Summer Internship Project

Report

Python library for Dispersion of ultrashort pulses

Submitted by

Louis Lafforgue

Engineering master student

Ecole des Mines de Saint-Etienne

Under the guidance of

Markus Gühr

Team leader of the quantum experimental physics group

Institute of Physics and Astronomy

POTSDAM UNIVERSITY

Karl-Liebknecht-Strae 24/25, 14476 Potsdam

Summer Internship 2020

Contents

Chapter 1

Introduction

Context

As a first-year master student engineering and fundamentals physics, I realise a 10 weeks internship with the Experimental quantum physics group at Potsdam University. They study the interaction of light with matter at a femtosecond time scale. The group is performing ultrafast spectroscopy experiments. For that, they use pump-probe experiments with ultrashort pulses (10 fs). By sending ultrashort pulses to a matter sample (Molecules) and probing the different excitation level, one can study ultrafast phenomenons. The pulse is then taking snapshot of the dynamics, the shorter it is, the better is the resolution. Knowing the parameters precisely of the pulse is then very important if one wants to deduce the dynamics of the sample studied. Nevertheless, the pulse has to travel along with multiple optical components before reaching the sample and experience modification of its parameters.

Problem definition and objectives

Since 1960 ultrafast laser technologies didn't stop to improve, producing shorter and more powerful pulses. As example combining Q-switching and passive mode-locking technologies, a Titanium Sapphire laser can generate pulsation to 5fs. The shorter is the pulse, the more it is subject to interaction with matter. The interaction is very useful to study samples but could be very annoying when it's with the glass of a lens or other optical devices.
The main effect is dispersion through a transparent medium, resulting in an elongation of the pulse and chirping of the frequency. The pulse becomes longer and the experiments less precise.

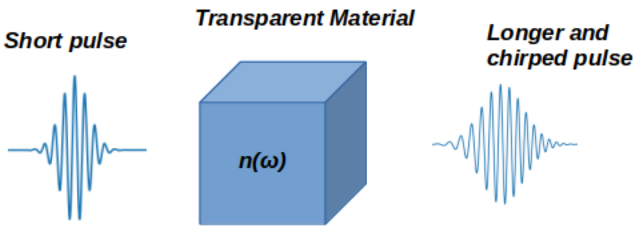


Figure 1.1: Dispersion of ultrashort pulse throught a medium

The objective during those 10 weeks was to write a python oriented object program computing the different change, enhanced by linear interactions, experienced by an optical pulse through a component. I implemented the propagation through a transparent medium for an arbitrative thickness, a double grating compressor and a double prism compressor. I only focus on linear effect as they occur from very low power and tried to bring as many details as possible for the characterisation of the pulse.

Chapter 2

physical description of the pulse

2.1 Representation

One can describe a light signal by its Electric Field $\overrightarrow{E}(t)$. For conveniencie, we use the complex representation $\overrightarrow{\mathcal{E}}(t)$ such that $2\overrightarrow{E}(t) = 2\Re(\overrightarrow{\mathcal{E}}(t)) = \overrightarrow{\mathcal{E}}(t) + \overrightarrow{\mathcal{E}}(t)^*$. Assuming linear polarization, this complex field can be written in term of module and argument $\mathcal{E}(z,t) = |\mathcal{E}(z,t)| \exp(i\phi(z,t))$ where $\phi(z,t)$ is called the temporal phase of the signal. Experimentally it's not possible to measure directly the field but the intensity $I = |E(z,t)|^2$.

A Gaussian pulse with duration t_0 of 10fs and central frequency ω_0 will have a Gaussian envelope with a constant phase with respect to z. The electric field will then be written: $\mathcal{E}(z,t) = A(z)\exp(-(t/t0)^2)\exp(i\omega_0 t)$

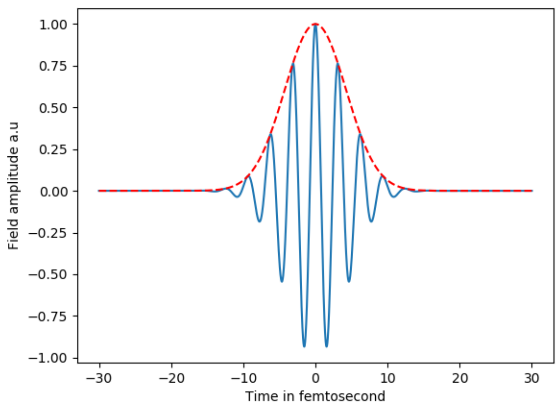


Figure 2.1: Temporal representation of a Gaussian pulse

From the preceding expression, the simplest electric field appears to be monochromatic waves, with a constant amplitude. ($\mathcal{E}(z,t) = A\exp(i\omega_0 t)$). Then, to make the equation easier,it is useful to write the pulses as a superposition of monochromatic waves. For that, we use the Fourier transform.

$$\mathcal{E}(t,z) = \int \mathcal{E}(\omega,z)e^{-i\omega t}\frac{d\omega}{2\pi} \leftrightarrow \mathcal{E}(w,z) = \int \mathcal{E}(\omega,z)e^{i\omega t}d\omega$$

As we did in the time domain, it is possible to write the complex field in the frequency domain in term of module and argument, introducing the spectral amplitude and phase. $\mathcal{E}(\omega,t) = |\mathcal{E}(z,t)|\exp(i\phi(z,\omega))$. From the porperties of the Fourier transform, the inequality $\Delta t\Delta\omega \geq \frac{1}{2}$ as to be respected. Thus, the shorter is the pulse, the wider is the spectrum and inversely.

In a medium, the electromagnetic wave will induce a polarisation of the medium $\overrightarrow{P}$and a magnetic field $\overrightarrow{H}$. Thus, we can characterise the wave and the effect by the displacement vector $\overrightarrow{D} = \varepsilon_0\overrightarrow{E} + \overrightarrow{P}$. Finally, we recall that in a medium, fields are determined by maxwell's equations:

$$\begin{aligned}\nabla.\overrightarrow{D} &= 0\\ \nabla.\overrightarrow{B} &= 0\\ \nabla\times\overrightarrow{E} &= -\frac{\partial\overrightarrow{B}}{\partial t}\\ \nabla\times\overrightarrow{H} &= -\frac{\partial\overrightarrow{D}}{\partial t}\end{aligned} \tag{2.1}$$

2.2 Propagation

Knowing the field and the equation driving it, we can derivate the propagation equation. We take the curl of the last equation and we get:

$$\nabla^2\overrightarrow{E} - \frac{1}{c^2}\frac{\partial^2\overrightarrow{E}}{\partial t^2} = \mu_0\frac{\partial^2\overrightarrow{P}}{\partial t^2} \tag{2.2}$$

The idea is to know the pulse properties after crossing a material for a distance z, knowing the initial pulse. For that, we will express the equation 2.2 in the frequency space, in which, it admits an easy analytical solution . We use $\frac{\partial^2}{\partial t^2} \leftrightarrow -\times\omega^2$. All that remains is to compute the Fourier transform to get the temporal expression.

We consider only the linear polarization of the material which can be written: $\overrightarrow{P}^{(1)}(t) = \epsilon_0 \chi^{(1)}(t)\overrightarrow{\mathcal{E}}$. We can then introduce, for isotropic materials, the refractive index $n^2(\omega) = (1 + \chi^1(\omega))$. From those considerations, we can deduce Helmholtz Equation ([2] for more details), available for a transparent medium without loss.

$$\left(\nabla^2 + k^2(\omega)\right)\overrightarrow{E}(\overrightarrow{r},\omega) = 0 \quad with \quad k^2(\omega) = \omega^2 n^2(\omega)/c^2 \tag{2.3}$$

If we considere a lineary polarized electric field, solving 2.3 gives in the frequency space the solution : $E(z,\omega) = |E(z=0,\omega)| \exp(i\varphi(0,\omega) + ik(\omega)z)$.
In material, we can express the refractive index as a function of the wavelength λ thanks to the SellMeier equation (next chapter), Nevertheless, when one wants to express it as a function of ω things become more complicated as the relation between ω and λ depends on $n(\omega)$. As we are treating with ultrashort pulses, we can't consider $k(\omega)$ as a constant. Using a Taylor limited development of k, we can have a good approximate solution. The final expression is then:

$$E(z,\omega) = |E(z,\omega)| \exp[i\varphi(0,\omega) + i(k_0 + k_0'(\omega-\omega_0) + \frac{1}{2}k_0''(\omega-\omega_0)^2 + ...)z] \tag{2.4}$$

The sorter is the pulse, the more the development will have orders for an accurate result. In the program, We stopped at the third order for the medium and the second for the other components

Example For an initial Gaussian pulse, we can easily calculate the second-order analytical expression of the pulse, after propagation of a distance z [1]. The initial pulse has the form $\mathcal{E}(t) = \mathcal{E}_0 \exp(-\Gamma t^2 + i\omega_0 t)$ We can easily go to the time domain and get the spectrum

$$\mathcal{E}(\omega) = \mathcal{E}_{\omega_0} \exp(-\frac{(\omega-\omega_0)^2}{4\Gamma})$$

After a distance z, the pulse is $\mathcal{E}(z,w) = \mathcal{E}_{\omega_0} \exp[-ik_0 z - ik_0'(\omega-\omega_0) - (\frac{1}{4\Gamma} + \frac{ik_0''}{2})(\omega-\omega_0)^2]$ going to the Fourier transform, the final temporal expression without the delay due to the group velocity is

$$\mathcal{E}(z,t) = \sqrt{\frac{\Gamma(z)}{\pi}}\mathcal{E}_0 \exp(-\frac{\Gamma}{1+\xi^2 z^2}t^2 + i\frac{\Gamma\xi z}{1+\xi^2 z^2}^2) \tag{2.5}$$

$$with \quad \frac{1}{\Gamma(z)} = \frac{1}{\Gamma(z)} + 2ik_0''z \quad and \quad \xi = 2\Gamma k_0''$$

The real part of the phase is inversely proportional to z and the complex part is z dependant. Since the different spectral components travel at different speeds (blue travels faster than red), the pulse will experienced chirping of frequency and a temporal broadening. Indeed, in the analytical expression, the gaussian coefficient becomes smaller with z. This is illustrated in 1.1.

Chapter 3

Work done

The main idea of the program is to take an analytical or experimental spectrum, compute the propagation through a medium with 2.4 and calculate the Fourier transform to have the temporal pulse shape. The different orders of k are calculated numerically with the refractive index of the material. We will detail the different implementation of the components and the pulse object.

3.1 Pulse implementation

As we have seen in the preceding part, a pulse can be fully characterized either in the frequency or time domain, knowing its amplitude and its phase. We use an object-oriented programming, enabling a better implementation of the different devices. The full organigram can be found in the annexe. The python *Pulse* object contains 7 relevant attributes for the characterization: 6 arrays of size *N_signal* (number of points for the discretization) and one dictionary parameters.

X	frequency x-axis in 10^{15}rad/s
Y	frequency amplitude in a.u
phase	spectral phase in rad
X_time	Time x-axis in fs
Y_time	Time amplitude in a.u
temporal_phase	temporal phase in rad
parameters	[”w0”=2, ”HMD”=10, ...]

The variable *phase* and *temporal_phase* are only used for the propagation or the initial creation of the variable. They are not updated during each Fourier transform to keep the shape. There are mainly two ways of implementing a pulse object.

Analytical input The different arrays will be automatically implemented with the analytical values of different pulse shapes.

shape	temporal envelope	spectral envelope
Gaussian	$\mathcal{E}_0 \exp[-\frac{4ln(2)}{\tau_{FWHM}^2} t^2]$	$\mathcal{E}_0 \sqrt{\frac{4ln(2)\pi}{\tau_{FWHM}^2}} \exp[-\frac{\tau_{FWHM}^2}{16ln(2)} (\omega - \omega_0)^2]$
Hyperbolic secant	$\mathcal{E}_0 / \cosh(2.634 \frac{t}{\tau_{FWHM}})$	$\mathcal{E}_0 \frac{\pi \tau_{FWHM}}{2.634} / \cosh(\frac{\pi}{2 \times 2.634} \tau_{FWHM} (\omega - \omega_0))$
Lorentzienne	$\frac{\mathcal{E}_0}{1+(1.287t/\tau_{FWHM})^2}$	$\mathcal{E}_0 2\pi \frac{\tau_{FWHM}}{1.287} \exp(\lvert \frac{\tau_{FWHM}(\omega - \omega 0)}{1.287} \rvert)$

The subclass *Analytics* is used and the constructor needs different parameters

```
__init__(self, shape="Gaussian", w0=2, E0=1, FWHM=10, poly_phase
    =[0], N_grid=2**15, N_signal=2**11, **kwargs)
```

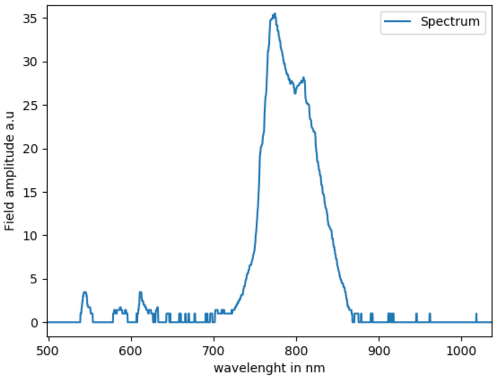
Where *FWHM* is the half maximum duration or τ_{FWHM}, *N_grid* corresponds to the grid size for the Fourier transform, must be bigger than *N_signal*, number of points of the signal. *poly_phase* stands if one wants to add an initial temporal phase to the signal. More details are present in the library documentation and examples.

Data input The program allows to implement a pulse object from experimental data, from ".dat" or ".csv" files. Nevertheless, most of the experimental spectrums are expressed as a function of the wavelength and contain noises. A preprocessing has to be done before implementing the pulse object.

- convert the axes. We use The relation $\lambda = 2\pi c/\omega$ and apply a Jacobian scaling $S_\omega(2\pi c/\lambda) = S_\lambda \frac{\lambda^2}{2\pi c}$ [7, p.15] to the spectrum.
- create linear spacing frequency tab and add zeros to have an axis starting from 0.
- Realize an interpolation with *scipy.interpolate* to have a spectrum corresponding to the new frequency axe.
- Apply a filter and remove the negative values. A scipy savgol filter is used with N=51, making a good smoothing without loosing too much information.
- As experimentally we have an Intensity spectrum, we take the squared root to implement the pulse with the field amplitude.

Finally, the inverse Fourier transform is applied to the signal to implement *X_time* and *Y_time*. The Fourier transform is applied with *N_grid* points, creating a 0-padding for a better resolution.

The different parameters are calculated through two function *SetTimeParameters* and *SetParameters*. They are then stored in the dictionary *parameters*



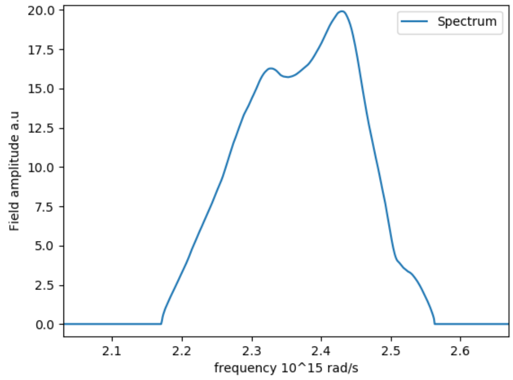


Figure 3.1: left : Initial experimental spectrum, right: Spectrum after processing

3.2 Components

Now that the pulse is implemented, we can compute the interaction with the different optical devices or matters.

3.2.1 Transparent medium

For linear interaction, a transparent medium is characterized by its refractive index given by the experimental Sellmeier equation

$$n^2(\lambda) = A + \sum_i \frac{B_i \lambda^2}{\lambda^2 - C_i}$$

. We represent this equation as an array, stored in a dict *media* from the class *Matter*.

media={ "matter_name":[A,[B1,C1],[B2,C2]...], ...}

The function *calculate_indice* is used to calculate the refractive index for a given wavelength. Knowing the refractive index, we can then compute the propagation of the pulse thanks to 2.4. The sequence 3.2 of instruction is executed by the program.

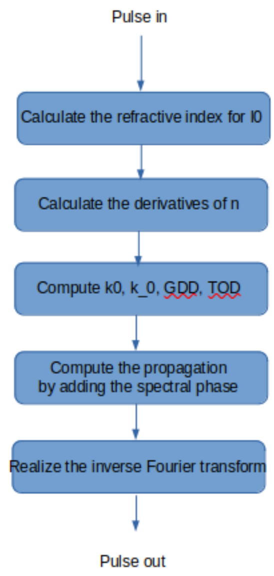


Figure 3.2: Sequence of instructions where $K0 = k(\omega_0), k\_0 = \frac{dk}{d\omega}|_{\omega_0}$. The GDD (group delay dispersion) is $z\frac{d^2k}{d\omega^2}|_{\omega_0}$ and TOD(the third-order dispersion) $\frac{d^3k}{d\omega^3}|_{\omega_0}$

Nevertheless, it's not possible to calculate them directly as ω is also λ dependant. We can use a variable change $\frac{df}{d\omega}|_{\omega_0} = \frac{df}{d\lambda}\frac{d\lambda}{d\omega}|_{\lambda_0}$ and we obtain

$$\begin{aligned} k\_0 &= \frac{1}{c}(n(\lambda) - \frac{dn}{d\lambda})|_{\lambda_0} \\ GVD = GDD/z &= \frac{\lambda^3}{2\pi c^2}\frac{d^2n}{d\lambda^2}|_{\lambda_0} \\ TOD &= -(\frac{\lambda}{2\pi c})\frac{1}{c}(3\lambda^2\frac{d^2n}{d\lambda^2} + \lambda^3\frac{d^3n}{d\lambda^3})|_{\lambda_0} \end{aligned} \tag{3.1}$$

From the Sellmeier equation, the derivatives of $n(\lambda)$ are easily numerically computed. We can then apply the propagation equation.

3.2.2 Double grating compressor

Another way to create dispersion is the geometrical dispersion. For that, we make the different spectral components travel at the same velocity but for different distances. A pulse after the propagation through a medium is positively chirped. meaning that the red components are at the front and the purple at the back. If we manage to make the red component travel longer than the purple one, then we can compensate for dispersion. The pulse will be negatively chirped. Different devices can be used for that. Here the grating compressor.

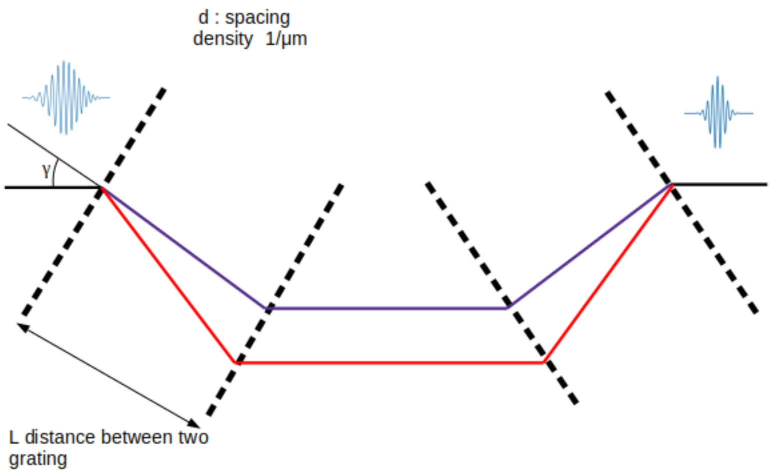


Figure 3.3: Double-parallel grating compressor

The diffraction angle of a grating is lambda-dependant. Thus, the different spectral components of the pulse will not have the same diffractive angle, and, if one adjusts properly, we can obtain a negative dispersion coefficient. Nevertheless, it's not possible to compensate both, the second and the third-order dispersion. Here, only the the second order is calculated . The GDD can be calculated anatically if one knows the optical path P with, $GDD = \frac{dP(\lambda)}{d\lambda}$ [**?**]. It finally gives [6] the expression:

$$GDD = -\frac{m^2\lambda_0^3 L}{2\pi c^2 d^2}\left[1 - (-m\frac{\lambda_0}{d} - sin(\gamma))^2\right]^{-3/2} \tag{3.2}$$

With m the diffraction order, often $m = -1$. L the distance between the two gratings, d the density spacing of the grating and γ the angle of incidence

3.2.3 Double prism compressor

The concept behind the prism compressor is close to the grating. We apply a geometric dispersion thanks to the lambda-dependent diffraction angle. It allows a better compacity compare to a grating compressor. But, as the pulse travels through a glass, one has to take the dispersion due to a medium into consideration in the calculation. The notations used in the program are resumed in Annex A. The general configuration is presented 3.4. By geometrical considerations [3] we get

$$GDD = \frac{2 * \lambda_0^3}{2\pi c^2}[l_g \frac{d^2n}{d\lambda^2} - (4\overline{BC} + l_g/n^3)] \frac{dn}{d\lambda}^2 |_{\lambda_0} \tag{3.3}$$

Where l_g stands for the travel length in the glass $\overline{AB} + \overline{CD}$

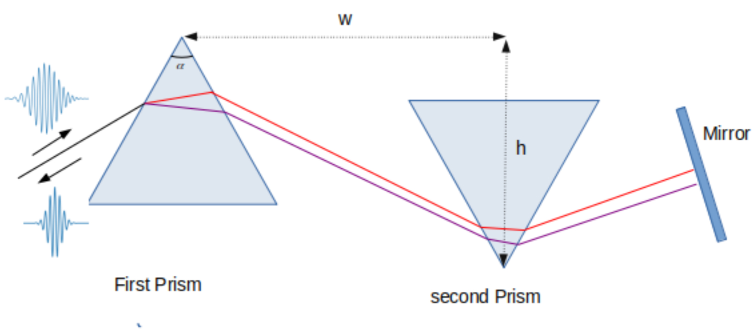


Figure 3.4: Double prism compressor

3.3 Examples and results

All the code can be found at To test the accuracy of the calculation for a transparent medium, we first compare the propagation of a gaussian pulse with the analytical expression 2.5. The test is written in the file *test.py*, *test_propagation*. We find a perfect fitting in between the two results. We run then the example *gaussian_propagation()* . The results can be seen in 3.5

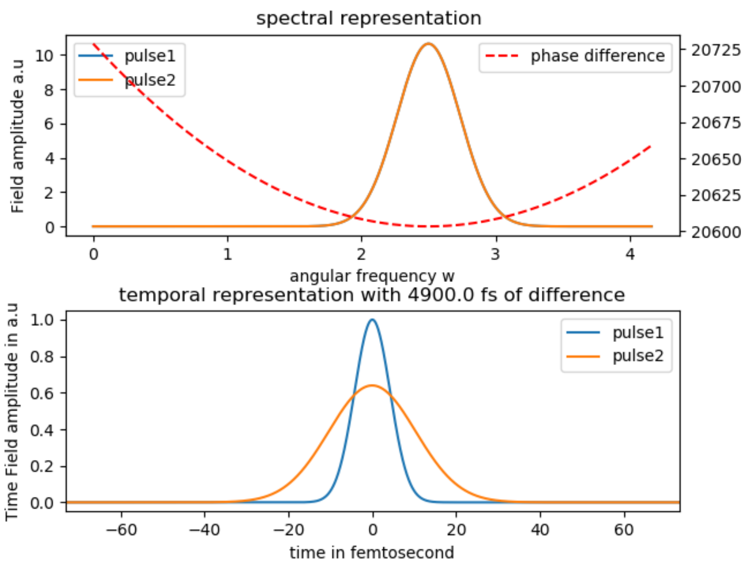


Figure 3.5: result window for the method *comparedouble*, propagation of a Gaussian pulse in 1mm of SiO2 glass

The numerical calculation matches the result from the theoretical approach, the pulse is broader and experienced a quadradic spectral phase. If one plots the real temporal value, the frequency chirped is observed.

It's now possible to run the program with an experimental spectrum such as 3.1 and get the pulse after the propagation. The complete program can be found in the utilisation notes in Annex or Github.

Some tests were also designed to measure the accuracy of the prism and the grating compressor. The GDDs calculated by the program are the same than for the analytical calculations.

Future Work

3.4 Adding Devices

3.4.1 Refractive index

The refractive index is now implemented thanks to the Sellmeier equation through the different coefficients. We may want to use an exotic material, without Sellmeier equation for the refractive index. Future work would be to implement the refractive index as a *callable*, taking the wavelength as an argument and returning the refractive index. Thus, the refractive index could be implemented externally and used by the library.

3.4.2 Compressor

We presented two components used to compensate for dispersion of ultrashort pulses. We took the most basic geometry to make easier the implementation. We can design a compression device form of both, prism and grating, enabling in some cases a better compression. Furthermore different geometries for prism exist, non-parallel alignment is used to optimize the space. Another device is also commonly used, the chirped mirror [3, p.24].

3.5 Non-linear optics and Split-steps method

In section 2.2 we only considered the linear polarization. However, some materials reveal non-linear polarization for high field intensity and 2.3 no longer holds. Starting from 2.2 and separating the linear to the non-linear polarization one gets

$$\nabla^2 \overrightarrow{E} - \frac{1}{c^2}\frac{\partial^2 \overrightarrow{E}}{\partial t^2} - \mu_0 \frac{\partial^2 \overrightarrow{P^{Lin}}}{\partial t^2} = \mu_0 \frac{\partial^2 \overrightarrow{P^{NL}}}{\partial t^2}$$

From the previous parts, we know how to solve the left part, moving to the frequency space and using a limited development. The non-linear part needs advanced tensor calculus and can't be calculated in the general case. To illustrate, we take a centrosymmetric third order material characterised by its $\chi^{(3)}$. In complex representation, the non linear polarisation can be written $\overrightarrow{P}^{(3)}(t) = \epsilon_0 \chi^{(3)}(t) |\overrightarrow{\mathcal{E}}|^2 \overrightarrow{\mathcal{E}}$. This term is responsible the self-modulation phase, creating a spectral spectrum broadening, very interested to compress pulses in the Fourier transform limit. Now, if we consider both dispersion and self-phase modulation, assuming paraxial approximation (envelop slowly varying compared to the pulse) and introducing the amplitude $A(t,z) = |\mathcal{E}(z,t)|$, we get the simplified Non-Linear Shrodinger equation (NLSE)

$$\frac{\partial A(z,t)}{\partial z} = -\frac{ik_0''}{2}\frac{\partial^2 A(z,t)}{\partial t^2} + i\frac{3\omega_0\chi^{(3)}}{8n_0 c}|A(z,t)|^2 A(z,t) \tag{3.4}$$

Here we considered only the second-order dispersion to simplify the equation. 3.4 can be rewritten in term of operators

$$\frac{\partial A(z,t)}{\partial z} = \hat{D}A(z,t) + \hat{N}A(z,t) \tag{3.5}$$

If we consider only the non-linear part 3.4, it can be easily solved numerically. We demonstrated that the linear part is solved in the frequency space. However, $\hat{D}$ and $\hat{N}$ don't commute. One way used to solve 3.4 is to split the sample into little steps and apply alternatively linear and non-linear propagation. This method is called split-step Fourier methods [5]. The linear propagation is solved in the Fourier space and the non-linear one in the temporal space. The pulse object implemented before is then perfectly suitable because it treats both the temporal and the frequency representation of the pulse. Thus, combining the object *DoublePrismCompressor* and *Pulse*, it could be then numerically possible, to find the output pulse, after spectral broadening by a non-linear fiber and a compression stage as did in [4].

Chapter 4

Conclusion

During this internship, I have had the opportunity to implement a python program designed to ease the propagation calculation, through a medium, of an ultrashort pulse. We saw that propagation can be easily solved and discretized in the Fourier space, enabling the numerical calculation of an experimental spectrum measured in the lab. I used a programming oriented object for the pulse and the matter, making it modulable. A large number of parameters are accessible just in the pulse object description. Two second-order compressor devices were implemented, a grating and a prism compressor. The program showed multiple ways of improvement and new functions will be implemented if needed. Thus, the library provides a tool which can be easily used in a lab to predict the pulse properties in an experiment. In addition to a deepening of *Python* language and physics of ultrashort pulse, this internship has been an opportunity to better understand the generation of ultrashort pulses and the phenomenon enhanced by them in the Linear and non- linear regime.

Bibliography

[1] C. Hirlimann. *Femtosecond Laser Pulses*, chapter Pulsed Optics, pages 57–87. springer, 2005.

[2] Manuel Joffre. *Cours d'optique non lineair en rgime continu et femtosecond*. Ecole polytechnique, 2014.

[3] Prof. Dr. Stefan Karsch. *Generation of ultraintense Laserpulses*. LMU, Mnchen, 2013.

[4] Mayer D. & Matthaei C.T & Heuer A. & Gühr M. Kagome-fiber prism compressor combination for yb:kgw laser pulsecompression to sub-40 fs. *Journal of optics*, 2019.

[5] S. W. Pool R. Deiterding. Robust split-step fourier methods for simulating the propagation of ultra-short pulses in single- and two-mode optical communication fibers. 2017.

[6] Clara Saraceno. Pulse stretching and compressing using grating pairs. *Ibsen Photonics*, 2007.

[7] Rick Trebino. *Frequency-Resolved Optical Gating: The Measurement of Ultrashort Laser Pulses*, chapter Ultrashort Laser Pulses, pages 11–35. springer, 2000.

Acknowledgment

I would like to express my gratitude to my internship supervisor Markus Gühr. To the university of Potsdam and Ecole des mines des Saint Etienne for having made possible this project. My home institution mentor, Dr. Rene Charrire. And finaly my thanks go to the Erasmus grant organisation for having provided me financial support.

Prism-compressor configuration

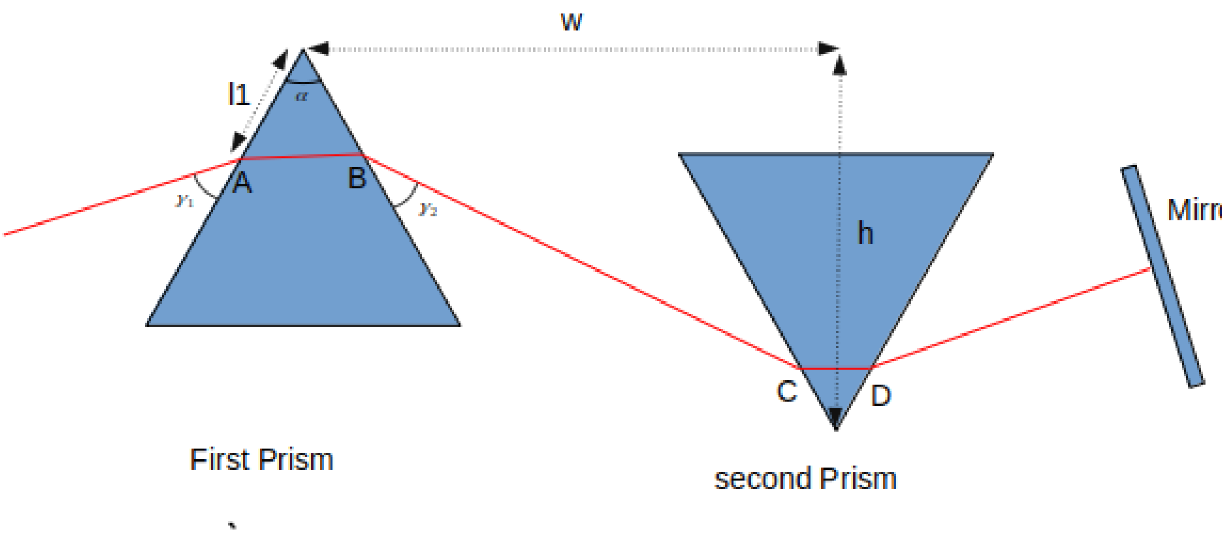


Figure 1: Notation used in the *DoublePrismCOmpressor* object

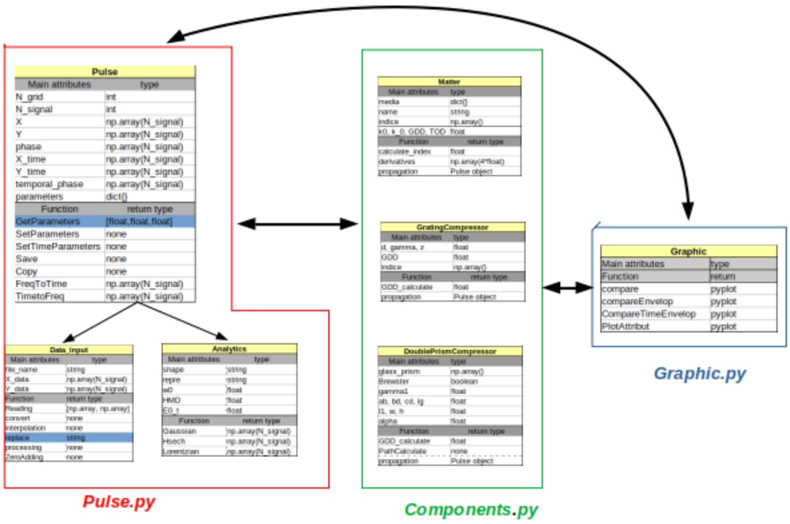


Figure 2: Notation used in the *DoublePrismCompressor* object