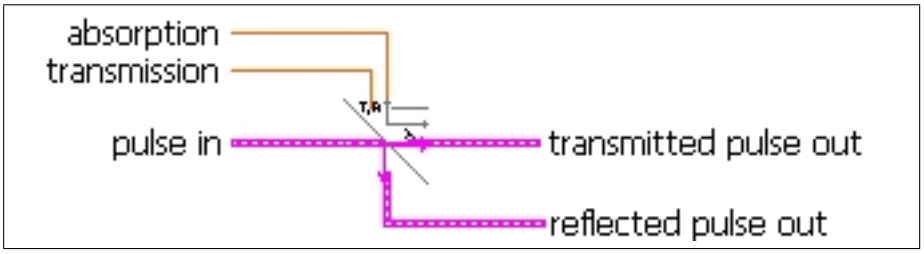
Chapter 4

Linear Elements

All elements have at least one control panel that you need to connect. It allows you to control the important parameters for the simulation and it also allows to change things and watch in real time how the laser pulse intensity or the spectrum or whatever you are looking at changes.

First there are a number of standard elements, such as a delay, a telescope, etc., nothing of great importance but sometimes they come in handy. Then there are a number of linear elements, mostly complex optical setups which are designed to stretch or compress short pulses. In addition, all of them have a little intelligent brother that automatically adjusts some parameters in order to optimize for maximum peak intensity at the output. Also you'll find most standard glass materials.

Ideal Beamsplitter



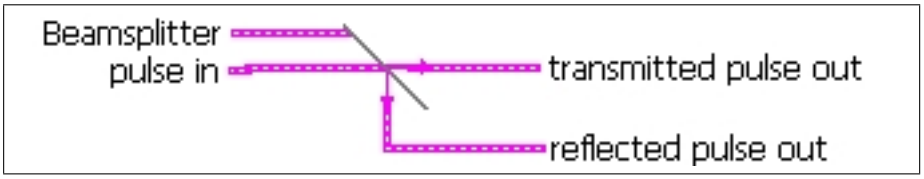
©MH

Input		Output	
pulse in:	input laser pulse	transmitted pulse:	transmitted laser pulse
absorption:	absorption $[0 \ldots 1]$	reflected pulse:	reflected laser pulse
transmission:	transmission $[0 \ldots 1]$		

The `ideal beamsplitter` is used to split the incoming beam in two outgoing replica. The transmission $T(0 \ldots 1)$ as well as the absorption $A(0 \ldots 1)$ are wavelength independent and the reflectivity is $R = 1 - A - T$. Make sure the sum of $A + T$ is always lower than 1.

A real beamsplitter where the transmitted beam actually passes through some dispersive material is easily simulated by introducing a piece of dispersive material in the transmitted beam path.

Beamsplitter



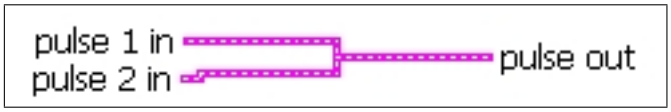
©MH

Input		Output	
pulse in:	input laser pulse	transmitted pulse:	transmitted laser pulse
beamsplitter:	file name	reflected pulse:	reflected laser pulse

The `beamsplitter` is a more realistic version of the module `ideal beamsplitter`. You may specify a file that lists the reflectivity as well as the transmission as a function of the wavelength. This file should have the extension `\*.bs` and must be located in the folder `..\beamsplitters\`. The files are text files and consist of three columns. The first is the wavelength in units of meter, the second the reflectivity, and the third the transmission. If you only specify a few values the module will interpolate in order to determine all necessary values. The absorption is calculated from the transmission and the reflectivity values. The following table shows an example of a low-pass filter with a cutoff wavelength at 800 nm. Below 800 nm it is fully transparent and above 800 nm completely opaque.

wavelength [nm]	reflectivity	transmission
2.0000E-7	0	1
3.0000E-7	0	1
4.0000E-7	0	1
5.0000E-7	0	1
6.0000E-7	0	1
7.0000E-7	0	1
7.9900E-7	0	1
8.0100E-7	1	0
9.0000E-7	1	0
1.0000E-6	1	0
1.1000E-6	1	0

Beamcombiner

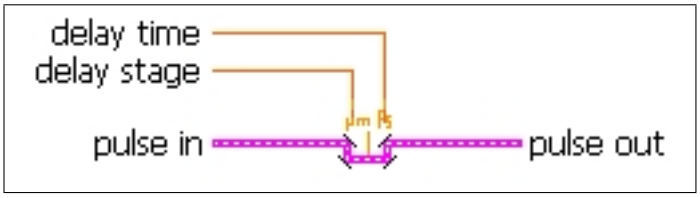


©TF

Input		Output	
pulse A in:	first incoming pulse	pulse A+B out:	combined pulse
pulse B in:	second incoming pulse		

The `beamcombiner` is designed to act as a beamsplitter that is used to combine two laser pulses. Of course in a collinear fashion because all calculations are one-dimensional. The two input pulses may have different center wavelengths, spectral bandwidths, or even a time delay with respect to each other. None of the two pulses looses energy which of course is not true for a real beamsplitter. In case the two beams have different diameters the diameter of the second beam is adjusted to match that of the first beam, that is, when the two beams are physically combined they end up having the same diameter. Also note that the combination is non-interferometric, which in practice means the the two beams are not really collinear but have a small angle.

Delay

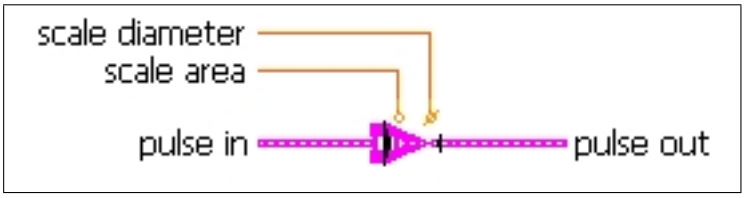


©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	delayed pulse
delay [fs]:	temporal delay		
delay stage:	mimics a mechanical delay stage		

The `delay` simulates a mechanical delay line. The delay may be specified in terms of the time delay or in terms of the optical path the beam has to propagate. If you enter the optical path the delay corresponds to the time it takes to travel the path back and forth just as it is the case for a real mechanical delay.

Telescope

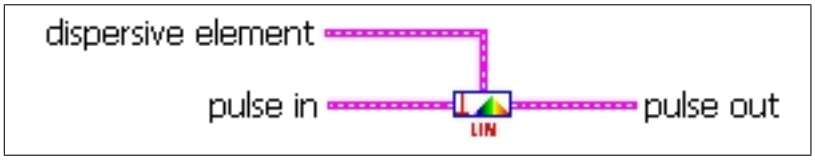


©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
scale diameter:	relative change of diameter		
scal area:	relative change of pulse area		

`telescope` simulates a telescope. It does nothing but to increase or decrease the fluence (energy per unit area) corresponding to the magnification you choose. This you may do in two different ways. First by specifying the magnification of the beam diameter or second of the beam area. A value larger than one increases the beam diameter and vice versa.

Transparent Dispersive Element



©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
dispersive element:	specify the glass material and thickness	phi2	second order phase

The `transparent linear medium` is usually just a piece of some glass material. This may be a vacuum window, a lens, or by combining two of those elements also an achromatic lens or more complicated arrangements may be simulated.

The phase change introduced by just a piece of transparent but dispersive material of a given length is

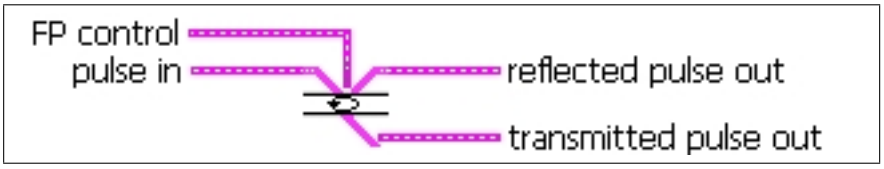
$$\Phi(\omega) = \frac{\omega}{c_0}\, n(\omega) z \tag{4.1}$$

where $n(\omega)$ is the index of refraction and z the thickness of the material. The coefficients of the Taylor series are easily calculated

$$\partial_\omega^k \Phi\big|_{\omega_0} = \frac{z}{c_0}\left[k\, \partial_\omega^{k-1} n\big|_{\omega_0} + \omega_0\, \partial_\omega^k n\big|_{\omega_0}\right] \tag{4.2}$$

To simulate such a material the index of refraction must be known. Usually it is tabulated or given as a Sellmeier equation. In $\mathcal{LAB}^{\mathrm{II}}$ many glasses have been implemented through the corresponding Sellmeier equation. The most prominent ones are fused silica (SQ1), BK7 and SF10 etc. You may just copy and paste any of the files under a new name (the material you wish to add) in the folder `../chi1/` and change the parameters accordingly. It is that easy to add a new material to the list.

Fabry-Perot Interferometer



©MH

Input		Output	
pulse in:	incoming laser pulse	pulse reflected:	reflected laser pulse
R1:	reflectivity of front surface	pulse transmitted:	transmitted laser pulse
R2:	reflectivity of back surface		
material:	material between front and back side		
d:	material thickness		
alpha:	angle of incidence		

The `Fabry Perot interferometer` consists of some material with two partially reflective coatings on both sides. The material may be air or any of the transparent materials available in the data base. The coatings are assumed to be ideal and impose no additional phase on the pulse.

The amplitude and the phase change for the transmitted pulse E_t and the reflected pulse E_r introduced by the Fabry-Perot interferometer are

$$E_t = E_i \left[t_1 t_2 \frac{\exp(i\delta/2)}{1 - r_1 r_2 \exp(i\delta)} \right] \tag{4.3}$$

$$E_r = E_i \left[\frac{r_1 - r_2 \exp(i\delta)}{1 - r_1 r_2 \exp(i\delta)} \right] \tag{4.4}$$

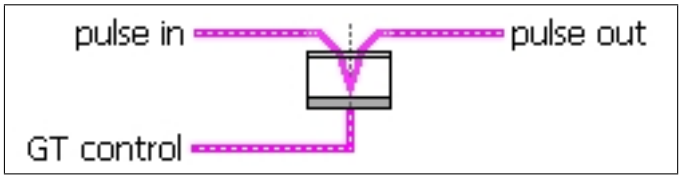
where E_i is the incoming laser pulse, t_1 and t_2 are the transmission coefficients for the transitions from air to the medium and the medium to air, respectively. The amplitude reflection coefficients are $r_1 = \sqrt{R_1}$ and $r_2 = \sqrt{R_2}$.

$$\delta = 2d \frac{n\omega}{c_0} \cos\theta \tag{4.5}$$

$$\sin\theta = \frac{1}{n} \sin\alpha \tag{4.6}$$

where n is the index of refraction of the medium between the two interfaces, ω is the laser frequency, c_0 the speed of light in vacuum, and θ the angle of propagation in the medium.

Gires-Tournois Interferometer



©MH

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
reflectivity:	reflectivity of front side		
material:	medium between front and back side		
d:	thickness of the medium		
alpha:	angle of incidence		

The `Gires Tournois interferometer` is a special case of the `Fabry Perot` with the back side being 100% reflective.

The phase change introduced by the Gires Tournois interferometer is

$$\Phi(\omega) = \arctan\left[\frac{(R-1)\sin\delta}{2\sqrt{R}-(R+1)\ \cos\delta}\right] \tag{4.7}$$

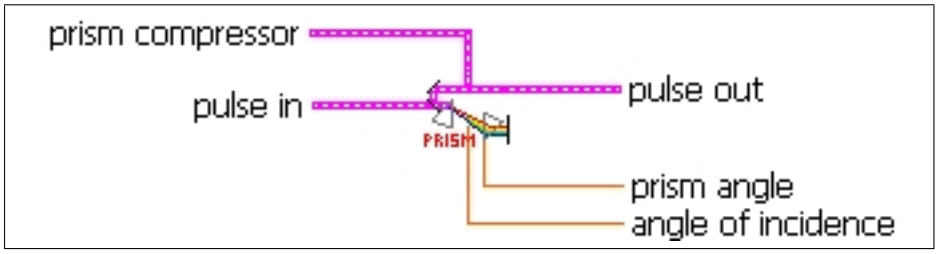
where $R = r^2$ is the (intensity) reflectivity of the front surface. The angle δ is

$$\delta = 2d\,\frac{n\omega}{c_0}\,\cos\theta \tag{4.8}$$

$$\sin\theta = \frac{1}{n}\,\sin\alpha \tag{4.9}$$

where n is the index of refraction of the medium between the semitransparent and the fully reflective surface, ω is the laser frequency, c_0 the speed of light in vacuum, and θ the angle of propagation in the medium.

Prism Compressor



©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	outgoing laser pulse
Brewster angle:	automatic alignment	phi2:	2nd order spectral phase
Min. deviation:	automatic alignment	prism angle:	calculated prism angle
exact/approx.:	exact ray tracing or approximate solution	angle of inc.:	calculated angle of incidence
prism angle:	adjust prism angle		
prism base:	base size of both prisms		
material:	prism material		
angle of inc.:	adjust angle of incidence		
distance tip-tip:	distance between the prism tips		
prism-mirror:	distance prism 2 and mirror		
in prism 1:	point of incidence prism 1		
in prism 2:	point of incidence prism 2		

In the case of more complex optical systems the situation is somewhat more complicated. However, in most cases the problem turns out to be a geometric one, in a sense that one needs to find the optical path length as a function of frequency for a given optical system. One may even use ray tracing programs to do that job.

A first widely used optical system is a prism compressor. It consists of four prisms or more frequently of two prisms and a folding mirror. Figure 4.1 shows a typical geometry of a prism compressor. The beam impinges on the first prism, gets dispersed, passes the second prism, is reflected at the folding mirror and goes all the way back. In order to separate the output from the incoming beam the folding mirror is usually tilted in the vertical dimension, such that the output beam is on top of the incoming or vice versa. Alternatively, a roof mirror may be used. Both prisms are identical and their apex angle is α. The index of refraction is determined by the corresponding Sellmeier equation.

Usually a prism compressor allows to adjust both positive and negative chirps depending on the distance between the two prisms if the input pulse was bandwidth limited. That is,

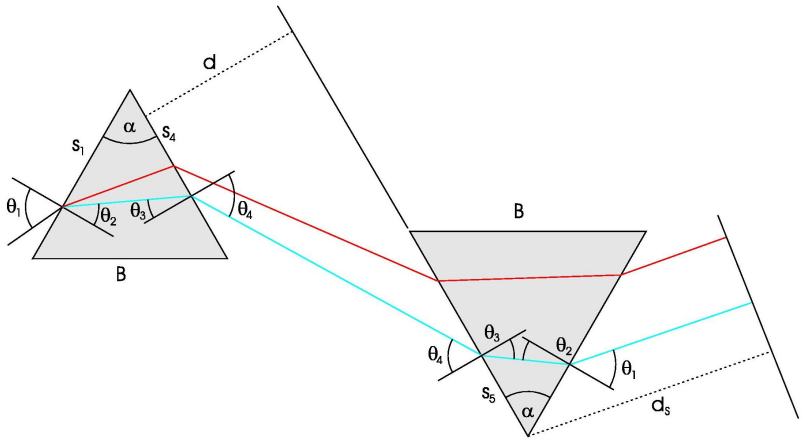


Figure 4.1: Prism compressor.

because the glass material of the prisms itself introduces some positive phase modulation and only if the distance between the prisms exceeds a certain threshold the overall phase modulation becomes negative. Therefore, one may 'fine tune' a prism compressor by moving one of the prisms perpendicular to its base in order to introduce slightly more or less glass.

Spectral clipping occurs whenever the spectrum at the second prism is larger than the prism itself. In other words spectral clipping happens when one of the two following equations is true

$$s_5(\omega_{max}) < 0, \tag{4.10}$$

for the highest frequency and

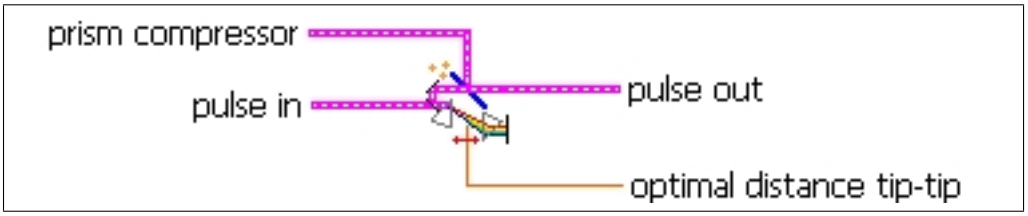
$$s_5(\omega_{min}) > \frac{B}{2\sin(\alpha/2)}. \tag{4.11}$$

for the lowest frequency, where B is the length of the prisms base and s_5 is the distance from the tip of the second prism to the point where a specific frequency enters the second prism.

If you activate the button `use Brewster angle` the VI automatically adjusts the input angle at the first prism such that the center frequency enters the first prism and also the second prism at Brewsters angle. The other choice you have is to specify `use minimum deviation` which causes the apex angle of the prism to be adjusted such that the center frequency experiences minimal deviation, or in other words it passes the first prism parallel to its base.

There are two choices to calculate the compressor phase. First, through exact ray tracing and second through an approximate solution which may be found in any textbook.

Intelligent Prism Compressor

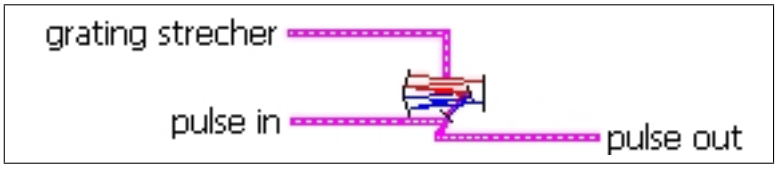


©GS

Input		**Output**	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
Brewster angle:	automatic alignment	optimal prism	
Min. deviation:	automatic alignment	separation:	optimized value
exact/approx.:	exact ray tracing or approximate solution	prism angle:	calculated prism angle
prism angle:	adjust prism angle	angle of inc.:	calculated angle of incidence
prism base:	base size of both prisms		
material:	prism material		
angle of inc.:	adjust angle of incidence		
distance tip-tip:	distance between the prism tips		
prism-mirror:	distance prism 2 and mirror		
in prism 1:	point of incidence prism 1		
in prism 2:	point of incidence prism 2		

The `intelligent prism compressor` is nothing else but a standard `prism compressor` which automatically adjusts the distance between the two prisms such that the second order phase present in the input pulse vanishes and the peak power is maximized.

Grating Stretcher



©GS

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
lines/mm:	grating constant	phi2:	2nd order spectral phase
angle of incidence:	refers to first grating		
focal length:	of the two mirrors or lenses		
g:	displacement (see text)		
grating-mirror:	distance from grating 2 to end mirror		
grating size:	width of both gratings		
diffraction order:	± 1		

Grating stretchers are found in many different variations. Here we only implemented the two most commonly used consisting of a single grating, one or two spherical mirrors, and a flat end mirror.

Fig. 4.2 shows the definition of the angles and when they are positive and when negative

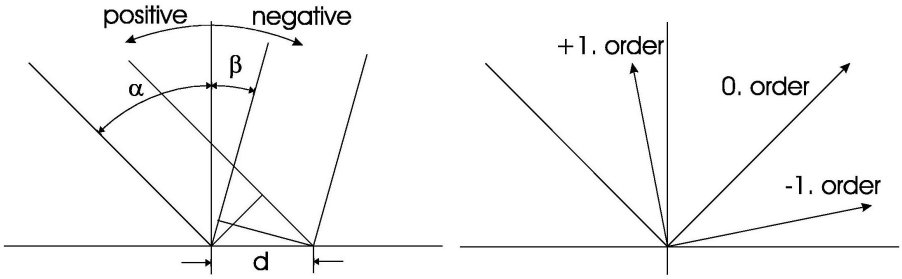


Figure 4.2: Definition of the angles.

In the same manner we define the diffraction orders left of the zero order positive and to the right hand side of zero order negative. The grating equation is

$$\beta = \arcsin\left(M\frac{2\pi c_0}{\omega d} - \sin\alpha\right) \tag{4.12}$$

where M is the diffraction order and d the grating constant.

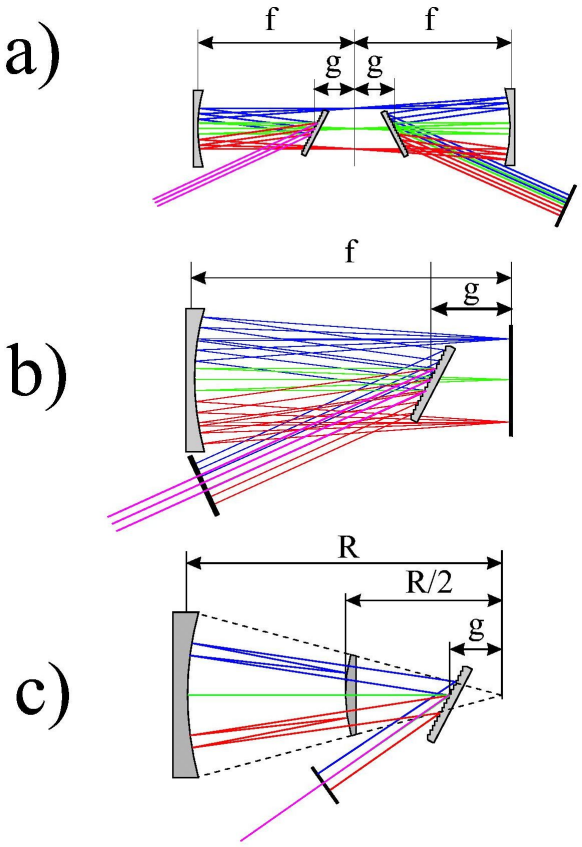


Figure 4.3: a) Stretcher geometry. b) If the setup a) is folded with respect to the plane between the two focusing mirrors, the most commonly used stretcher geometry emerges. c) Second, Oeffner-triplet: This setup is aberration-free and another commonly used stretcher. R and $R/2$ are the radii of curvature of the two spherical mirrors.

Figure 4.3 shows the two most commonly used stretcher geometries. The overall stretching factor depends on the grating dispersion, the angle of incidence and the displacement g. Using the more detailed graph shown in fig. 4.4 allows to calculate the phase modulation as a function of the frequency

$$\Phi(\omega) = \frac{\omega}{c_0}\left[8f + 2r - 4g\frac{\cos\beta_0 - \sin(\beta_0 - \beta)\sin\alpha}{\cos\beta}\right] + \frac{8\pi g\cos\beta_0}{d}\ \tan\beta \tag{4.13}$$

where d is the grating constant, α the angle of incidence on the grating, β the diffraction angle (which is negative), and β_0 the diffraction angle (also negative) at the center wavelength. The diffraction order in this case is +1. The same formula may be used for the Oeffner triplet if the focal length f is replaced by $(R_1 + R_2)/2 = \frac{3}{4}R$. From the phase we may calculate the second and third order phase term by calculating the second and third derivative with respect to frequency

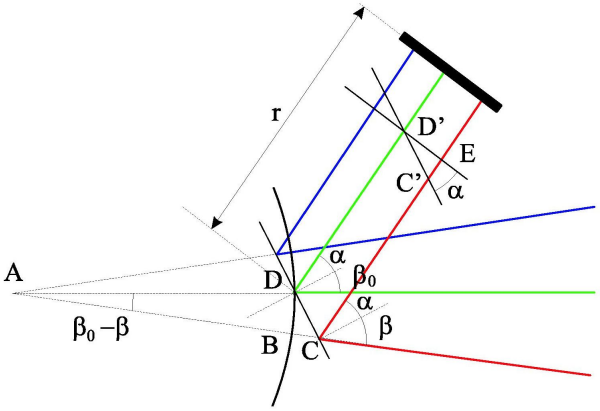


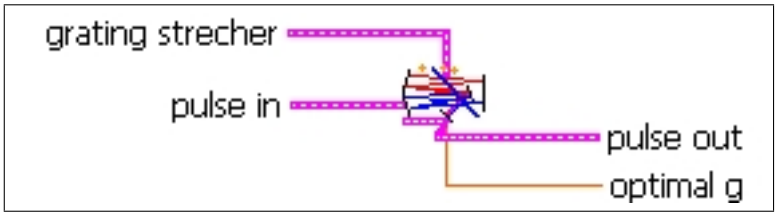
Figure 4.4: Detailed stretcher geometry.

$$\Phi_2 = \frac{2g\lambda_L}{\pi c_0^2}\left(\frac{\lambda_L}{d}\right)^2 \frac{1}{\cos^2\beta_0} \tag{4.14}$$

$$\Phi_3 = -\frac{3g\lambda_L^2}{\pi^2 c_0^3}\left(\frac{\lambda_L}{d}\right)^2 \frac{1}{\cos^2\beta_0}\left[1+\frac{\lambda_L}{d}\frac{\sin\beta_0}{\cos^2\beta_0}\right] \tag{4.15}$$

Obviously, the second order dispersion is positive and that the third order dispersion is negative.

Intelligent Grating Stretcher

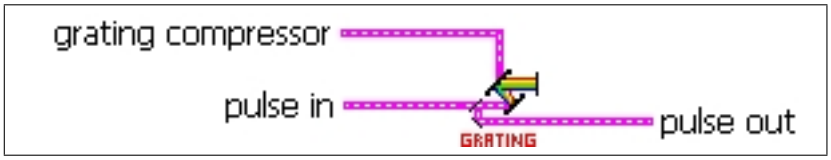


©GS

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
lines/mm:	grating constant	optimal g:	optimal displacement g
angle of incidence:	refers to first grating		
focal length:	of the two mirrors or lenses		
g:	displacement (see text)		
grating-mirror:	distance from grating 2 to end mirror		
grating size:	width of both gratings		
diffraction order:	± 1		

The `intelligent grating stretcher` is nothing else but a standard `grating stretcher` that adjusts automatically the displacement g such that the second order phase present in the input pulse is minimized and the peak intensity of the out going pulse is maximal.

Grating Compressor



©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
lines/mm:	grating constant	phi2:	2nd order spectral phase
angle of incidence:	refers to first grating		
delta:	distance between gratings measured at center frequency		
grating-mirror:	distance from grating 2 to end mirror		
grating size:	width of both gratings		
diffraction order:	± 1		

Again let's start defining all relevant parameters as shown in fig. 4.5. We split the whole optical path in two parts. The first represents the wavelength dependent distance between the two gratings and the second the distance from the second grating to the end mirror. The total phase is two times the sum of the two parts plus a relative phase term

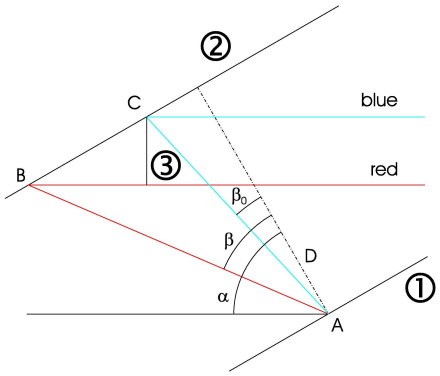


Figure 4.5: Grating compressor.

$$\Phi(\omega) = \frac{2\omega}{c_0}\left\{\frac{D}{\cos\beta} + \overline{B_0C} + D(\tan\beta - \tan\beta_0)\sin\alpha\right\} - \frac{4\pi D}{d}\tan\beta \tag{4.16}$$

where $\overline{B_0C}$ is the distance between the second grating and the end mirror along the

center wavelength. Again we assume +1st order diffraction.

We again may calculate the second and third order Taylor coefficients

$$\Phi_2 = -\frac{D\lambda_L}{\pi c_0^2}\left(\frac{\lambda_L}{d}\right)^2 \frac{1}{\cos^3\beta_0} \tag{4.17}$$

$$\Phi_3 = \frac{3D\lambda_L^2}{2\pi^2 c_0^3}\left(\frac{\lambda_L}{d}\right)^2 \frac{1}{\cos^3\beta_0}\left[1+\frac{\lambda_L}{d}\,\frac{\sin\beta_0}{\cos^2\beta_0}\right] \tag{4.18}$$

which is just the opposite of the grating stretcher if we set

$$D = 2g\cos\beta_0 \tag{4.19}$$

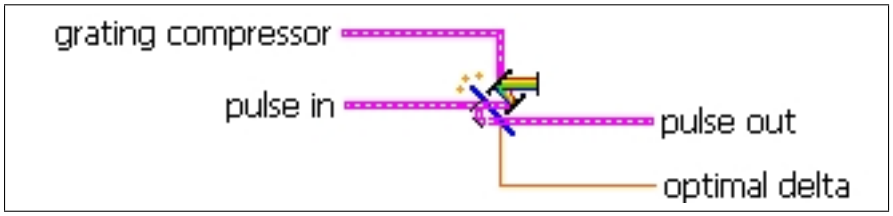
So, in principle a grating compressor should be able to fully compensate for the phase modulation introduced by a grating stretcher.

Finally, we also consider spectral clipping that occurs whenever the second grating is too small. We always assume that the center frequency impinges on the center of the second grating. The effective transmission therefore is

$$T(\omega) = \begin{cases} 0 & : \quad \left|\frac{D\sin(\beta-\beta_0)}{\cos\beta_0\cos\beta}\right| \geq \frac{S}{2} \\ 1 & : \quad \left|\frac{D\sin(\beta-\beta_0)}{\cos\beta_0\cos\beta}\right| \leq \frac{S}{2} \end{cases} \tag{4.20}$$

where S is the lateral size of the grating.

Intelligent Grating Compressor

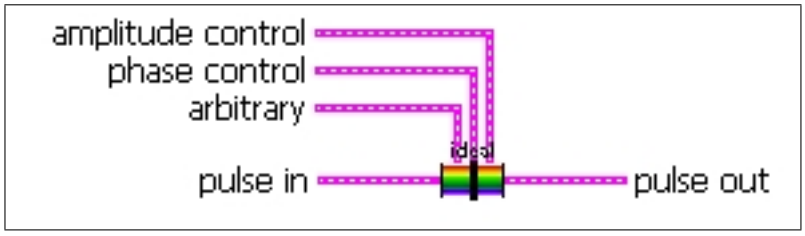


©GS

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
lines/mm:	grating constant	optimal delta:	optimal grating separation
angle of incidence:	refers to first grating		
delta:	distance between gratings measured at center frequency		
grating-mirror:	distance from grating 2 to end mirror		
grating size:	width of both gratings		
diffraction order:	± 1		

The `intelligent grating compressor` is nothing else but a standard `grating compressor` that adjusts automatically the distance between the two gratings such that the second order phase present in the input pulse is minimized and the peak intensity of the out going pulse is maximal.

Ideal Shaper



Input		Output	
pulse in:	incoming laser pulse	pulse out:	output laser pulse
amplitude control:	specify amplitude modulation		
phase control:	specify phase modulation		
arbitrary:	array with frequency, phase and amplitude modulation		

The `ideal shaper` simulates an ideal pulse shaper which modulates the spectral amplitude and/or phase. This shaper shows no pixelation effects. There are a number of options available for amplitude and phase shaping:

- **none:** No amplitude and/or phase modulation is applied.
- **interpolate:** The amplitude and/or phase modulation specified in the array `arbitrary` is used. The array contains three columns, i.e. the absolute frequency, the amplitude-, and the phase modulation. The frequency axis must not necessarily coincide with the internally used sampling of the pulse. The amplitude and phase values are resampled by interpolation to match the internal sampling of the pulse. The frequency vector must contain absolute frequencies.
- **arbitrary:** The amplitude and/or phase modulation specified in the array `arbitrary` is used. The number of amplitude and/or phase values MUST coincide with the number of samples of the incoming pulse. The frequency vector is not considered at all as no interpolation is performed. Assume the pulse is sampled with 256 individual samples, then the array amplitude has to contain 256 values which then are used to modulate the 256 amplitude values of the incoming pulse.
- **Taylor:** Only the phase modulation can be specified in terms of Taylor coefficients. The units of the n-th order coefficient is fs^n.
- **Sinusoidal:** The amplitude and/or phase of the pulse is modulated with a sinusoidal function.

- **Square:** The amplitude and/or phase of the pulse is modulated with a periodic square function of amplitude
- **Sawtooth:** The amplitude and/or phase of the pulse is modulated with a periodic sawtooth function of amplitude

For sinusoidal, square and sawtooth the parameters are: amplitude A, offset Δ, periodicity dt and phase ϕ. The modulations are calculated according to

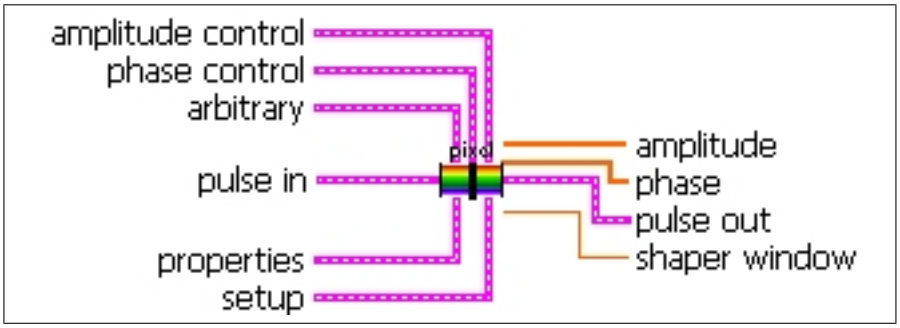
$$f(\omega) = A \sin\left(dt(\omega - \omega_0) + \phi\right) + \Delta,$$

where f stands for either amplitude or phase modulation and sin may be replaced by the square or sawtooth function. A negative amplitude is converted to a phase value keeping in mind that $-1 = \mathrm{e}^{i\pi}$.

$$f(\omega) = \begin{cases} |f(\omega)|\mathrm{e}^{i\pi} & f(\omega) < 0 \\ |f(\omega)| & f(\omega) \geq 0 \end{cases}$$

The cluster `arbitrary` contains three columns, the absolute frequency (with units), the amplitude modulation (a modulation of one leaves the original frequency component unchanged), and the phase modulation in radians.

Pixelated Shaper



©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output pulse
amplitude control:	type of amplitude modulation	shaper window:	full spectral window
phase control:	type of phase modulation		
arbitrary:	arbitrary amplitude and phase	amplitude:	applied amplitude modulation
properties:	shaper properties		
setup:	setup geometry	phase:	applied phase modulation

This pulse shaper has been designed to simulate a real laboratory setup as realistic as possible. The amplitude and phase modulation controls are identical to those of the `ideal shaper`, except the selection `arbitrary` as type of amplitude or phase modulation is ignored. Note, new here is two additional control panels, i.e. the properties and the setup control.

`Setup` allows to choose between two alternative experimental schemes. First, a zero dispersion compressor consisting of two gratings and two lenses (or two curved mirrors), and second, a zero dispersion compressor where the gratings are replaced by prisms. In both cases you need to specify the focal length of the two lenses (or the two curved mirrors). When you pick the grating setup you need to specify the number of lines/mm, the diffraction order, and the difference between the angle of incidence and the diffraction angle at the center wavelength. If you pick the prism setup you need to specify the prism material and the prism angle (apex). From this information the program calculates the correspondence between pixel and frequency. In addition to the zero dispersion compressor geometry you need to provide information on the pixelated modulator itself. Here, the program needs to know the number of pixels, the pixel width, and the wavelength of the center pixel.

`Properties` allows to switch on or off various effects connected to the pixelated nature of the modulator device. In total there are 6 options:

pixelated: This switch allows to turn on or off all effects related to pixelation. This way you may find out which experimentally observed details in the shaped waveform are due to the pixelated nature of the modulator.

gaps: If switched on, two neighboring pixels are separated by a gap. The width of the gap is in units of pixel width. For example 0.04 means that the gap width is 0.04 times the pixel width. The phase modulation of all gaps is zero and the amplitude modulation is equal to 1.

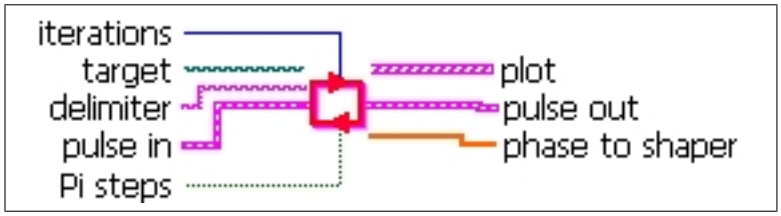
wraps: Most pixelated devices have a maximum accessible range of phase values, typically a few times 2π. Mathematically we don't need a phase larger than 2π, because 3π for example leads to the same result as 3π modulus $2\pi = 1\pi$. That is, whenever the phase modulation exceeds 2π we may subtract 2π; this process is known as phase wrapping. You may specify the wrap point in terms of 2π, that is, wraps may occur whenever the phase exceeds 2π (1) or 4π (2) etc.

crosstalk pixel: If turned on the amplitude and phase pattern are convoluted with a Gaussian. The width is given in units of pixel. With this option you may simulate a misaligned setup where the modulator is not exactly in the focal point of the lens/curved mirror.

diffraction: If turned on the phase and amplitude modulation pattern is convoluted with the spatial resolution of the optical system. The system resolution is assumed to be Gaussian and the width is determined by the beam diameter and the focal length of the lens/curved mirror according to Fourier optics.

frequency mapping: You may select between `linear frequency mapping` and `geometric`. If you select the first, then the relation between pixel number and frequency is assumed to be linear. With this you may simulate what happens if you do not calibrate your setup correctly. If you select the latter, the frequency mapping is calculated from the optical arrangement and everything is as it should be.

Iterative FFT Shaper



©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output pulse
iterations:	the number of iterations	phase to	
Pi steps:	use π phase jumps only	shaper:	desired spectral phase
target:	file name	plot:	target and output pulse
delimiter:	tab (default), space, etc.		

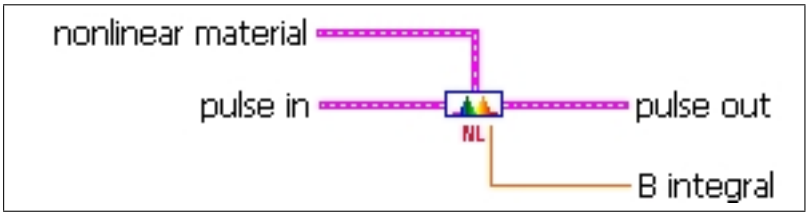
The `iterative FFT shaper` module allows to specify target pulse shape, i.e. the temporal intensity as a function of time. It uses an iterative Fourier transform algorithm to evaluate the phase pattern necessary to transform any input intensity as close as possible to the desired target intensity by phase-only shaping. The number of iterations must be specified. The target intensity profile must be available as a file in the folder `../iffttargets/`. The file contains two columns separated by a delimiter. The first column is the time in femtoseconds and the second the intensity. If the time samples do not coincide with the time samples of the pulse the target will be resampled automatically.

Chapter 5

Nonlinear Elements

The nonlinear elements implemented in $\mathcal{LAB}^{\mathrm{II}}$ allow to simulate the most commonly found nonlinear effects. There is self-phase modulation which appears in almost any material when the intensity gets high enough and there are frequency mixing processes. Because the spectral width of the laser pulses is very broad the pulse spectra have to be convoluted which sometimes leads to counterintuitive results. For example, it is not always so easy to predict what the second harmonic spectrum might look like if the fundamental pulse has some funny shaped spectral phase. Peter Blattnig has programmed the optical fiber and most of the demo vi's which illustrate some important effects. Fred van Goor from the University of Twente has used the amplifier vi to simulate a number of published multi-pass amplifier designs. He generously agreed to let us include his results in the present manual.

Transparent Nonlinear Medium



©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output pulse
glass:	material	phi2:	2nd order spectral phase
z:	thickness	B-integral:	B integral

A `transparent nonlinear medium` has the same linear dispersive properties as the previously described `linear transparent medium`. In addition, it handles nonlinear effects due to self-phase modulation (SPM). The numerical simulation uses a split step Fourier transform algorithm. Whereas dispersive effects are incorporated in the spectral domain, SPM is treated in the time domain. The integration z-step width is dynamically adjusted so that the nonlinear phase per integration interval is always $\pi/20$, i.e.

$$\Delta z = \frac{\pi}{20} \frac{c}{\omega_c n_2 I_{\max}}. \tag{5.1}$$

The output B-integral yields

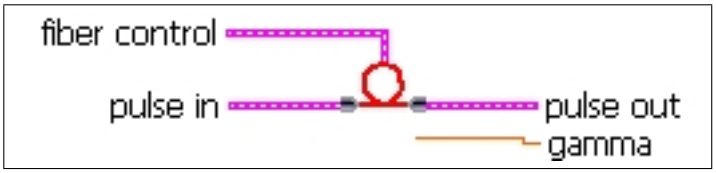
$$B = \frac{2\pi}{\lambda_0} \int \mathrm{d}z \; n_2 I(z) \tag{5.2}$$

The linear material properties come from material files in the folder `../chi1/`. The `transparent nonlinear medium` requires nonlinear material properties and those are stored in the folder `../chi3/`. The material files in this folder provide the nonlinear index of refraction commonly known as n_2 plus the Raman response. Here, we only need n_2. As an example we inspect fused silica (SQ1). To simulate the propagation of a short pulse through a piece of silica we need two files:

1. `../chi1/SQ1.vi` for the linear material properties and
2. `../chi3/X3_SQ1.vi` for the nonlinear material properties.

If you wish to add a new material, make sure you know the nonlinear index of refraction. Then copy the file `X3_SQ1` for example to `X3_SF6`, edit the material constants, and you're done.

Optical Fiber



©PB

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output pulse
fiber control:	fiber properties	gamma:	nonlinear constant

Even more sophisticated than the `transparent nonlinear medium` is the `optical fiber`. Besides linear dispersion and self-phase modulation it also accounts for self-steepening and the Raman effect. The fiber control allows you to specify the following values:

dispersion: Here you need to specify the material which is responsible for the dispersive properties of the fiber. This may be a simple material file, such as fused silica (SQ1) or it may be a file which contains the dispersion properties of a specific complex photonic bandgap fiber.

nonlinear: The nonlinear properties may be specified independent of the linear properties. This allows to use a fused silica core (SQ1) which specifies the nonlinear properties together with any dispersion file which is dominated for example by the hole structure surrounding the core.

length: The fiber length.

effects: Here you may switch on or off linear or nonlinear effects in order to study their influence for example on super-continuum generation.

The integration step width is dynamically adjusted so that the nonlinear phase never exceeds a specified limit. The nonlinear constant is calculated to

$$\gamma = \frac{\omega_0 n_2}{c A_{\text{eff}}} \tag{5.3}$$

where ω_0 is the center frequency and A_{eff} is the effective area, calculated from the beam diameter d simply through $A_{\text{eff}} = \pi d^2/4$. The vi numerically integrates the nonlinear differential equation for the slowly varying electric field envelope

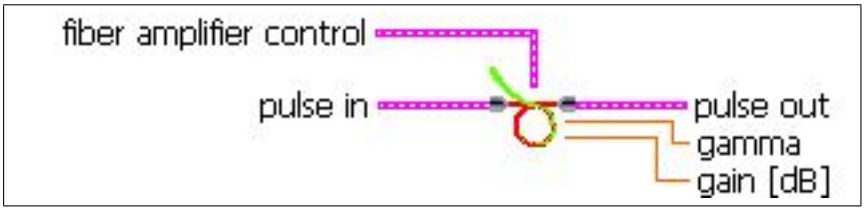
$$\frac{\partial \mathcal{E}(\xi;\tau)}{\partial \xi} = -i \sum_{m=2}^{\infty} \frac{k_m}{i^m\, m!} \frac{\partial^m \mathcal{E}(\xi;\tau)}{\partial \tau^m} - i\gamma \left[1 - \frac{2i}{\omega_c}\,\frac{\partial}{\partial \tau}\right] \mathcal{E}(\xi;\tau) \int d\tau_1\, R(\tau_1) |\mathcal{E}(\xi;\tau-\tau_1)|^2, \tag{5.4}$$

where we have combined the instantaneous electronic response and the Raman response of the medium to

$$R(t) = (1 - f)\,\delta(t) + f R_R(t). \tag{5.5}$$

The relative weight is governed by the constant f which for fused silica is approximately 0.18. If you switch off `dispersion` then the sum (first term on the right hand side) will be set to zero. If you switch of the Raman effect then f is set to zero. If you switch off `self steepening` then $\frac{2i}{\omega_c}$ is set to zero and if you switch off `SPM` then γ is set to zero.

Optical Fiber Amplifier



©TF

Input		Output	
pulse in:	incoming laser pulse	pulse out:	output pulse
fiber amplifier control:	fiber properties	gamma:	nonlinear constant
		gain [dB]	overall gain in dB

The vi `optical fiber amplifier` adds to the vi `optical fiber` the option to account for gain. The fiber control is identical to that of the `optical fiber`. In addition you find two more control clusters, where one specifies the gain medium and the other details on the inversion profile.

Besides specifying the gain medium and the dopant concentration in weight percent, one has the option to switch on or off gain saturation or gain dispersion effects. For each gain medium there must exist a file containing the necessary data. If you'd like to add a new gain medium the best thing is to copy an existing one and change the numbers appropriately.

Details on the inversion profile are first of all the longitudinal inversion profile $\Delta N(z)$. You may select

constant: In which case the inversion profile is assumed to be constant.

exponential: The gain profile decays exponentially with the decay constant specified by α.

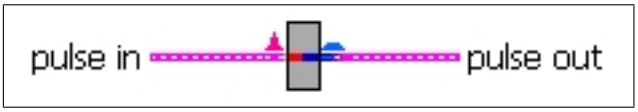
Fermi-Dirac: The gain profile resembles a Fermi-Dirac distribution in order to account for saturation in the pumping process. z_0 marks the position to which the gain is roughly constant and after which it decays exponentially. Again α determines the decay constant.

from file: This option allows to load an arbitrary inversion profile from a file which has two columns separated by a tab. The first column is the longitudinal position scaled by the fiber length. That is, the first column contains numbers between zero and one which must be evenly spaced. The second column contains the upper state population in arbitrary units.

No matter what profile you pick the absolute values are automatically adjusted so that the overall small signal gain at the center wavelength of the pulse is as specified by the

control `g0 [dB]`. It may happen that the simulated small signal gain is smaller than the specified one. One possible reason might be that the dopant concentration and the length of the fiber are not large enough to allow for the desired small signal gain.

Ideal SHG



©TF

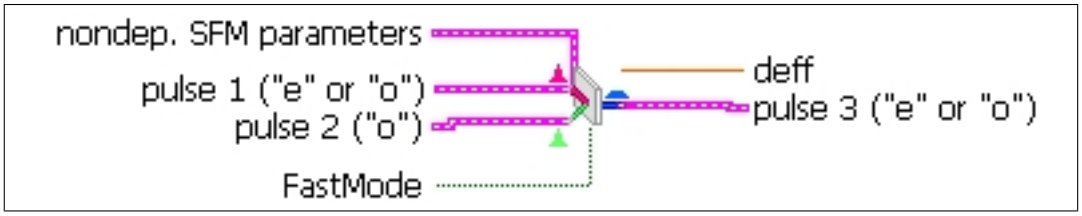
Input		Output	
pulse in	input pulse	pulse:	output pulse

`Ideal SHG` simulates an ideal second harmonic generation process. Ideal in a sense that the phase matching condition is always perfectly fulfilled no matter how broad the input spectrum.

$$E_{\mathrm{SHG}}(t) = E_{\mathrm{F}}^2(t) \tag{5.6}$$

Note that the fluence of the second harmonic field is without meaning as the ideal SHG process assumes some fictive conversion efficiency.

SFM Crystal (nondepleted three wave mixing)



©MH

Input		**Output**	
pulse in 'e' or 'o':	1st pulse in	pulse out 'e':	output pulse
pulse in 'o':	2nd pulse in	deff:	effective nonlinear
crystal:	crystal material		constant
type:	phase matching type		
length:	crystal length		
theta:	angle($\vec{k}$,opt. ax.)		
phi:	rotation angle		
delay=0!:	ignore delay between input pulses		
sinc^2 theory?:	no group velocity effects		
output array:	full, half, or every second		
fast mode:	no interpolations		

`SFM Crystal (nondepleted three wave mixing)` simulates a three wave mixing process in a nonlinear crystal, such as second harmonic generation, sum frequency mixing, or difference frequency mixing. In order to save computation time we assume that both fundamental input beams are not substantially depleted by the conversion process. As shown in an earlier chapter this leaves us with only one differential equation to solve.

Presently we have only implemented BBO, LBO, KDP, $AgGaS_2$, and $LiIO_3$. Other crystals may be added upon request. One may choose between type I (ooe) and type II (eoe) phase matching.

Choosing the mode `delay=0` allows to ignore a possibly present time delay between the two input pulses.

With the `sinc^2 theory` mode it is possible to ignore effects on the mixing process caused by phase modulation. This is achieved by simply using the absolute value of the phase matching term only

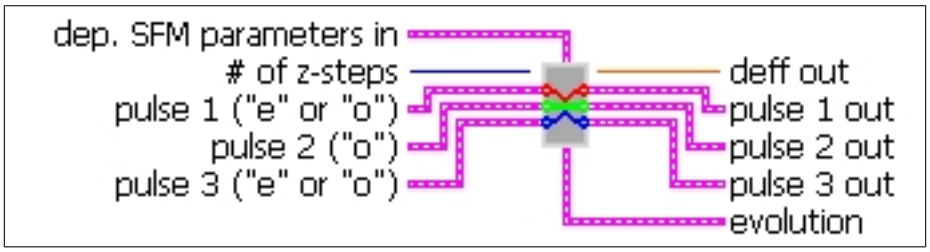
$$\eta(\omega_1, \omega_2)' = |\eta(\omega_1, \omega_2)| \tag{5.7}$$

$$\eta(\omega_1, \omega_2)'^2 = \mathrm{sinc}^2 \left[\frac{1}{2} \Delta k(\omega_1, \omega_2)\, L \right] \tag{5.8}$$

Since this approximation is often found in textbooks, especially when dealing with the mixing of three monochromatic fields, this mode of operation allows to compare results of simulations with results given in the literature. In addition, the influence of a limited spectral bandwidth acceptance may be studied without the influence of phase modulation, for example caused by a difference in the group velocities.

The calculated output waveform has twice the number of sample points as the input pulses, which is caused by the convolution. One may stick to that or one may reduce the number of samples by using every second sample point or all samples in only half of the spectral window.

SFM Crystal (depleted three wave mixing)



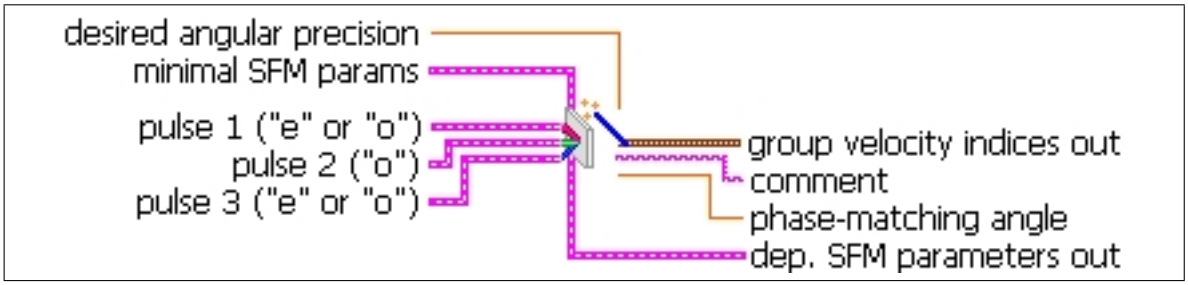
©MH

Input		**Output**	
pulse 1 ('e' or 'o'):	1st pulse in	pulse 1 out:	output pulse 1
pulse 2 ('o'):	2st pulse in	pulse 2 out:	output pulse 2
pulse 3 ('e' or 'o'):	3rd pulse in	pulse 3 out:	output pulse 3
crystal:	crystal material	deff:	effective nonlinear
type:	phase matching type		constant
length:	crystal length	evolution chart:	evolution of spectra
# of z-steps:	integration steps		
theta:	angle($\overrightarrow{k}$,opt. ax.)		
phi:	rotation angle		
delay=0!:	ignore delay between input pulses		
fast mode:	no interpolations (recommended)		

In principle `SFM Crystal (depleted three wave mixing)` is a very powerful tool and is capable of handling any three wave mixing process ($\chi^{(2)}$-process). Dispersion effects to all orders of Taylor coefficients are included since we use the Sellmeier equations to calculate the index of refraction. Quite naturally, back conversion and other processes occurring as the fields propagate through the crystal are accurately incorporated.

As mentioned earlier we assume $\omega_3 = \omega_1 + \omega_2$ which in the case of difference frequency generation may be rearranged to $\omega_2 = \omega_3 - \omega_1$. Note, that you need to consider this by connecting the pulses to the right inputs of `SFM Crystal (depleted three wave mixing)`.

Crystal Wizard

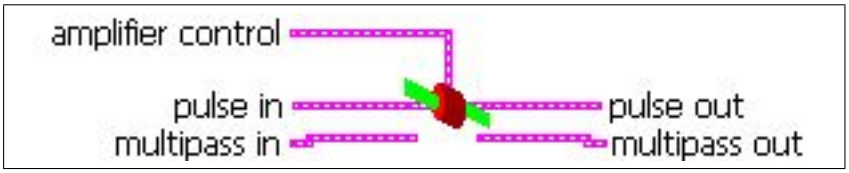


©MH

Input		Output	
pulse 1 pulse 2 pulse 3	the input pulses have to be connected such that the crystal wizard knows the central wavelengths	dep. SFM parameters out: group velocity indices out:	cluster that is compatible to depleted SFM ratio of c_0 over the group velocities of the pulses
minimal SFM params: desired angular precision:	essential SFM parameters, everything but angles required precision	comment: phase-matching angle:	information on which angle is the phase matching angle calculated phase-matching angle

The `crystal wizard` is a tool to determine the optimal angular orientation of a nonlinear crystal for a specific sum frequency mixing situation. Therefore, it requires a control panel similar to that of the `SFM Crystal (depleted three wave mixing)` crystal but without theta or phi angles. The output is a control panel compatible to that of `SFM Crystal (depleted three wave mixing)`. Additional inputs are the three pulses of the three wave mixing and an angular precision value (for example 0.05 deg). The `crystal wizard` determines whether theta or phi is the phase matching angle, finds the optimal phase matching angle, and in addition optimizes the other angle for a maximal d_{eff}.

Amplifier



©TF

Input		Output	
pulse in	input pulse	pulse out:	output pulse
input angle:	0 deg or Brewster angle		
pump duration:	duration of pump pulse		
delay to 1st pulse:	delay between pump and first pass		
pump fluence left:	pump fluence left		
pump fluence right:	pump fluence right		
% concentration:	active ion concentration		
crystal length:	crystal length		
medium:	laser medium		
effects:	switch on/off		

The `amplifier` simulates a one- or two-sided longitudinally pumped laser amplifier. In order to avoid reflection losses at the crystal, the angle of incidence may be chosen to be equal to Brewster angle instead of normal incidence. The amplifier is characterized by its concentration of active ions and the rod length. Both pump beams have the same duration, but may have different energy densities, and the delay between the pump beam and the first pass must be specified. The module accounts for saturation in the amplification process, for dispersive effects due to the resonant amplification, and for dispersive effects of the host material. The number of iterations is automatically determined. If one wants to simulate a two- or other multi-pass amplifier the inversion cluster must be connected as shown in the tutorials. We also need to specify whether the direction of the seed beam through the amplifier crystal alternates, as in a regenerative amplifier, or does not alternate, as in a ring-type multi-pass amplifier.

Fred van Goor from the University of Twente did simulate a number of published Ti:sapphire amplifiers. We are very happy that he agreed to publish his results here; thank you Fred!

'... The TiSa amplifier vi is very helpful for the design of a multi-pass Ti-sapphire amplifier. We compared the simulation results (see table 5.1) with a number of amplifier systems published in the literature. We selected systems reported after 1994, which were

designed to operate at maximum efficiency. In table 5.1 six different amplifiers are simulated. The fluences where calculated using the published input- and output energies and beam diameters. Also the published lengths and diameters of the TiSa rods and the number of passes were used in the simulations. Some papers did not mention the size of the input- or output beam or the length of the rod. In these cases we made a guess. We believe that these guesses are reasonable because the diameter of the pump beam should be slightly larger than that of the input- and output beam. The Titanium concentration was not given in most papers, we guessed 0.15%. This value turns out to be not so critical. The diameter of the output beam was not given in the papers. We have chosen it equal to the diameter of the input beam. We also compared the peak intensities of the six systems, calculated from the published output energies, pulse lengths of the stretched pulses, and the given or guessed output beam diameter. It turns out that some of the systems operate close to the damage threshold of Ti:sapphire, which is about 5GW/cm2. We are not sure about the values resulting from reference [5], as the peak intensity is unrealistic high. Maybe the guessed beam diameter of the output is not correct in this case.

Table 5.1: Comparison between published results and simulations. Number of passes n, pump fluence F_{pump}, seed fluence F_{in}, amplified seed fluence F_{out}, pump beam diameter d_{pump}, seed beam diameter d_{seed}, pump energy W_{pump}, seed energy W_{in}, simulated amplified energy W_{sim}, measured amplified energy W_{exp}, length of TiSa crystal L_{TiSa}, diameter of TiSa crystal d_{TiSa}, pulse duration τ, and maximum intensity I_{max}.

n	F_{pump} J/cm^2	F_{in} mJ/cm^2	F_{out} J/cm^2	d_{pump} mm	d_{in} mm	W_{pump} mJ	W_{in} mJ	W_{sim} mJ	W_{exp} mJ	L_{TiSa} mm	d_{TiSa} mm	Ti %	τ ps	I_{max} GW/cm^2	ref.	rem.
4	2.82	45.84	1.21	6.20	5.00	850	9	237.6	235	13.00	13.00	0.15	300	4.03	1	good
3	3.70	57.69	1.72	8.50	8.00	2100	29	864.6	790	12.00	20.00	0.15	330	5.21	2	ok
4	1.99	28.29	1.23	6.70	6.00	700	8	347.8	340	20.00	?	0.15	1700	0.72	3	good
4	2.95	15.92	1.25	5.00	4.00	580	2	157.1	160	10.00	12.00	0.15	950	1.32	4	good
4	2.55	19.89	0.99	5.00	4.00	500	2.5	124.4	120	7.00	10.00	0.23	40	24.75	5	good
4	3.16	15.28	1.44	5.50	5.00	750	3	282.7	280	18.00	12.00	0.15	800	1.80	6	good
4	2.48	61.89	1.01	12.00	12.00	2800	70	1142.3	1340	10.00	20.00	0.15	80	7.10	7	ok
4	2.63	15.59	1.10	4.00	3.50	330	1.50	105.8	100	15.00	?	0.25	80	13.75	7	good

The simulations agree reasonably good or good with the published results. Most systems operate above the saturation fluence at the last pass yielding maximum efficiency. The pulse lengths of the stretched pulses are such that the amplifiers operate at a safe peak intensity well below or sometimes close to the damage threshold. The simulations have been performed with pulse durations of about 90 ps. It was not possible to use the published pulse lengths because the number of samples became too large when increasing the chirp. However, we believe that this will not influence the simulation results much because we are dealing with energies and not with intensity waveforms...'

[1] Barty, Opt. Lett. **19** (1994), p1442
[2] Itani, Opt. Comm. **134** (1997), p134
[3] Yamakawa, Opt. Lett. **23** (1998), p1468
[4] Barty, Opt. Lett. **21** (1996), p668
[5] Zhou, Opt Lett. **20** (1995), p64
[6] Walker, Optics Express **5** (1999), p196
[7] Wang, J. Opt. Soc. Am. B **16** (1999), p1790

Chapter 6

Detectors

To fully characterize a femtosecond laser pulse it is necessary to measure the amplitude as well as the phase of it either in the spectral or in the time domain. All electronic devices however are too slow to come up to this challenge. And because no direct methods are applicable indirect techniques have to be used. All of those techniques rely on an instantaneous nonlinear effect.

In $\mathcal{LAB}^{\mathrm{II}}$ we have implemented a large number of those indirect techniques, but we have also implemented rather unrealistic devices which are extremely useful for debugging purposes. All detectors have a number of different outputs. Some may require an input, which, if not connected, is set to a useful default value.

Correlation measurements are the most useful tools to determine the temporal intensity in femtosecond world. They are conceptually easy and may be realized in different variations depending on what exactly it is you want to measure. There are background free versions and those which do have background, there are second and third order versions, depending on what nonlinear process you are using, and there are interferometric and intensity correlation versions, depending on whether the two beams interact in a collinear or non-collinear fashion.

Background-free versions have the great advantage that they may be used to measure 'deep' correlations, that is, to measure the pulse intensity over many orders of magnitude.

Second order as compared to third order correlations have intrinsic time symmetry, which prevents you from telling your lab mate whether the little bump showing up in the measurement is a pre- or a post-pulse.

Interferometric version in contrast to intensity correlations give you some additional information on the phase of the pulse, but they are almost never background free.

All intensity correlators in $\mathcal{LAB}^{\mathrm{II}}$ have either one or two input pulses, so they allow for auto as well as cross correlation measurements.

FROG is an abbreviation for (**F**requency **R**esolved **O**ptical **G**ating) and seems to become one of the standard diagnostic tools in femtosecond laser science. It allows to almost fully characterize a femtosecond laser pulse. The only missing thing is the absolute phase of the pulse, but other than that you get the amplitude and the relative phase from a FROG

measurement. There is some mathematics involved in that too, because you need to process the measured FROG trace, run an iterative algorithm, and if you have done everything as you should you'll get the field.

FROG in principle is conceptually easy, take any intensity correlation setup and spectrally resolve the correlation signal. This is why there are many different types of FROG 's out there. The most common ones are SHG FROGs because they are easy to build and require very little energy to operate. They have one disadvantage, however, and that is they have no idea in which direction the time axis points. For example a SHG FROG never allows you to distinguish between a positive and a negative phase modulation if they only differ by the sign. The THG FROG resolves this problem but needs much more optical elements to align and a little more energy, so does the PG FROG.

The FROG signal is a function of both the frequency and the delay time

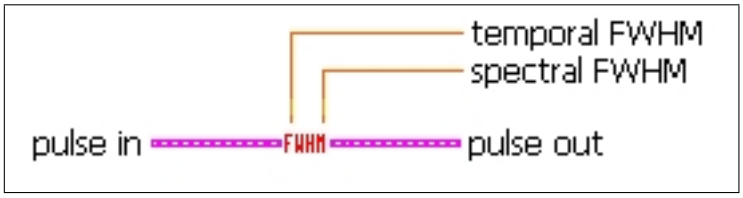
$$F(\omega,\tau) \propto \left| \int_{-\infty}^{\infty} dt\ f(t,\tau)\ e^{-i\omega t} \right|^2 . \tag{6.1}$$

where the function $f(t)$ is determined by the nonlinear process you're employing

$f(t,\tau) = E_1(t)\ E_2^*(t-\tau)$	for	DFG	FROG
$f(t,\tau) = E_1(t)\ E_2(t-\tau)$	for	SFG	FROG
$f(t,\tau) = E_1^2(t)\ E_2(t-\tau)$	for	THG	FROG
$f(t,\tau) = E_1(t)\ \lvert E(t-\tau)\rvert^2$	for	PG	FROG
$f(t,\tau) = E_1^2(t)\ E^*(t-\tau)$	for	TG	FROG
$f(t,\tau) = E_1^2(t)\ E^*(t-\tau)$	for	SD	FROG

All FROG's implemented here assume that the nonlinear crystal you are using has a sufficiently broad spectral acceptance to support the full laser bandwidth. If you want to see what happens if this is not the case you may construct a FROG by using more fundamental $\mathcal{LAB}^{\mathrm{II}}$ modules.

FWHM

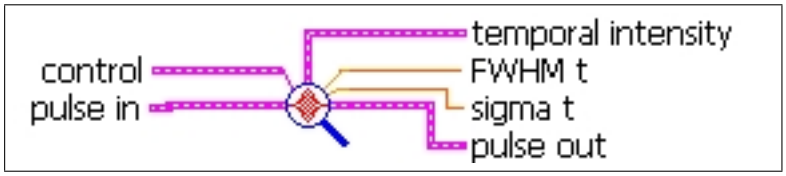


©TF

Input		Output	
pulse in:	input pulse	pulse out:	output pulse
		temporal FWHM:	FWHM of temporal intensity
		spectral FWHM:	FWHM of spectral intensity

The `FWHM in time` module calculates the FWHM (**F**ull **W**idth at **H**alf **M**aximum) of the temporal and spectral intensity distribution, no matter what kind of slowly varying envelope the pulse has. It may lead to funny results whenever the intensity is heavily structured.

Intensity



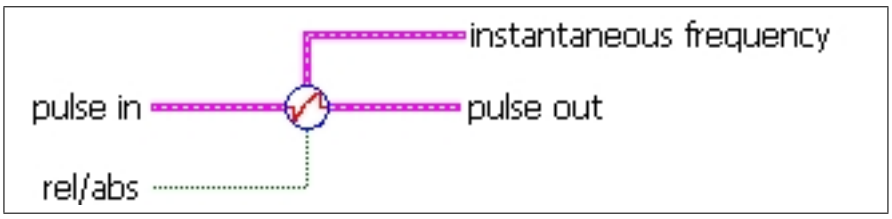
©TF

Input		Output	
pulse in:	input pulse	pulse out:	output pulse
rel/abs:	relative or absolute time axis	intensity:	temporal intensity
		FWHM t:	FWHM of the temporal intensity
time step:	detector time step	sigma t:	2nd order moment of the temporal intensity
samples:	number of samples		

The `intensity` module gives you a number of useful information on the pulse. The pulse is going in on the left and coming out on the right, so you may use it at any place in your virtual experiment. Its outputs are a plot and two numbers.

The plot shows the temporal intensity and the two numbers are the temporal FWHM and the second moment. As mentioned above the FWHM may lead to funny results whenever the pulse intensity becomes heavily structured. In this case the second moment leads to much more reliable results if you need a number that specifies the temporal spread. The `rel/abs` selector allows to toggle between a relative time axis, that is, the pulse is always centered around time zero, or an absolute time axis, that is, the first order phase t_0 is included and the time axis is centered around that time.

Instantaneous Frequency

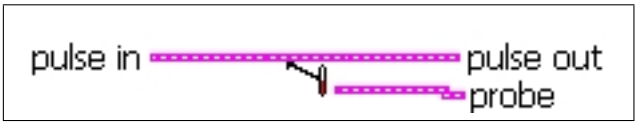


©TF

Input		Output	
pulse in:	input pulse	pulse out:	output pulse
		instantaneous frequency	instantaneous frequency distribution

The `instantaneous frequency` module plots the relative instantaneous frequency. Relative here means relative to the laser's center frequency. So, if the pulse has no chirp, you'd expect the relative frequency to be zero everywhere.

Probe

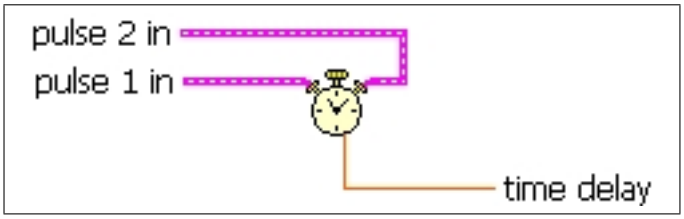


©TF

Input		Output	
pulse in:	input pulse	pulse out	output pulse

The `probe` module was originally designed as a debugging tool. But it seemed to be very useful so we made a detector out of it. The output of `probe` are four plots showing the spectral amplitude and phase and the temporal amplitude and phase. `probe` allows you to access these quantities at whatever position in the virtual experiment you need to look at them.

Run Time Difference

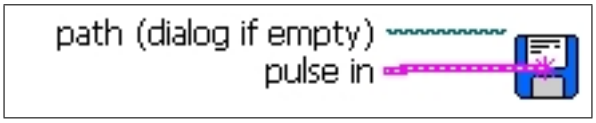


©TF

Input		Output	
pulse 1 in:	1st pulse in	time delay:	time delay between 1st and 2nd pulse
pulse 2 in:	2nd pulse in		

`run time difference` determines the delay time between two pulses by calculating the difference of the first order phase terms. This is sometimes useful if you want to calculate the time difference between two pulses having a different center frequency but which have traveled through the same piece of glass and now you want to know how far are they apart in time.

Save Pulse To File

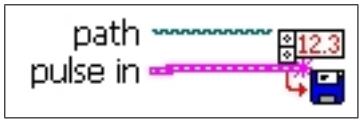


©TF

Input		Output
pulse in:	input pulse	
path:	specify file name (dialog if empty)	

The module `save to file` takes a pulse at any point in the virtual experiment and stores it to a file. You may load it later using the corresponding `read pulse` module.

Save Pulse To Spreadsheet

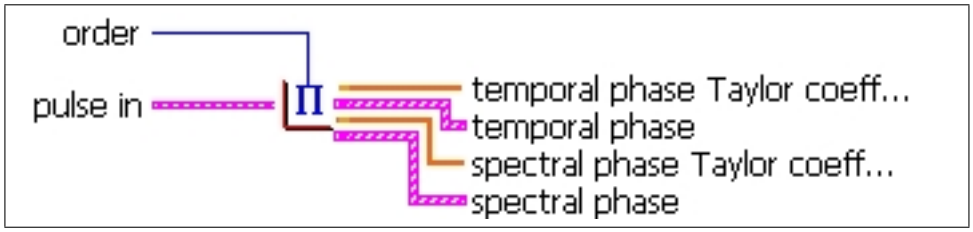


©TF

Input		Output
pulse in:	input pulse	
path:	specify file name (dialog if empty)	

The `save pulse to spreadsheet` module stores a pulse to a spreadsheet file. The format is equivalent to that used by the FROG program (©Femtosoft). The first column gives the wavelength in nm, the second the square of the absolute value ('spectral intensity') in units of J/(m^2 nm), the third column the phase, the fourth column the real part, and the last column the imaginary part of the spectral field in units of V/(m nm). Obviously, there is too much information here, but we decided to simply use the format structure of the FROG software in order to ensure compatibility.

Taylor Phase

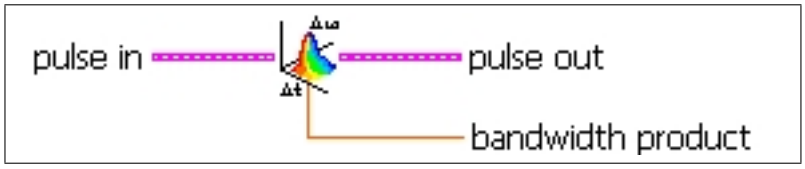


©TF

Input		Output	
pulse in:	input pulse	temporal phase Taylor...:	Taylor coefficients of temporal phase
order:	maximum order of series expansion	temporal phase:	plot to check fit
		spectral phase Taylor...:	Taylor coefficients of spectral phase
		spectral phase:	plot to check fit

The `Taylor phase` module expands the spectral and the temporal phase in a Taylor series and outputs the Taylor coefficients of both up to the order you specify. You may check the quality of the polynomial fit by inspecting the two plots.

Time Bandwidth Product



©TF

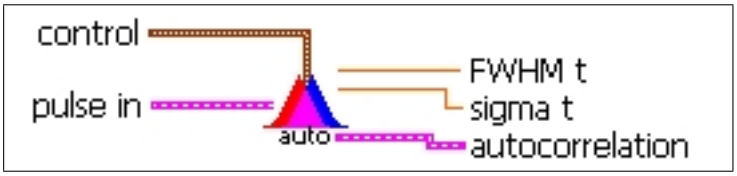
Input		Output	
pulse in:	input pulse	pulse out:	output pulse
		bandwidth product:	time frequency bandwidth product

The `time bandwidth product` module calculates for a given input pulse the product of the temporal intensity FWHM and the spectral intensity FWHM.

$$\mathrm{FWHM}_{I(t)} \ \ \mathrm{FWHM}_{S(\nu)}$$

If you prefer $\Delta\tau\Delta\omega$, simply multiply the result by 2π.

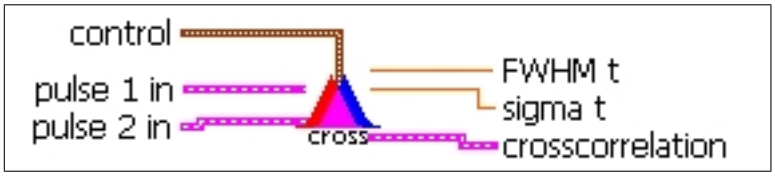
Intensity Autocorrelation



Input		**Output**	
pulse in:	pulse in	FWHM t:	FWHM of the correlation
time step:	detector time step	sigma t:	2nd moment of the correlation
samples:	number of samples	autocorrelation	intensity autocorrelation
type:	type of correlation		

The `intensity autocorrelation` measures the intensity autocorrelation of the incoming pulse. It can be second or third order and it can be background-free or the opposite.

Intensity Cross-Correlation

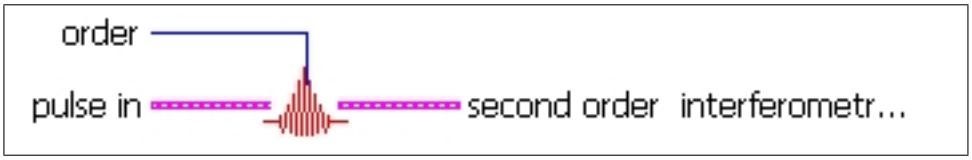


©TF

Input		Output	
pulse 1 in:	pulse in	FWHM t:	FWHM of the correlation
pulse 2 in:	pulse in	sigma t:	2nd moment of the correlation
time step:	detector time step	cross-correlation	intensity cross correlation
samples:	number of samples		
type:	type of correlation		

The `intensity cross-correlation` measures the intensity cross correlation of the two incoming pulses. It can be second or third order and it can be background-free or the opposite.

Interferometric Autocorrelation



©TF

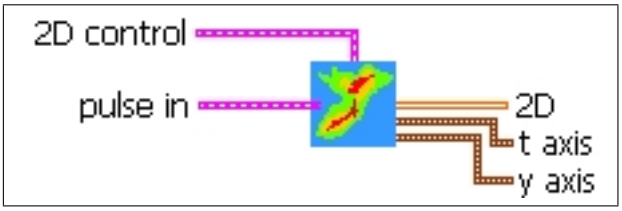
Input		Output	
pulse in:	input pulse	second order inter...:	2rd order interferometric correlation
order:	order of nonlinearity		

The `interferometric autocorrelation` has the great advantage that it allows to get some information on the phase of the pulse. In principle such a setup combines correlation and interferometry. The signal is given by

$$I(\tau) = \int dt \, |E(t) + E(t+\tau)|^{2n}$$

where n is the order of the autocorrelation. A linear correlation corresponds to $n = 1$, a second order to $n = 2$ and a third order to $n = 3$.

2D Distribution



©TF

Input		Output	
pulse in:	input pulse	2D distribution:	intensity plot
time step:	time step	t axis:	x scale information
field size:	sampling grid size	y axis:	y scale information
save option:	specify output format		
path to file:	file name		
type:	select distribution		
y axis:	frequency or wavelength		

The `2D distribution` calculates a time-frequency distribution from the electric field of the pulse. You may select between:

- Spectrogram

$$S(\omega,t) = \left| \frac{1}{\sqrt{2\pi}} \int \mathrm{d}t' E(t') E(t'-t)\, \mathrm{e}^{-i\omega t'} \right|^2 \tag{6.2}$$

- WFFT (windowed fast Fourier transform). The function $f(t)$ is a real Gaussian of some appropriate width.

$$S(\omega,t) = \left| \int \mathrm{d}t' E(t') f(t'-t)\, \mathrm{e}^{-i\omega t'} \right|^2 \tag{6.3}$$

- Wigner-Ville

$$S(\omega,t) = \frac{1}{2\pi} \int \mathrm{d}t' E^*(t-t'/2) E(t+t'/2)\, \mathrm{e}^{-i\omega t'} \tag{6.4}$$

- Page

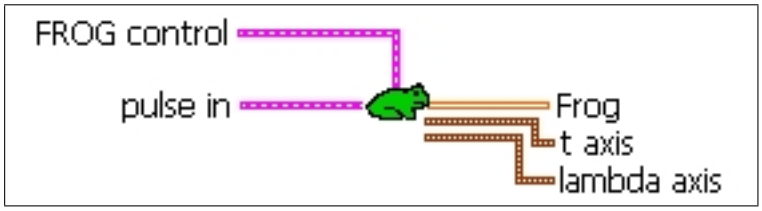
$$S(\omega,t) = \frac{\partial}{\partial t} \left| \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{t} \mathrm{d}t' E(t')\, \mathrm{e}^{-i\omega t'} \right|^2 \tag{6.5}$$

- Rihaczek

$$S(\omega,t) = \frac{1}{\sqrt{2\pi}} E(t) \widetilde{E}^*(\omega)\, \mathrm{e}^{-i\omega t} \tag{6.6}$$

With the time step and the field size you determine the total time window and from this the frequency step and the frequency window are derived. The resulting intensity plot may be saved in jpeg format to the specified file name. You can also select whether you would like the y-axis to be frequency or wavelength.

FROG



©TF

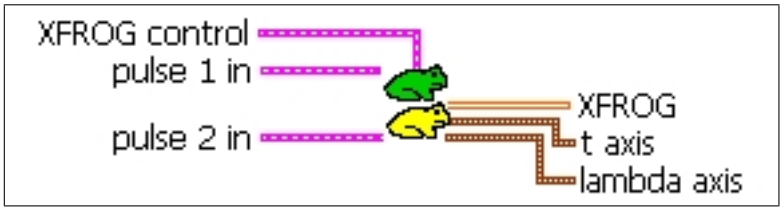
Input		Output	
pulse in:	input pulse	FROG:	spectrogram
time step:	FROG time step	t axis:	x scale information
field size:	sampling grid size	lambda axis:	y scale information
save option:	specify output format		
path to file:	file name		
type:	nonlinear process		

The `FROG` module mixes two replicas of the input pulse in a nonlinear medium and the spectrum of the generated nonlinear signal is detected as a function of the time delay between the two replicas. The FROG trace is displayed as a function of delay time (x-axis) and wavelength (y-axis). Connect the scale information to a property node of the intensity plot to get properly calibrated axis.

The FROG module requires a grid size and a time step. The larger the grid size the longer it takes to calculate the FROG trace. Since Fourier transformation automatically determines the frequency step size once you specify the time step there is no need to input both. We decided to use the time step as input because in most experiments a stepper or a dc motor driving a translation stage determines the resolution. Most spectrometer usually measure far too many spectral samples (typically 512 to 2048 samples) and most of them have to be thrown away in the retrieval process.

At the moment there are two save options available, namely jpeg format and retrieval format. If you chose one of those make sure to specify a path and a filename. The color map used for the jpeg format is equal to that used by Femtosoft, which makes it easier to compare simulated to measured or retrieved FROG traces. If you select retrieval format an output file will be generated that can be read directly into the FROG 3.0 software. Check the box 'use header information' and specify 'order: delay' and 'read in as: constant frequency'.

XFROG



©TF

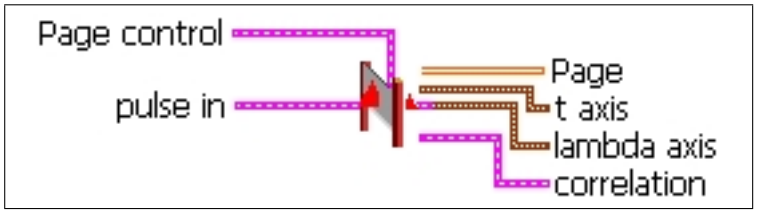
Input		Output	
pulse 1 in:	input pulse	THG FROG:	THG spectrogram
pulse 2 in:	input pulse	t axis:	x scale information
time step:	FROG time step	lambda axis:	y scale information
field size:	sampling grid size		
save option:	specify output format		
path to file:	file name		

XFROGS are identical to FROGs except the two pulses may be different. In addition XFROG offers the option to use difference frequency generation as a nonlinear process, which doesn't make sense for FROGs.

The XFROG module requires a grid size and a time step. The larger the grid size the longer it takes to calculate the FROG trace. Since Fourier transformation automatically determines the frequency step size once you specify the time step there is no need to input both. We decided to use the time step as input because in most experiments a stepper or a dc motor driving a translation stage determines the resolution. Most spectrometer usually measure far too many spectral samples (typically 512 to 2048 samples) and most of them have to be thrown away in the retrieval process.

At the moment there are two save options available, namely jpeg format and retrieval format. If you chose one of those make sure to specify a path and a filename. The color map used for the jpeg format is equal to that used by Femtosoft, which makes it easier to compare simulated to measured or retrieved FROG traces. If you select retrieval format an output file will be generated that can be read directly into the FROG 3.0 software. Check the box 'use header information' and specify 'order: delay' and 'read in as: constant frequency'.

Knife Edge Correlator



©MH

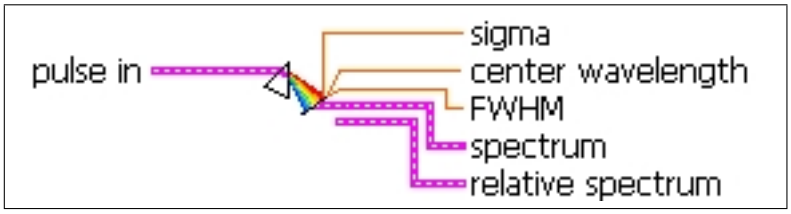
Input		Output	
pulse in:	input pulse	Page:	integral over Page function
time step:	detector time step	t axis:	x scale information
field size:	sampling grid size	lambda axis:	y scale information
save option:	specify output format		
path to file:	file name		
rise time:	rise time of the shutter		

The `knife edge correlator` is something we have been using in the lab quite often to temporally overlap two beams of different center frequency. One beam needs to be strong enough to create a plasma. The other beam is either diffracted or absorbed depending on the electron density in the plasma. If you record the intensity of the weak beam as a function of the delay between the two pulses it turns out that the whole setup is more or less the temporal equivalent to a knife edge experiment. What you measure is the integral of the intensity up to the delay time. If you spectrally disperse the signal and check out the math it turns out that what you measure is the integral of the Page distribution, assuming that the shutter is infinitely fast

$$S(t,\omega) \propto \left| \int d\tau \; e^{-i\omega\tau} \; E^{+}(\tau) \; \Theta(t-\tau) \right|^2 \tag{6.7}$$

The infinitely fast shutter is incorporated by using the Heaviside step function $\Theta(t-\tau)$. Even in the case the shutter rise time is finite, equation (6.7) is a pretty good description. So all you need to do is to calculate the derivative of the time frequency distribution with respect to time and you obtain the Page distribution of the pulse. The Page distribution has the advantage, in contrast to the spectrogram (FROG), that you can invert it in a straight forward way to extract the electric field. No need to iterate here. The `knife edge correlator` has two outputs and that is the correlation signal and the time integral over the Page function. What you need to specify is the rise time of the shutter. If you start with a small rise time and increase it step by step you will also see the influence of a finite rise time on the appearance of the time frequency distribution.

Spectrometer



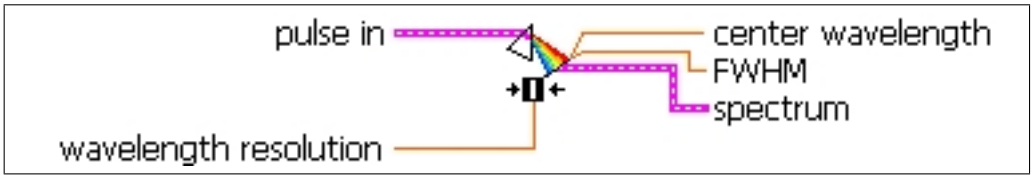
©TF

Input		Output	
pulse in:	input pulse	center wavelength:	center wavelength
		sigma:	second moment of spectral intensity
		FWHM:	FWHM of spectral intensity
		spectrum:	spectral intensity vs. wavelength
		relative spectrum:	relative frequency spectrum

The `ideal spectrometer` has a number of useful outputs. First of all there are two graphics, the wavelength spectrum and the relative frequency spectrum. In addition you'll find three numbers, the center wavelength, the spectral FWHM in units of nm, and the second moment. The second moment is a useful quantity when the spectrum becomes heavily structured for example due to self-phase modulation and the FWHM becomes a somewhat meaningless number.

The spectrometer is ideal in a sense that it has no device limited spectral resolution. The resolution is only determined by the number of samples and the full spectral window you are using. In addition, neither the grating nor the detector have a wavelength dependent sensitivity.

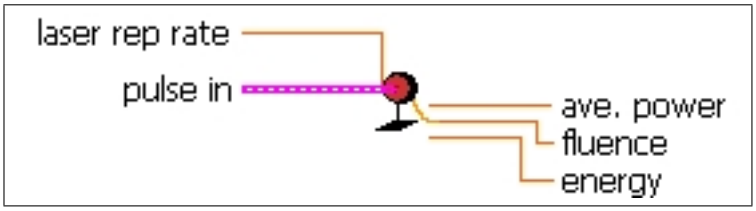
Spectrometer with Resolution



Input		**Output**	
pulse in:	input pulse	center wavelength:	center wavelength
wavelength resolution:	spectral resolution of spectrometer	FWHM:	FWHM of spectral intensity
		spectrum:	spectral intensity vs. wavelength

The `spectrometer with resolution` is almost the same as the `ideal spectrometer` except that you need to specify a device resolution in units of nm.

Energy Meter

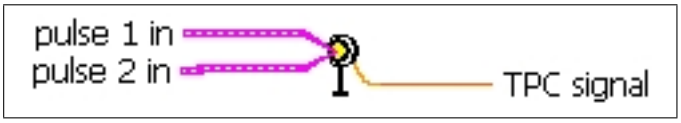


©TF

Input		Output	
pulse in:	input pulse	fluence:	energy per unit area
laser rep rate:	repetition rate	ave. power:	average power
		energy:	pulse energy

The fluence of the laser pulse may be measured with the `energy meter`. The device integrates the temporal intensity and outputs the pulse energy per unit area. If you need you may also use the `energy meter` whenever you need a photo diode somewhere. Just think of it as a calibrated photo diode. If you specify a laser repetition rate you may also look at the average power.

TPC Diode



©TF

Input		Output	
pulse A in:	1st input pulse	TPC signal	output signal
pulse B in:	2nd input pulse		

Conventional photo diodes with a large bandgap may sometimes be used to measure the pulse intensity. That is because the generated photo-current depends on the square of the intensity since you need two photons to produce one electron hole pair if the bandgap is $\hbar\omega_0 < E_{gap} \leq 2\hbar\omega_0$. So the signal you'd measure in this case is

$$S = \int dt\ I^2(t) \tag{6.8}$$

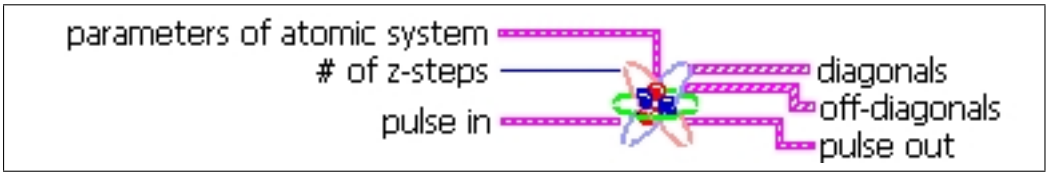
The `TPC-diode` has two inputs. Usually both inputs get the same pulse, but you may want to use two different pulses to realize something like a correlator.

Chapter 7

Interactions

In addition to the standard elements we have included special elements which have been programmed by users of $\mathcal{LAB}^{\mathrm{II}}$. You will find a resonant three level system, a n-level ladder system, and a diatomic molecule. Both allow you to study fundamental interaction processes between light and matter. The resonant three level system numerically solves the coupled Maxwell-Bloch equations and has been programmed by Ralf Netz. He is also the author of the ladder system vi. The diatomic molecule solves the time dependent Schrödinger equation for a diatomic molecule with a ground and one excited state and has been programmed by Tamas Rozgonyi.

Resonant Three Level System



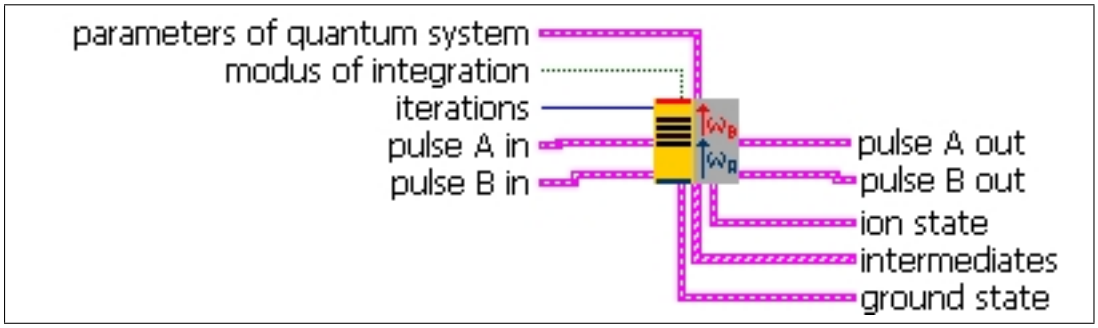
©RN

Input		**Output**	
pulse in:	input pulse	pulse out:	output pulse
wavelength of 13 transition:	1st resonance	diagonals:	diagonal elements
dipole moment of 13 transition:	1st dipole moment		of density matrix
wavelength of 12 transition:	2nd resonance	off diagonals:	off-diagonal
dipole moment of 12 transition:	2nd dipole moment		elements of
cell length:	interaction length		density matrix
z-steps:	number of integration steps		
density:	particle density		
initial values of density matrix:			
ρ_{11}:	population state 1		
ρ_{22}:	population state 2		
ρ_{33}:	population state 3		
ρ_{31}:	off-diagonal 31		
ρ_{21}:	off-diagonal 21		
ρ_{32}:	off-diagonal 32		

The `resonant three level system` simulates the interaction of a short laser pulse with a resonant three level system. The simulation is based on solving the coupled Maxwell Bloch equations. The three level system is characterized by two resonance wavelengths, two corresponding dipole moments, the atomic density, and the interaction length. In addition, in manual mode the number of z integration steps has to be specified. After the calculations the diagonal and separately the off-diagonal elements of the density matrix may be plotted as a function of time.

With an appropriate choice of the transition wavelengths and the initial population distribution, one may either simulate a V-type, a Λ-type, or a ladder-type system.

Parallel Ladder System



©RN

Input		**Output**	
pulse in (A):	input pulse A	pulse out (A):	output pulse A
pulse in (B):	input pulse B	pulse out (B):	output pulse B
modus of integration	slow/fast	ion state	ion state population
iterations:	# of z-steps	intermediates	population of
transition wavelengths:	wavelengths		intermediate states
	of $\|g\rangle \rightarrow \{\|k\rangle\}$	ground state	ground state population
	and $\|g\rangle \rightarrow \|\gamma\rangle$		
dipole moments m_k:	dipole moments		
	of transition		
	$\|g\rangle \rightarrow \{\|k\rangle\}$		
dipole moments d_k:	dipole moments		
	of transition		
	$\{\|k\rangle\} \rightarrow \|\gamma\rangle$		
# of states:	number of		
(Min.=3)	involved states;		
T_{ion}:	lifetime of		
	upper state $\|\gamma\rangle$		
	$(\gamma = 1/T_{\text{ion}})$		
particle density:	particle density		
propagation length:	propagation length		
initial values	$c_g(t_0)$, $\{c_k(t_0)\}$		
of amplitudes:	and $c_\gamma(t_0)$		

The `ladders parallel` module calculates the space-time evolution of a multi-level quantum system (Fig. 7.1) and its driving fields. The multi-level system can be described mathematically by the differential equations (in the rotating wave approximation, RWA)

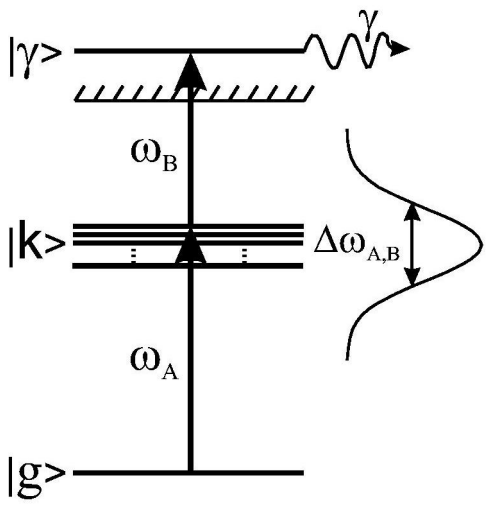


Figure 7.1: Parallel connection of three-level ladder systems. The resonant transitions are driven by the electric fields $E_{\rm A}(t) = \frac{1}{2}\,\tilde{\mathcal{E}}_{\rm A}(t)\,{\rm e}^{-i\omega_{\rm A}t} + c.c.$ and $E_{\rm B}(t) = \frac{1}{2}\,\tilde{\mathcal{E}}_{\rm B}(t)\,{\rm e}^{-i\omega_{\rm B}t} + c.c.$.

$$\dot{c}_{\rm g} = i\sum_{k=1}^{N}\Omega_{k{\rm g}}^{*}\,c_k \tag{7.1}$$

$$\dot{c}_k = i(\omega_{\rm A}-\omega_{k{\rm g}})\,c_k + i\Omega_{k{\rm g}}\,c_{\rm g} + i\Omega_{\gamma k}^{*}\,c_\gamma \tag{7.2}$$

$$\dot{c}_\gamma = i(\omega_{\rm A}+\omega_{\rm B}-\omega_{\gamma{\rm g}}+i\gamma)\,c_\gamma + i\sum_{k=1}^{N}\Omega_{\gamma k}\,c_k. \tag{7.3}$$

$\Omega_{k{\rm g}} = \mu_{k{\rm g}}\tilde{\mathcal{E}}_{\rm A}(t)/2\hbar$ and $\Omega_{\gamma k} = d_{\gamma k}\tilde{\mathcal{E}}_{\rm B}(t)/2\hbar$ are the Rabi frequencies of the transitions $|{\rm g}\rangle \rightarrow |k\rangle$ and $|k\rangle \rightarrow |\gamma\rangle$. Since a decay process (e.g. ionization) is included in Eq. 7.3 the atomic system is not closed for $\gamma > 0$.

The evolution in space-time of the driving fields $\tilde{\mathcal{E}}_{\rm A}(t)$ and $\tilde{\mathcal{E}}_{\rm B}(t)$ is given by the simplified wave equation in the RWA

$$\left[\frac{\partial}{\partial y}+\frac{1}{c_0}\,\frac{\partial}{\partial t}\right]\tilde{\mathcal{E}}_{\rm A}(y,t) = i\frac{\mu_0\;\omega_{\rm A}\;c_0}{2}\;\tilde{\mathcal{P}}_{\rm A}(y,t)$$

$$\left[\frac{\partial}{\partial y}+\frac{1}{c_0}\,\frac{\partial}{\partial t}\right]\tilde{\mathcal{E}}_{\rm B}(y,t) = i\frac{\mu_0\;\omega_{\rm B}\;c_0}{2}\;\tilde{\mathcal{P}}_{\rm B}(y,t),$$

where $\tilde{\mathcal{P}}_{\rm A}(y,t)$ and $\tilde{\mathcal{P}}_{\rm B}(y,t)$ are calculated by

$$\begin{aligned}\tilde{\mathcal{P}}_{\mathrm{A}}(y,t) &= 2\mathcal{N}\sum_{k=1}^{N}\mu_{k\mathrm{g}}\; c_k\; c_{\mathrm{g}}^{*}(y,t)\\ \tilde{\mathcal{P}}_{\mathrm{B}}(y,t) &= 2\mathcal{N}\sum_{k=1}^{N}d_{\gamma k}\; c_\gamma\; c_k^{*}(y,t)\end{aligned}$$

and $\mathcal{N}$ is the particle density. This means that the system of equations combines MAXWELLs wave equation with BLOCHs quantum-mechanical description of the field-matter interaction and has to be solved simultaneously.

In the case of $\tilde{\mathcal{E}}_{\mathrm{A}}(t) = \tilde{\mathcal{E}}_{\mathrm{B}}(t) = \tilde{\mathcal{E}}(t)$ only one wave equation has to be solved with $\tilde{\mathcal{P}}_{\mathrm{A}}(y,t) = \tilde{\mathcal{P}}_{\mathrm{B}}(y,t) = \tilde{\mathcal{P}}(y,t) = 2\mathcal{N}\sum\limits_{k=1}^{N}\left\{\mu_{k\mathrm{g}}\; c_k\; c_{\mathrm{g}}^{*}(y,t) + d_{\gamma k}\; c_\gamma\; c_k^{*}(y,t)\right\}$.

To solve numerically MAXWELLs wave equation and the differential equations for the probability amplitudes — in particular for fast decay processes (faster than the pulse duration) and a broad temporal window — the Predictor-Corrector-Method (`slow modus`) is implemented

$$\begin{aligned}y_{n+1}^{(0)} &= y_n + h\; f(x_n,y_n)\\ \vdots &\qquad\vdots\\ y_{n+1}^{(\nu+1)} &= y_n + \frac{h}{2}\;\left[f(x_{n+1},y_{n+1}^{(\nu)}) + f(x_n,y_n)\right],\end{aligned}$$

where $\nu = 0\ldots 1$, resulting in

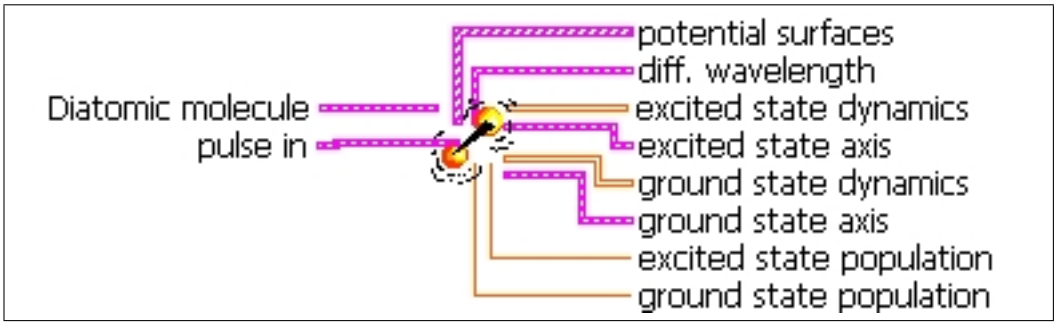
$$\begin{aligned}y_{n+1}^{(0)} &= y_n + h\; f(x_n,y_n)\\ y_{n+1}^{(1)} &= y_n + \frac{h}{2}\;\left[f(x_{n+1},y_{n+1}^{(0)}) + f(x_n,y_n)\right]\\ y_{n+1}^{(2)} &= y_n + \frac{h}{2}\;\left[f(x_{n+1},y_{n+1}^{(1)}) + f(x_n,y_n)\right].\end{aligned}$$

In the case of the `fast modus` the first derivative is approximated by

$$y_{n+1} - y_{n-1} = 2h\; f(x_n,y_n). \tag{7.4}$$

The minimal number of involved states is 3 (one three-level ladder). To create a two-level system only one dipole moment should be $\neq 0$. To examine the field-matter interaction without propagation, e.g., the # of z-steps can be 2 (Minimum), the propagation length 0.005 mm and the particle density 1×10^{12} cm^{-3}.

Diatomic Molecule



©TR

Input		**Output**	
pulse in:	input pulse	potential surfaces:	the two Morse
grid size:	spatial grid size		potentials
xmin:	minimum atom separation	diff:	resonance wave-
dx:	spatial step size		length vs. atomic
mass 1:	mass of atom 1		separation
mass 2:	mass of atom 2	excited state wavepacket:	2D wavepacket
dipole matrix:	dipole matrix elements		motion
ground state:		excited state population:	final population
D:	dissociation energy	ground state population:	final population
beta:	width	ground state wavepacket:	2D wavepacket
x0:	equilibrium separation		motion
C:	asymptotic energy		
excited state:			
D:	dissociation energy		
beta:	width		
x0:	equilibrium separation		
C:	asymptotic energy		
simulation stop:	time to stop simulation		

The module `diatomic molecule` simulates the interaction of a pulse and a diatomic molecule, which is characterized by an electronic ground and an excited state. The molecule is represented by a pair of Morse potentials, the two masses, and the transition dipole matrix. It is necessary to specify the spatial grid that is used to calculate the space time propagation of the wavepackets. The initial ground state wavepacket is calculated from the parameters of the ground state potential. You may plot the difference between the two potentials in terms of wavelength as a function of the internuclear separation to determine the resonant wavelengths. During the calculations a pop up window shows the evolution of

both wavepackets. At the end of the calculation a two dimensional plot shows the space-time distribution of the wavepackets on both potential surfaces.

Spectral lines

©TF

Input		Output	
pulse in:	input pulse	spectrum:	laser spectrum and atomic lines
min ionization:	minimum ionization stage; neutral = 0, singly ionized = 1 etc.	elements:	array of strings length equal to number of samples
max ionization:	maximum ionization stage		
spectral resolution:	spectral resolution		
list of elements:	list of elements to include		

The module `spectral lines` allows to display atomic emission spectra together with the laser spectrum to find out what lines are within the laser bandwidth. One needs to specify a list of elements, for example Hg and Ar for a mercury lamp with argon buffer gas, and the minimum and maximum ionization stage. Ionization stage zero refers to the neutral state, one to the singly ionized atom etc. The spectral resolution determines the width of the atomic lines and is determined by the spectrometer used. The module expects the following file structure

Rb I	775.7651	300
Rb I	775.9436	60
Rb I	780.027	90000
Rb I	792.526	5
Rb I	792.554	4
Rb I	794.760	45000
Rb I	827.141	40
Rb I	827.171	30
Rb II	860.396	2000
Rb I	886.8512	40

where the first line labels the ionization stage, the second the transition wavelength in nm, and the third the relative intensity. The module can handle files downloaded from the NIST atomic database.

Chapter 8

Optimizers

Here, we introduce five optimization algorithms, i.e. a simplex downhill combined with simulated annealing, two genetic algorithms, an evolutionary algorithm, and an adaptive algorithm. All algorithms are designed to optimize a single feedback signal, or fitness, by continuously varying a set of parameters, for example the phase values of a spatial light modulator. While the simplex downhill algorithm is strong for a low number of parameters and fails to converge for a large search space, the other four are well suited for large search spaces.

All optimizer vis call an experiment vi. Its input are the parameters (an array of numbers) to optimize and its output is a single feedback (a dimensionless number). The experiment vi is called by the optimizer vi whenever it needs to evaluate a new set of parameters. Its file and path information must be entered in the appropriate field. The experiment vi may consist of $\mathcal{LAB}^{\mathrm{II}}$ modules as shown in the following two examples. It may also communicate with real devices control their setting through the parameters and read out a real feedback signal, for example from a photodiode.

We first start with a virtual experiment for the simplex downhill algorithm, which is shown in fig.8.1

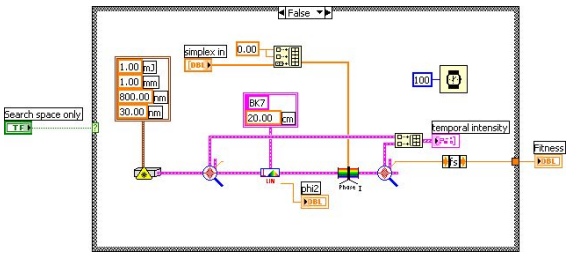


Figure 8.1: The virtual experiment.

A Gaussian shaped pulse is stretched by a linear dispersive medium and the task is to compress it through a pulse shaping apparatus. The parameters to optimize are the second to fourth order phase terms of a Taylor expansion. Note that the zeroth (absolute phase) and first order phase (group delay) are zero as they have no influence on the pulse width.

The feedback signal or the fitness is the FWHM of the signal. That is, the signal needs to be minimized in order to achieve the desired goal. Also note that the actual experiment needs to be located in the 'false' part of the vi. The 'true' section is called only once by the optimizer algorithm and this happens at the very beginning. Here, the boundaries of the search space have to be defined.

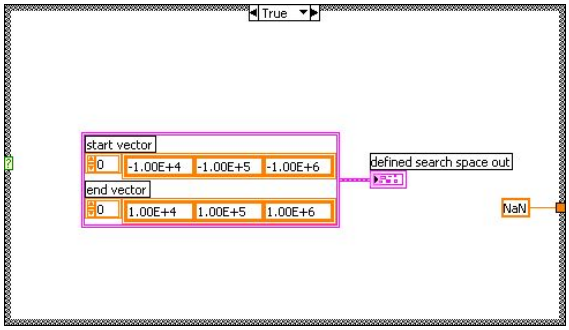


Figure 8.2: Definition of the search space.

Figure 8.2 shows the 'true' case where the search space for the optimization is defined. Allowed values for second order phases are $[-10^4 \ldots 10^4]$ fs^2, for third order phases $[-10^5 \ldots 10^5]$ fs^3, and for fourth order phases $[-10^6 \ldots 10^6]$ fs^4, respectively.

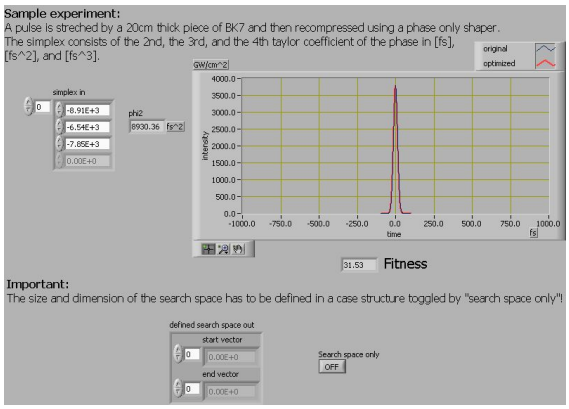


Figure 8.3: Front panel.

Figure 8.3 shows the front panel after optimization has been accomplished. The view graph shows the original and the optimized pulse shape. It is recommended to always open both the experiment vi, in order to follow the progress, and the optimizer vi. Arrange them so that both front panels are visible and follow their progress as the optimization process runs.

Things are quite similar for the other four optimizer vis. Again we will explain an example vi, as shown in fig. 8.4. The task is also quite similar. A Gaussian shaped laser pulse is stretched by a 2 cm thick piece of SF14 and compressed by a pixelated pulse shaper operated

in phase-only mode. The pulse shaper has 128 pixel where each pixel covers 0.75 nm giving a full shaper window of 96 nm which is roughly four times the FWHM of the spectrum. The center pixel coincides with the center wavelength of the laser. The pulse itself is sampled in manual mode with 1024 samples.

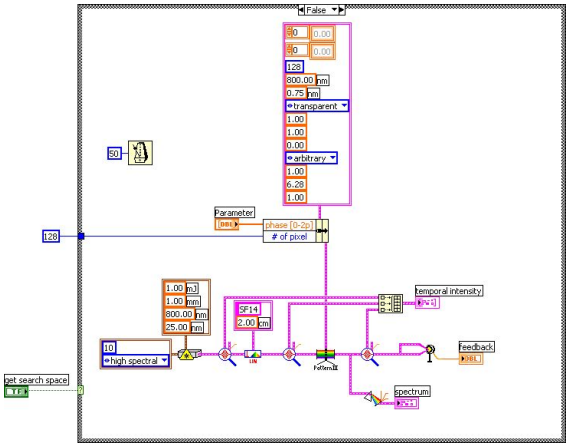


Figure 8.4: Virtual experiment.

Figure 8.5 shows the 'true' case structure where the boundaries of the search space are fixed. Here all 128 pixels may vary between 0 and a maximum phase of 2π.

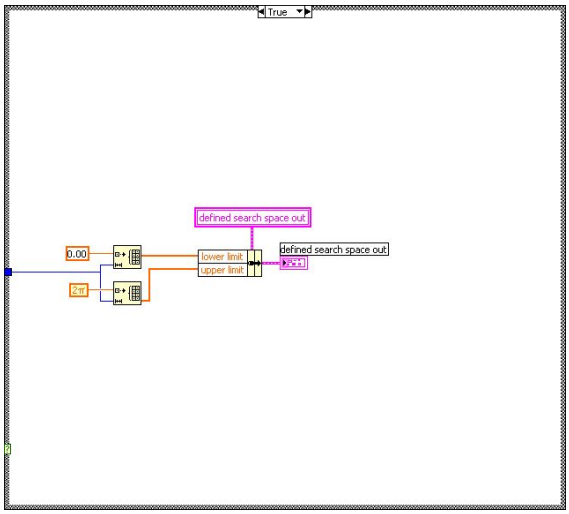


Figure 8.5: Definition of the search space.

The front panel is shown in fig 8.6. Besides the pulse spectrum we see the shaper setting, i.e. the value of all pixel in a bar graph, and the pulse before and after the glass window and after the shaper. As the search space is large convergence is usually reached only after many many iterations.

Use either one of the four 'genetic'-type algorithms together with the example experiment to see their performance. When you want to design your own experiment, use the experiment

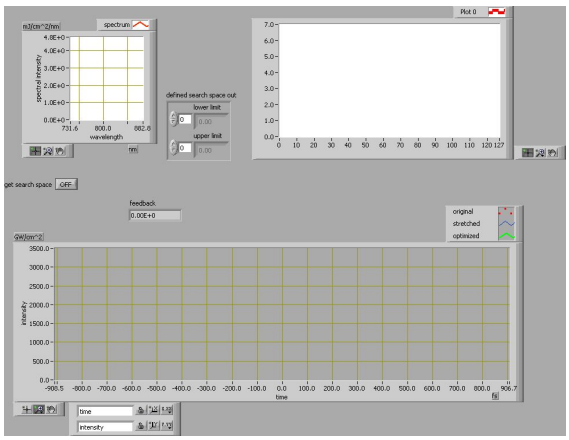
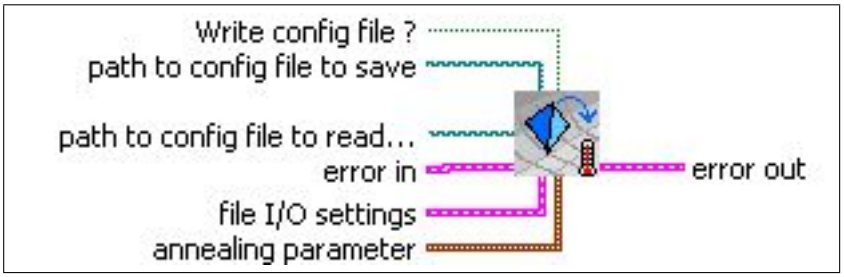


Figure 8.6: Front panel.

vi that comes with $\mathcal{LAB}^{\mathrm{II}}$, rename it, throw away everything except the input array and the feedback indicator, and insert your own code.

Simplex Downhill



©MH

Input **Output**

The `simplex downhill` is a combination of the deterministic simplex downhill algorithm and simulated annealing. Simulated annealing does contribute the stochastic part to the otherwise deterministic simplex downhill strategy and makes it a very powerful search algorithm for small search spaces.

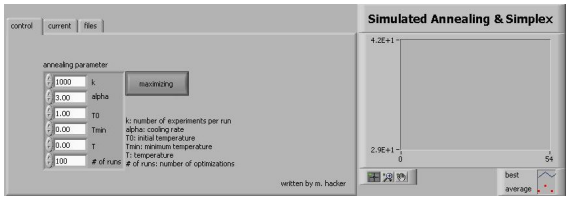


Figure 8.7: Front panel.

Figure 8.7 shows the first control tab. The input parameters control the simplex optimization. These are the annealing parameters k and α, which determine the rate with which the annealing temperature decreases.

$$T = T_{\min} + \begin{cases} T_0 \left(1 - \frac{j}{k}\right)^{\alpha} & j < k \\ 0 & j \geq k \end{cases}$$

where j is the number of the actual experiment. The temperature starts at T_0 and goes all the way down to $T_{\min}$. The input T is irrelevant. The number of runs determines how many subsequent optimizations are performed. A boolean button allows to determine whether the feedback signal or the fitness should be minimized or maximized.

Figure 8.8 shows the second control tab. It displays the actual number of the present run, the number of individual experiments that have been performed within this run, the final fitness of the run, and the best fitness of all runs performed so far. It shows the temperature

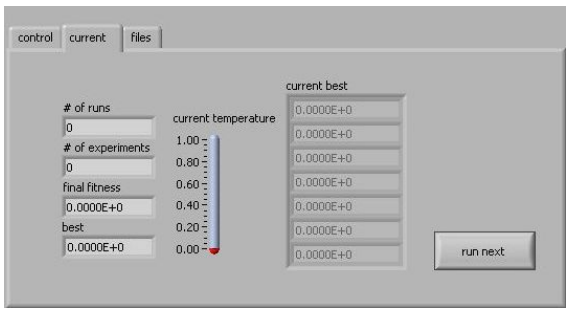


Figure 8.8: Front panel.

and the actual parameters. The button `run next` allows to manually start the next run without waiting for final convergence.

Figure 8.9: Front panel.

Figure 8.9 shows the file structure used. Most importantly the file information of the experiment vi that is used. Then a result file may be generated that saves the final parameters of all runs, and finally a log file that saves the results of all experiments of all runs. Optionally one may generate a config file that saves the actual optimization parameters for later use. If the config file load is not empty then the panel settings are overwritten by the settings of the config file.

Genetic Algorithm

Input	Output
none	none

The `genetic algorithm` is an optimization routine that relies on reproduction and mutation.

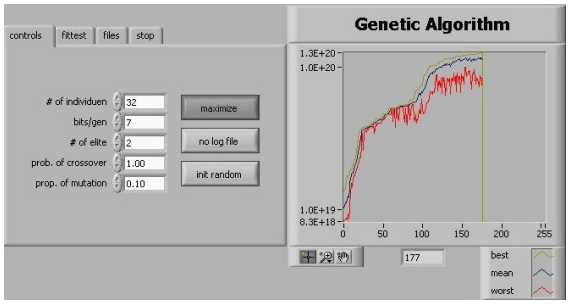


Figure 8.10: Front panel.

Figure 8.10 shows the first control tab. The input parameters are the number of individuals per generation, the bits/gene, the number of elite, the probability of crossover, and the probability of mutation. The bits/gene determine the resolution of the discrete sampling of the search space and the number of elite determines the number of the best individuals that move on from generation to generation without experiencing any crossover or mutation. Three buttons allow to set whether the feedback signal is minimized or maximized, whether a log file is generated, and whether all individuals of the first generation are created from random numbers or loaded from a previously saved individuum.

Figure 8.11 shows the second control tab. It displays the file structure. Most importantly the file information of the experiment vi that is used. Then a log file and finally the file which saves the best individuum.

Figure 8.12 shows the best individuum of the actual generation.

Figure 8.13 allows to select whether the optimization stops after a given number of generations or manually by pressing the stop button.

Figure 8.11: Front panel.

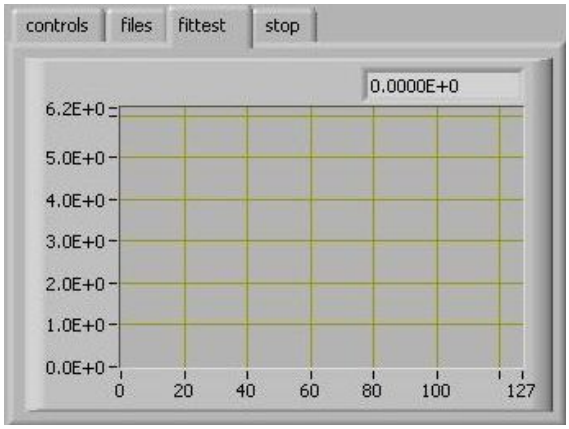


Figure 8.12: Front panel.

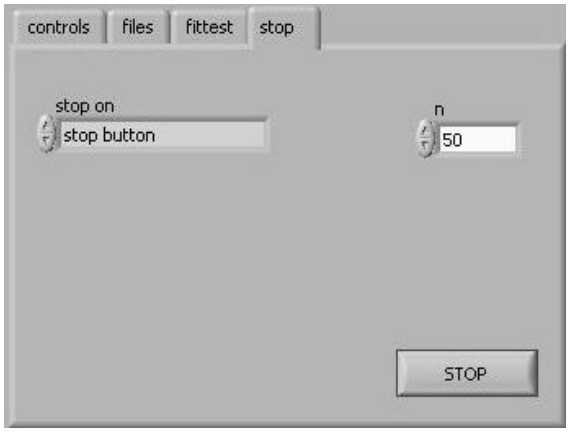


Figure 8.13: Front panel.

Genetic Algorithm 2

©GS

Input	Output
none	none

The `genetic algorithm 2` is a identical to the `genetic algorithm` except that the gens may be binned. Binning can be reduced manually as the optimization proceeds. For example, setting binning equal to two means that always two genes have the same value and the total number of genes reduces by a factor of 2.

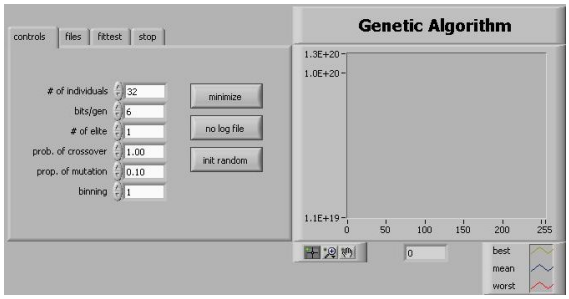


Figure 8.14: Front panel.

Evolutionary Strategy

©TF

Input	Output
none	none

The `evolutionary strategy` is an optimization routine that relies on mutation mainly. In contrast to the genetic algorithm the search space parameters are stored as floating point numbers.

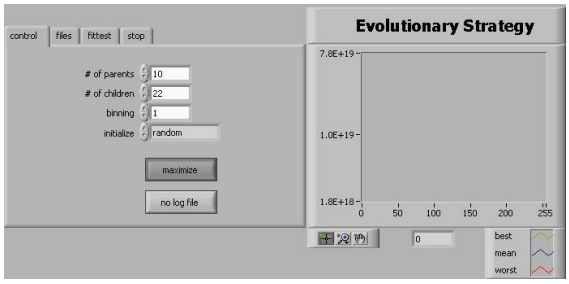


Figure 8.15: Front panel.

Figure 8.15 shows the first control tab. The input parameters are the number of parents per generation, the number of children produced per generation, the amount of binning, and whether the initial individuals are generated from random numbers or initialized from a file, i.e. a previously stored individuum. Two boolean buttons select whether the feedback signal is minimized or maximized and whether a log file is generated.

Figure 8.16 shows the second control tab. It displays the file structure. Most importantly the file information of the experiment vi that is used. Then a log file and finally the file which saves the best individuum.

Figure 8.17 shows the best individuum of the actual generation.

Figure 8.18 allows to select whether the optimization stops after a given number of generations or manually by pressing the stop button.

Figure 8.16: Front panel.

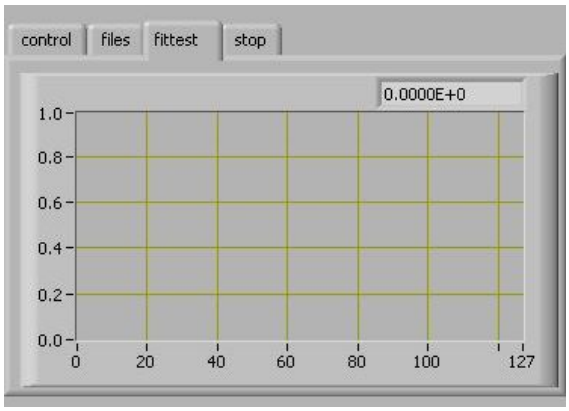


Figure 8.17: Front panel.

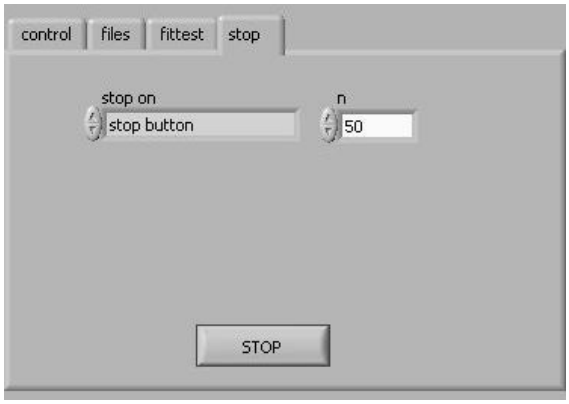


Figure 8.18: Front panel.

Adaptive Strategy

©TF

Input	Output
none	none

The `adaptive algorithm` is an optimization routine that relies on a large number of mutations. The mutation rate itself is adjusted according to the success rate.

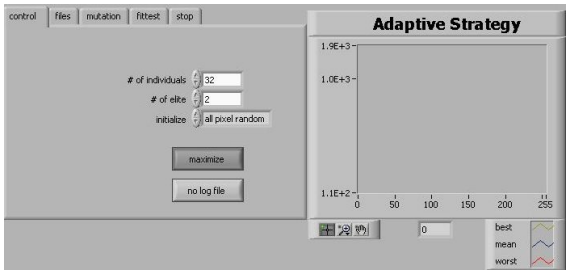


Figure 8.19: Front panel.

Figure 8.19 shows the first control tab. The input parameters are the number of individuals per generation, the number of elite, and whether all individuals of the first generation are created from random numbers or load from a previously saved individuum. Two boolean buttons select whether the feedback signal is minimized or maximized and whether a log file is generated.

Figure 8.20 shows the second control tab. It displays the file structure. Most importantly the file information of the experiment vi that is used. Then a log file and finally the file which saves the best individuum.

Figure 8.21 shows a statistics of the mutations. Individuum zero is the best, one the second best etc. In green are the successful mutations and in red the unsuccessful. A mutation is rated successful if the individuum can increase its rating by at least one rank in the next generation.

Figure 8.22 shows the best individuum of the actual generation.

Figure 8.23 allows to select whether the optimization stops after a given number of generations or manually by pressing the stop button.

Figure 8.20: Front panel.

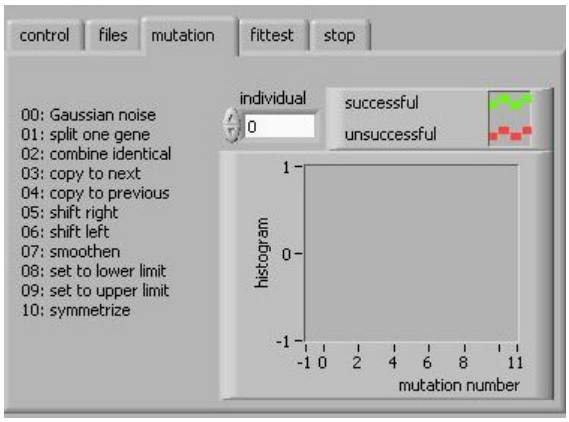


Figure 8.21: Front panel.

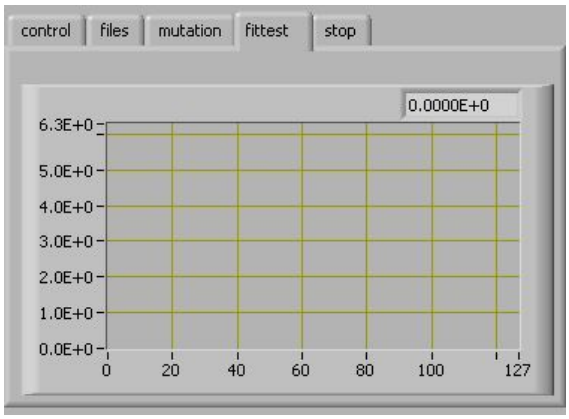


Figure 8.22: Front panel.

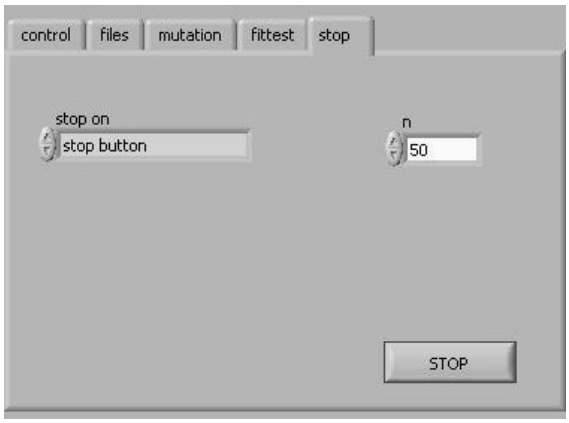


Figure 8.23: Front panel.