



Femtosecond Pulse Shaping and Characterization: From Simulation to Experimental Pulse Retrieval Using a Python-Based User Friendly Interface

Sébastien J Weber, Romain Gêneaux

► To cite this version:

Sébastien J Weber, Romain Gêneaux. Femtosecond Pulse Shaping and Characterization: From Simulation to Experimental Pulse Retrieval Using a Python-Based User Friendly Interface. Jérémie Léonard; Charles Hirlimann. Ultrafast Laser Technologies and Applications, EDP Sciences, pp.270, 2022, 978-2-7598-2719-0. 10.1051/978-2-7598-2719-0.c005 . hal-03727162

HAL Id: hal-03727162

<https://hal.science/hal-03727162>

Submitted on 17 Aug 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Chapter 5

Femtosecond Pulse Shaping and Characterization: From Simulation to Experimental Pulse Retrieval Using a Python-Based User Friendly Interface*

Sébastien J. Weber^{1,**} and Romain Gêneaux^{2,***}

¹CEMES-CNRS, Université de Toulouse, 29 rue Jeanne Marvig, 31055 Toulouse, France

²Université Paris-Saclay, CEA, CNRS, LIDYL, 91191 Gif-sur-Yvette, France

5.1 Introduction

Measuring short laser pulses is a mandatory and difficult task when dealing with experimental ultrashort measurements. To overcome the non-existence of fast enough measuring devices at the picosecond or femtosecond scale, many techniques have been invented. Most of them rely on a non-linear interaction to gain information on the spectro-temporal features of the laser pulse, and yield the complete phase and amplitude profile of the measured pulse. This spectro-temporal information sought is an experimental equivalent to Gabor or Wigner functions presented in the previous chapter. So far, each invented technique has required one specific algorithm, experimental layout, and acquisition program to retrieve pulse characteristics.

The recently developed COPRA approach¹ solves the first of these issues: it provides one algorithm for all methods after noticing that most of these algorithms

*Supplementary electronic material available at: https://github.com/CEMES-CNRS/pymodaq_femto.

**sebastien.weber@cemes.fr

***romain.geneaux@cea.fr

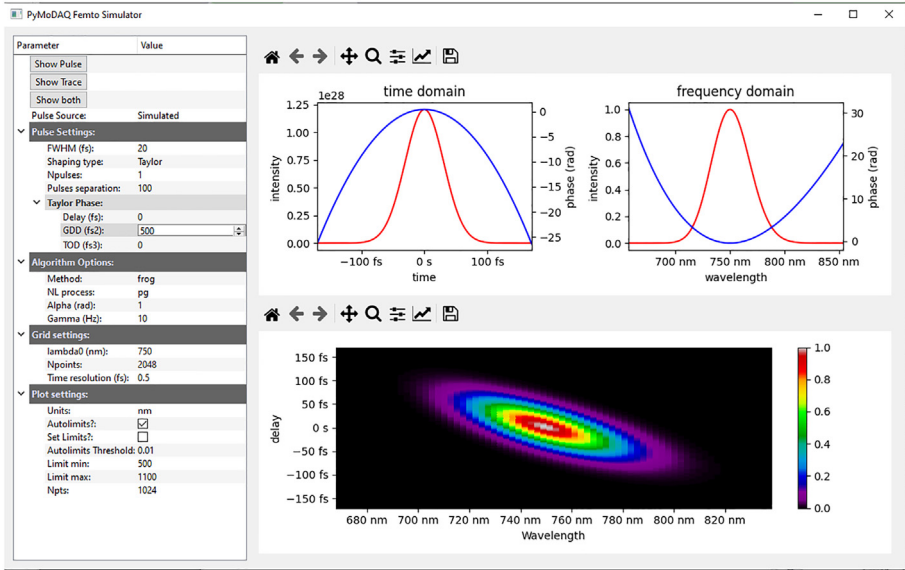


FIG. 5.1 – Simulator user interface in the case of a phase shaping using Taylor expansion with a GDD of 500 fs^2 . Left panel: User settings to set the initial pulse parameters, the shaping type (here Taylor), the simulated experimental technique (here FROG with its third-order non-linearity polarisation gating variant) and some plotting options. Top right panel: temporal and spectral representation of the shaped pulse. The intensity is plotted in red (left axis), the phase in blue (right axis). Bottom right panel: time–frequency representation using the selected experimental technique (FROG) and non-linearity (PG).

share a common mathematical formulation. Indeed, all methods do measure the light spectrum after a non-linear process, as a function of a parameter (delay, dispersion...). COPRA formulates these methods together as “Parameterized Nonlinear Process Spectra” (PNPS). The COPRA article¹ gives a very detailed analysis of the reconstruction mathematical problem where both concepts and comparisons between existing algorithms are presented.

With PyMoDAQ-Femto, we take a step further and incorporate this algorithm (available in the Python for Pulse Retrieval *PyPret*² package published under MIT license) within the PyMoDAQ³ framework, in an attempt to provide an open-source and widely applicable application for pulse retrieval and simulation purposes. PyMoDAQ-Femto (available on GitHub⁴) is a PyMoDAQ extension featuring a stand-alone user interface for simulating various experimental characterization techniques and a stand-alone user interface to perform pulse retrieval using the COPRA algorithm. Because it can be integrated as a PyMoDAQ extension, all PyMoDAQ features for data acquisition can be used to produce experimental PNPS traces, which are then processed by the retrieval extension. Measurements performed by other means can also be used if converted appropriately⁵.

PyMoDAQ-Femto is therefore a unified interface for acquiring and processing non-linear pulse characterisation traces. The following sections focus on how to use both the simulator and the retriever on specific examples. More specific details can be found on the documentation websites^{6,7}.

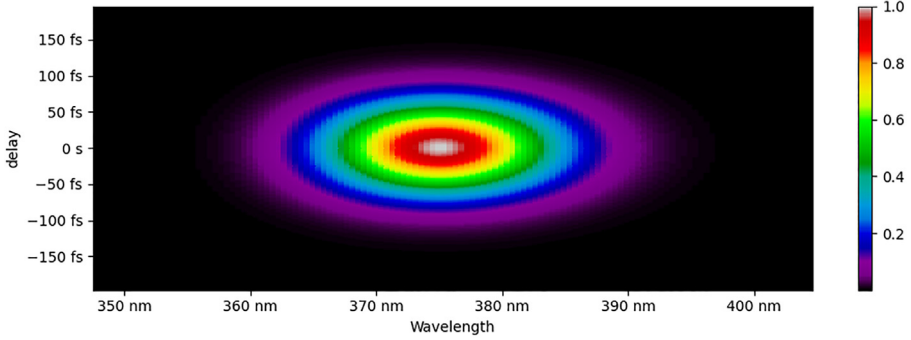


FIG. 5.2 – SHG-FROG trace for a 20 fs FWHM pulse duration having a GDD of 500 fs^2 .

5.2 Simulation and Retrieval of Synthetic Laser Pulses

PyMoDAQ-Femto has two main python modules: the *Simulator* and the *Retriever*. Both will be illustrated here using two typical shaped femtosecond pulses. The first one has second-order and third-order spectral phase due, for instance, to dispersion during propagation in air and materials. The second one presents a Gaussian temporal phase due to self-phase modulation typically used for production of ultrashort pulses ($< 10 \text{ fs}$).

5.2.1 Pulse Shaping and Field Representation

As stated in chapter 4, the temporal profile of a light pulse is obtained by Fourier transforming its complex spectrum. The spectral intensity $I(\omega)$ is a relevant quantity but it does not contain all the information about the pulse. For example, let us take two pulses sharing the same spectral intensity, one with a constant spectral phase and the other with a quadratic one. This leads to different temporal lengths: Fourier limited for the former and broadened for the latter.

A Fourier-limited Gaussian narrow-band pulse and a chirped broadband Gaussian pulse can have the same spectral intensity. However, these two pulses have a different spectral phase, constant for the former, quadratic for the latter. To aid with visualizing the interplay between time and spectral features, time-frequency representation has been introduced, such as the Gabor analysis or the Wigner distribution. Time-frequency distributions show the temporal evolution of the instantaneous frequency of a pulse, exactly like a music score where different notes (*i.e.* frequency components) are depicted as a function of time.

The goal of experimental characterisation techniques is therefore to obtain as much information as possible in the form of a time–frequency representation in order to extract valuable information from the pulse. Among the zoology of experimental time–frequency techniques, FROG⁸ and d-scans⁹ are widely used ones. This section will focus on simulation of FROG traces while the section on experimental data will focus on the d-scan technique. In the FROG technique, the pulse to characterize is split into two delayed replicas recombined into a non-linear process (second harmonic generation for its most widespread variant) where the non-linear signal outcome is spectrally resolved. By varying the delay, one obtains a FROG time–frequency trace.

5.2.2 Simulation Module

The main purpose of this module is to become familiar with pulse shaping and associated time–frequency non-linear traces one can obtain using various techniques. It is also used to test the efficiency of the *Retriever* module.

The initial pulse used in this paragraph has a Gaussian shape. In the *Simulator* module, the initial conditions can be set in the *Settings* panel (see figure 5.1 left), in the *Pulse Settings* section comprising Fourier limited pulse duration and the possibility to define a sequence of pulses. Two types of shaping are available at the moment, Taylor spectral phase expansion or Gaussian temporal phase.

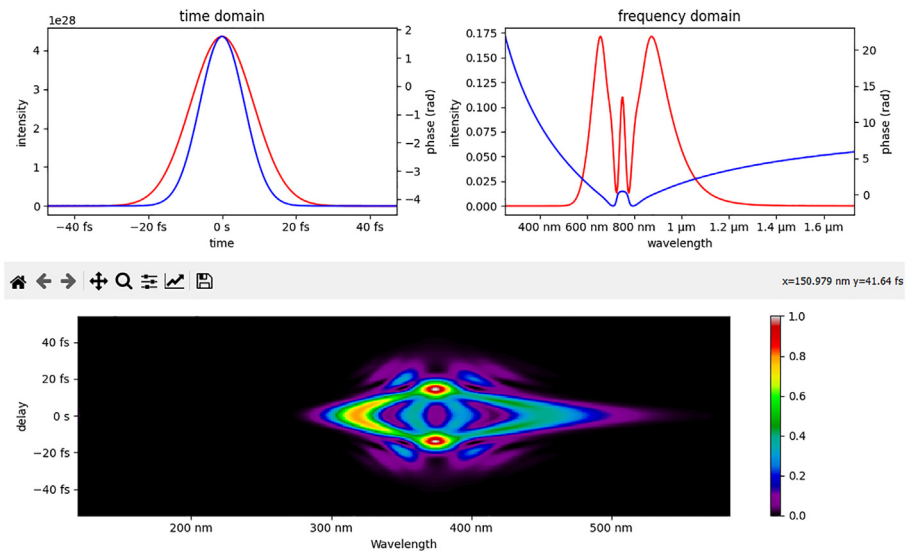


FIG. 5.3 – Time–frequency representation of a pulse which experienced self-phase modulation using SHG-FROG. On the top row, intensity and phase are shown in red and blue, respectively.

5.2.2.1 Taylor Phase

Let us first consider a 20 fs FWHM long pulse undergoing dispersion in air or a transparent material leading to a Group Delay Dispersion (GDD) of 500 fs^2 . The GDD will lead to a longer pulse duration with a linear arrival time delay of the frequency components within the pulse envelope, hence a well known chirped pulse. The *Simulator* module represents both the temporal and spectral intensities (see figure 5.1 upper panels). One clearly see a longer duration of the Gaussian shape up

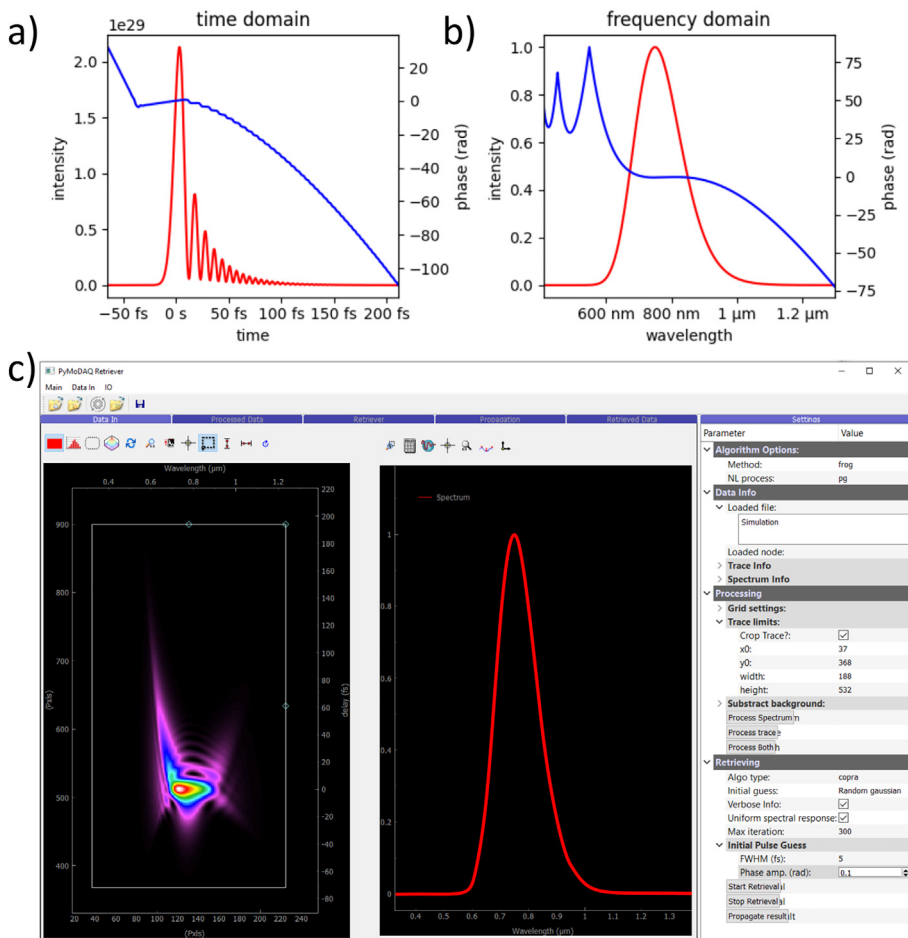


FIG. 5.4 – Ultrashort pulse (5 fs FWHM) shaped with 50 fs^2 GDD and 500 fs^3 TOD plotted in (a) the time domain, (b) the spectral domain. (c) Screenshot of the retriever module where the time–frequency representation of the above pulse using PG-FROG has been loaded. The loaded fundamental spectrum is displayed in the middle. The *Retriever*’s settings are visible on the right.

to about 80 fs. The choice of the characterization technique and its non-linearity can be made in the *Settings* panel, *Algorithm Options* section. Here, we choose the polarisation gating FROG (pg-frog) representation (bottom panel), which uses a third-order non-linear process. It clearly displays the linear arrival time delay of the spectral components. Depending on the technique, some other parameters have to be set (for instance with the d-scan, a dispersive material has to be chosen and its minimum and maximum insertion set to create the dispersion scan). The chosen non-linearity can have a great impact on the readability of the time–frequency trace and its interpretation. For instance, if using second harmonic generation FROG (shg-frog), as shown in figure 5.2, the temporal axis is symmetric and no clear information can be directly interpreted without running a retrieval algorithm¹⁰. It also means that the direction of time obtained from the algorithm is random in that case. However, for other details about the pulse shape, SHG-FROG is a good choice as it is easier to implement in the laboratory.

5.2.2.2 Self Phase Modulation

Let us now consider self-phase modulation (SPM), which is an often encountered pulse shaping effect due to a third-order non-linear process, and induces the creation of new spectral components. It is mathematically formulated by adding a temporal Gaussian phase to the pulse:

$$E^{\text{SPM}}(t) = E(t)e^{-\alpha \frac{t^2}{\delta t^2}} \quad (5.1)$$

The outcome of the shaping is represented in figure 5.3. The Gaussian phase is clearly visible in blue and the pulse duration is not modified. Compared to the pulse in figure 5.1, the spectral intensity is here broader and highly modulated. The consequence is that a shorter Fourier-limited pulse is produced. The real pulse duration can therefore be decreased if the remaining spectral phase is properly compensated for. Physically, SPM can be produced by the interaction of the pulse with a rare gas leading to chirp on top of the SPM effect. To get close to the Fourier-limited duration, only the GDD should be compensated for, for instance, using reflections on chirped mirrors¹¹.

5.2.2.3 Other Shaping

At the moment, only Taylor spectral phase and Gaussian temporal phase can be simulated. Sequence of pulses with identical shape can also be produced. If the need arises, please contact the authors to add some other shaping types or dive into python and propose some modification of the source code on GitHub¹²!

Using the *Simulator* module it is therefore easy to explore various pulse shaping and experimentally achievable¹³ time–frequency representations in order to be familiar with what to expect in an experiment. The chosen non-linearity has an impact on the clarity of the raw trace, but has no real impact on the retrieval details of the tested pulse (except the direction of time if using SHG-FROG, which can be of importance in some circumstances).

5.2.3 Retrieving Traces

Once the time–frequency trace has been obtained (either using the *Simulator* or an experimental one), pulse characterization can be performed using the *Retriever* module. It allows some pre-processing of the trace (cropping, re-sampling, background subtraction) and after the correct settings (mostly the experimental technique used to produce the trace, for instance PG-FROG in the top right of figure 5.4c) have been entered, the reconstruction algorithm can be chosen and run for a given amount of iteration. Figure 5.4c displays the data loaded into the module. Both a PG-FROG trace and the fundamental spectrum have been loaded corresponding to 5 fs FWHM shaped with 50 fs^2 GDD and 500 fs^3 TOD (Third-order dispersion term of the Taylor expansion). Section 5.3.3 fully explains how to use pre-processing on real experimental data where it becomes mandatory for the cleanest possible algorithm outcome.

While it is recommended to use the COPRA algorithm (as it has been fully tested), implementation of other algorithms has been done. The user can therefore compare various outcomes of the reconstruction mostly for education purposes.

Figure 5.5 shows the ongoing reconstruction at the 4th iteration of the COPRA algorithm. The left part displays the current retrieved temporal intensity while the right part displays information on the chosen algorithm and the current error value. The temporal shape is, at that iteration, still far from the expected one from figure 5.4a. However, after a few tens of iterations, both the temporal and spectral shapes (intensity and phase) are very close to the original one.

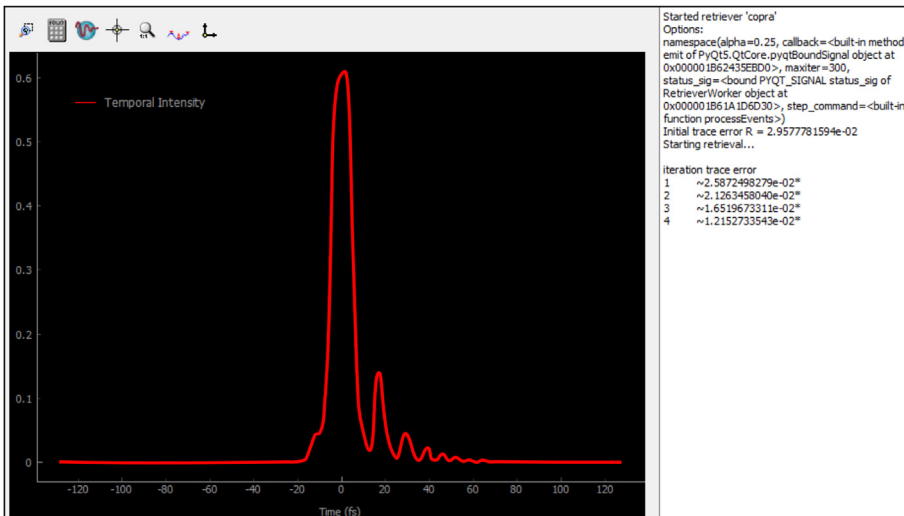


FIG. 5.5 – Live temporal intensity retrieved at the 4th iteration of the COPRA algorithm.

Figure 5.6 displays the final result from the algorithm as directly saved from the *Retriever* module. From there, the pulse is entirely characterized and numerical values can be saved from the module for further processing. Considerations on the initial phase/amplitude guess and the convergence criteria are discussed in the following section.

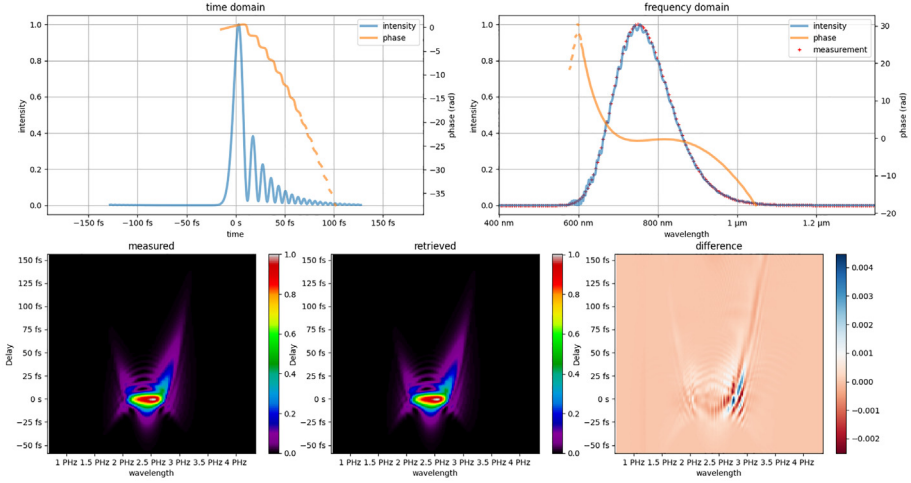


FIG. 5.6 – Retrieved pulse after the retrieval has been stopped at the 30th iteration, using the input trace and pulse shown in figure 5.4a. From top left to bottom right are, respectively, displayed (i) the temporal intensity and phase, (ii) the spectral intensity, phase and measured intensity, (iii) the normalized measured time–frequency trace, (iv) the normalized retrieved trace and (v) the difference between these normalized measured and retrieved traces.

5.3 Implementation in the Laboratory and Retrieval of Experimental Data

5.3.1 Experimental Pulse Retrieval: The Example of Post-Compression in a Hollow-Core Fiber

We now present the use of PyMoDAQ-Femto to characterize real pulses in a real experiment. As an illustration, we consider the problem of post-compressing femtosecond laser pulses to the few-cycle regime using a hollow-core fiber.¹⁴ As explained above, this technique combines self-phase modulation in a waveguide and subsequent negative second-order dispersion to shorten a femtosecond pulse. A significant difficulty in optimizing such a setup lies in obtaining the maximal spectral

broadening while avoiding high-order spectral phase components, which are very challenging to compensate. Therefore, a fast and robust pulse characterization interface is essential when setting up the device.

Several resources can be found on the physics and implementation of hollow-core fiber compressor; for instance, the review by T. Nagy *et al.*¹⁵ or the thesis of F. Böhle.¹⁶ Here, we briefly describe the components of such a setup. We used a 25 fs (close to its Fourier limited duration), 2 W, 1 kHz laser system, which we aimed to post-compress down to less than 5 fs. In the setup, sketched in figure 5.7, the beam is focused into a 1.5 m long stretched hollow cylindrical fiber (few-cycle Inc.) with 400 μm diameter and filled with neon. A telescope allows fine tuning of the beam size to obtain optimal incoupling with the fiber eigenmodes. The beam angle and position are actively stabilized using the leak of the last mirror, and we use circularly polarized light to reduce self-focusing and ionization.¹⁷ The output beam is then converted back to linear polarization, collimated, and the pulse is compressed using double angle chirped mirrors (PC1332, Ultrafast Innovations) which compensate the group delay dispersion. The output beam is then converted back to linear polarization, collimated, and the pulse is compressed using double angle chirped mirrors (PC1332, Ultrafast Innovations) which compensate the group delay dispersion.

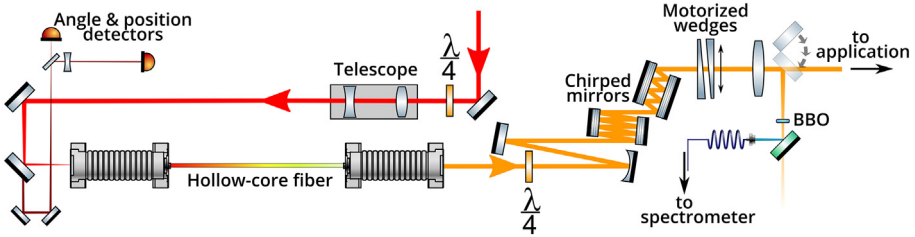


FIG. 5.7 – A typical post-compression setup using a hollow-core fiber, a chirped mirror compressor, and a simple dispersion scan setup.

Here, we choose to implement a dispersion-scan (d-scan) characterization,⁹ which consists in measuring the second harmonic spectrum as a function of dispersion added to the pulse. This is typically done by translating a wedge of a known material. The method is particularly simple to formulate as a Parameterized Non-linear Process Spectrum (PNPS), and thus readily compatible with the COPRA algorithm.¹ It also has the advantage of only necessitating a motorized set of wedges. The same wedges can be used to finely optimize the pulse compression in the experiment. As shown in figure 5.7, a fraction of the short pulses is focused into a 5 μm -thick BBO crystal (which is thin enough to phase-match the entire bandwidth of a short pulse¹⁸), and the second harmonic generation (SHG) is separated from the fundamental using a dichroic mirror.

5.3.2 Interfacing a Dispersion Scan Measurement in PyMoDAQ

The d-scan module must work with two hardware elements: a motorized translation for the moving glass wedge (in our case from Smaract), and a spectrometer covering the spectrum of interest (StellarNet Inc). The interfacing and data acquisition are performed with PyMoDAQ, as shown in figure 5.8: we use one adequate PyMoDAQ plugin to communicate with each instrument. In our case, both plugins had already been written¹⁹ and were freely available in the PyMoDAQ plugin database²⁰. The plugins are wrappers of manufacturer functions, which are typically given as dll/py/pyd/... files upon purchase. The plugin translates all hardware functionalities onto a set of mandatory functions common to all PyMoDAQ instruments, so that their operation is transparent regardless of the actual hardware. Here, the wedge linear translation is cast as a DAQ_Move object, and the spectrometer as a DAQ_Vviewer generating 1D data.

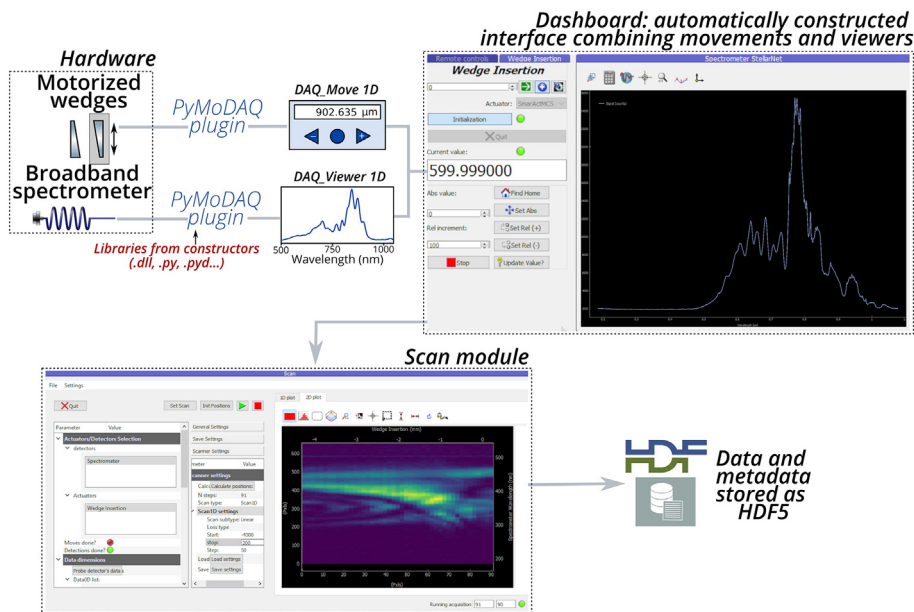


FIG. 5.8 – PyMoDAQ Workflow to construct an interface and perform a dispersion scan. The hardware is connected using PyMoDAQ plugins and combined in a dashboard. Then, the scan module performs the measurement and stores the results and metadata as a HDF5 file. For a demonstration of PyMoDAQ features, see the videos at <https://youtu.be/ZdYpQIZHMCY> or https://youtu.be/PWuZggs_HwM.

Once the plugins are installed, all PyMoDAQ functionalities can be used directly. When the plugins already exist (which is increasingly the case as more users do contribute to PyMoDAQ), the scanning program is thus constructed in a few

minutes and later loaded in less than a minute. As illustrated in figure 5.8, the DAQ_Move and 1D DAQ_Viewer are combined into a dashboard, providing an interface to optimize the acquisition settings. Then, the scan module is used to scan the wedge insertion while measuring the SHG spectrum. Figure 5.8 shows a screenshot of the live display of a d-scan measurement; this particular trace is not exceptionally nice (it is tilted and a lot of rebounds are present), directly indicating poor post-compression conditions. At the end, all data and metadata are stored into an HDF5 file, which is an open source, portable and self-describing file format well suited to storing heterogeneous and potentially large datasets.

5.3.3 Retrieving the Temporal Profile of Few-Cycle Pulses Using PyMoDAQ-Femto

From this point, the PyMoDAQ-Femto retriever module can be directly loaded from the scan program, and will automatically load the last measured trace. Otherwise, it can also be launched separately and load any previously measured experimental file, in a manner similar to that of synthetic traces (see section 5.2.3). The module requires two inputs: a measured PNPS trace, and the measurement of the fundamental spectrum of the characterized pulses. This spectrum can serve as a starting point for the algorithm, as a comparison to gauge the quality of the retrieval, or even be used to constrain the solution to have the same spectrum. In practice, its measurement can be done sequentially (by moving the spectrometer or by switching the beam path arrangement), or in parallel using two spectrometers.

5.3.3.1 Pre-Processing

Upon loading the trace and fundamental spectrum, much information is extracted from the data: the central wavelengths, the spectral widths, as well as the size of each axis. An option allows to rescale either axis to standard units in case the instruments used different ones.

The next important step is the pre-processing of the PNPS trace. Because of the variety of measurement types that can be retrieved in PyMoDAQ-Femto, as well as the various kinds of hardwares that can be used, it seemed unwise to try to automatize this process in a way that would work for all users. Instead, our philosophy has been to provide as much freedom to the users as possible for all parameters. This is perhaps off-putting at first, especially compared to more specialized and integrated commercial softwares, but the interface could easily be simplified by each user to fit their specific needs, after finding the optimal parameters. The first parameter to set is the type of measurement that was performed. All measurements supported by COPRA are available and summarized in table 5.1.

Three aspects must be then considered:

- (1) Any non-zero background in the measured trace or the fundamental spectrum must be carefully subtracted. Otherwise, a static error between retrieved and measured data will persist and prevent convergence. As shown in figure 5.10a, the user can graphically select a spectral region corresponding to the background level, which will be subtracted to the trace or spectrum.

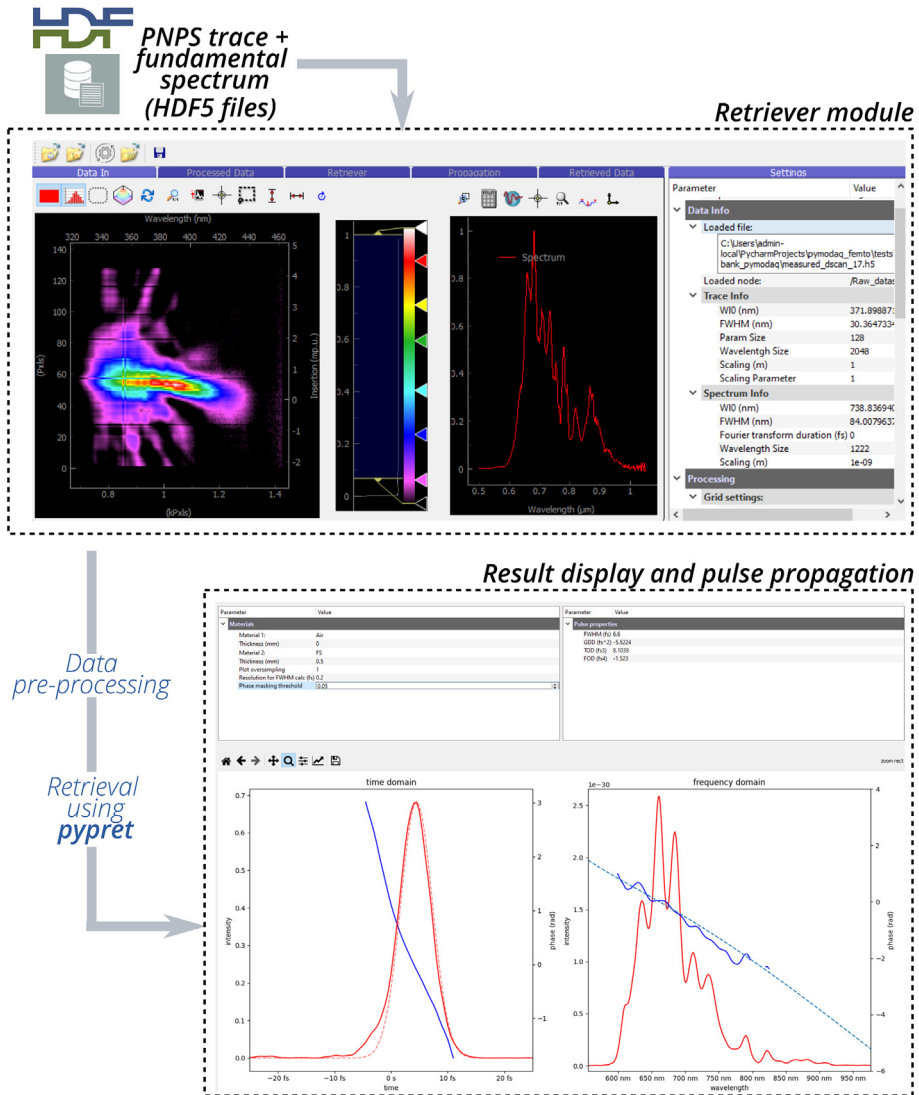


FIG. 5.9 – Structure and workflow of pulse analysis using PyMoDAQ-Femto.

- (2) The input trace must be cropped over a well-chosen region, both as a function of wavelength and measurement parameter (insertion for d-scan). This is particularly important here, as hollow-core fiber compressors can yield octave-spanning spectra; for instance, at a fundamental wavelength of 780 nm, it is possible to obtain spectra covering 400 nm to 1050 nm. This will cause

TAB. 5.1 – Available measurement types, with their full names, and supported non-linear processes: shg (Second Harmonic Generation), thg (Third Harmonic Generation), sd (Self Diffraction), pg (Polarization Gating), tg (Transient Grating).

Method	Full name	Supported non-linear processes
frog	Frequency-resolved optical gating	shg, pg, tg
d-scan	Dispersion scan	shg, thg, sd
ifrog	Interferometric frequency-resolved optical gating	shg, thg, sd
miips	Multiphoton intrapulse interference phase scan	shg, thg, sd
tdp	Time-domain ptychography	shg, thg, sd

some parts of the fundamental to spectrally overlap with the second harmonic radiation, perturbing the retrieval. Thus the retrieval must be restricted to regions without the fundamental. This only marginally affects the retrieval, as the robustness of d-scan allows the retrieval of the pulse over a wider spectral range²¹ than the chosen one. An option allows the user to graphically select the retrieval region.

- (3) Finally, a proper grid in the spectral dimension must be chosen. We follow the convention of *pypret* (which is used for the retrieval) and use a centered frequency axis. It is defined by the user with the most physically meaningful quantities: a center wavelength, a number of points for the grid (in the time and energy domain), and the spacing between two grid points in the time domain (in fs) (default settings are calculated from the loaded traces). The spacing corresponds to the time resolution of the retrieval. After setting these options, the trace and fundamental spectrum are pre-processed and interpolated on the new grid, and displayed in the “Processed Data” panel. The Fourier transform limited duration corresponding to the measured spectrum is also calculated. The user must ensure from these plots that the options are adequate (that is enough points to properly sample the trace features, but not too many to not lengthen the retrieval time); for instance, figure 5.10b and c shows two poor grid choices which are under-sampled in time and frequency domains, respectively. These options, in particular the points in each axis, have an impact on the calculation time needed for convergence – from seconds to a few minutes in some cases.

5.3.3.2 Retrieval

The pre-processing step converts the measured data onto instances of *pypret* classes: a MeshData object (the trace), a Pulse object (the fundamental spectrum), and a PNPS object. Again, we purposely chose to give the user as many options for the retrieval as possible:

Algorithm type: The *pypret* module implements several reconstruction algorithms (COPRA, PCGPA...), which allows to compare their efficiency in various situations.

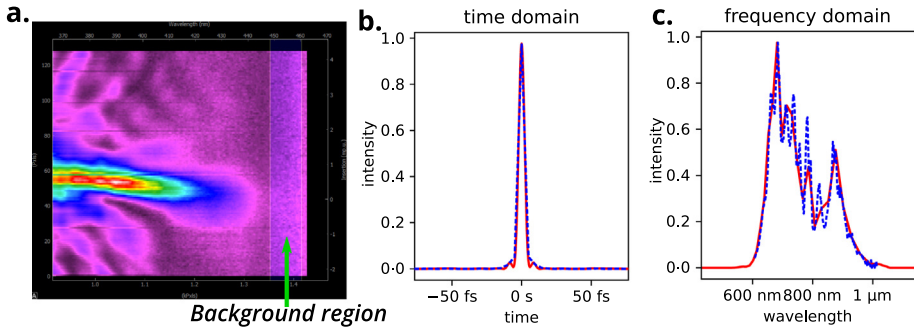


FIG. 5.10 – Pre-processing of data prior to retrieving. (a) Graphical tools allow the user to select several regions: here, the background region, whose average signal will be subtracted to the trace. (b) The time profile of the Fourier transform limited pulse after interpolation on a chosen grid; (c) the spectral intensity interpolated on the grid. The two curves show two poor choices of an interpolation grid: the full-red interpolation is good in time but poor in frequency while the dashed blue one is under-sampled in time but good in frequency.

However, we have only used and tested the COPRA algorithm, which readily works on all PNPS methods. In the future, other algorithms could be tested and debugged, and the interface could be modified to display only algorithms compatible with the chosen PNPS.

Verbose information: Displays the error between retrieved and measured traces at each iteration, together with the spectral and temporal intensity of the current solution.

Maximum iteration: In our implementation, we simply let the algorithm run for a given number of iterations. Other approaches, such as specifying a convergence criterion in terms of trace error, could also be implemented in the future.

Uniform spectral response: This parameter determines whether we assume the entire apparatus to have a flat spectral response, or an energy-dependent one. In a real experiment, the measured SHG spectrum will be convoluted by several factors which in our case are: the reflectivity of the dichroic mirror, the coupling and the propagation into the fiber, the spectral response of the spectrometer, and the phase matching curve of the BBO. If we assume a flat spectral response, the algorithm, which computes the ideal SHG from a given pulse, will converge towards a flawed solution. Instead of trying to calibrate the absolute response of the entire apparatus, the overall spectral response can be factored into an energy-dependent weighting function in the algorithm.⁹

Initial guess: We experimented with two approaches for the starting point of the algorithm. The initial guess can either be taken as the transform limited pulse obtained from the fundamental spectrum, which can favor convergence towards a

similar spectrum. This can however be seen as a bias to gauge the algorithm performances. Alternately, the algorithm can use a gaussian pulse with a random phase as a starting point, with user-defined duration and phase noise amplitude. In some cases, this choice actually gave faster convergence and lower residual error.

Fixed spectral intensity: Finally, this last option depends entirely on the user's strategy. When retrieving a pulse, it can be difficult to gauge the validity of the retrieval. It is customary to compare the retrieved spectral intensity with the measured one – this way, one can assess if the solution is realistic. Another approach is to constrain the algorithm to match the measured spectral intensity.²² If this option is ticked, the intensity of the guessed pulse at each iteration is replaced by the measured one. In other words, only the spectral phase is allowed to evolve during the retrieval. Since less variables are let free to evolve, this will typically result in a slower convergence and larger residual error, but with the benefit of a possibly more realistic solution. Below we show the retrieval of the same trace, with and without this option.

5.3.4 Examples of Retrieved Pulses

Figure 5.11 displays the retrieved trace obtained with relatively little gas in the hollow-core fiber. The solution is obtained after 200 iterations using a 4096 points spectral grid, a random gaussian pulse as a starting point and a non-uniform spectral response. In figure 5.11b, both the spectral amplitude and phase are let free to evolve. The retrieval error, defined as the normalized root mean square error between the measured and retrieved trace, is 1.45%. Note that the final error cannot be lower than the Gaussian noise present in the experiment (stemming from intensity fluctuations and measurement noise).¹ In addition, the retrieved spectrum matches fairly well with the measured one (figure 5.11d), suggesting a satisfactory pulse measurement retrieval. In figure 5.11c we also show the retrieved trace obtained while constraining the fundamental spectral intensity to the measured one. The retrieval quality is slightly worse (1.78% error), which is expected as explained before.

Finally, in the last step of the retriever module, we allow the user to propagate the retrieved pulse through a chosen amount of common optical materials (air, fused silica, KDP...) and to observe the resulting temporal shape. This allows a quick (but manual) feedback with the experiment: exactly how much material should be added to obtain the shortest pulse on-target? How can the measured spectral phase best be compensated? In figure 5.11e, inserting 0.50 mm of glass gives the optimal pulse duration (5.88 fs FWHM). The retrieval with free or fixed fundamental spectra gives very similar pulse profiles for this dataset. In this particular case, the pulses were sent into a vacuum chamber to produce high harmonic generation, and the entrance window was 0.5 mm thick – the pulses were therefore optimal for the application.

In figure 5.12 we display a measured and retrieved trace for even shorter pulses (obtained with higher pressure in the hollow core fiber). The trace is visibly thinner

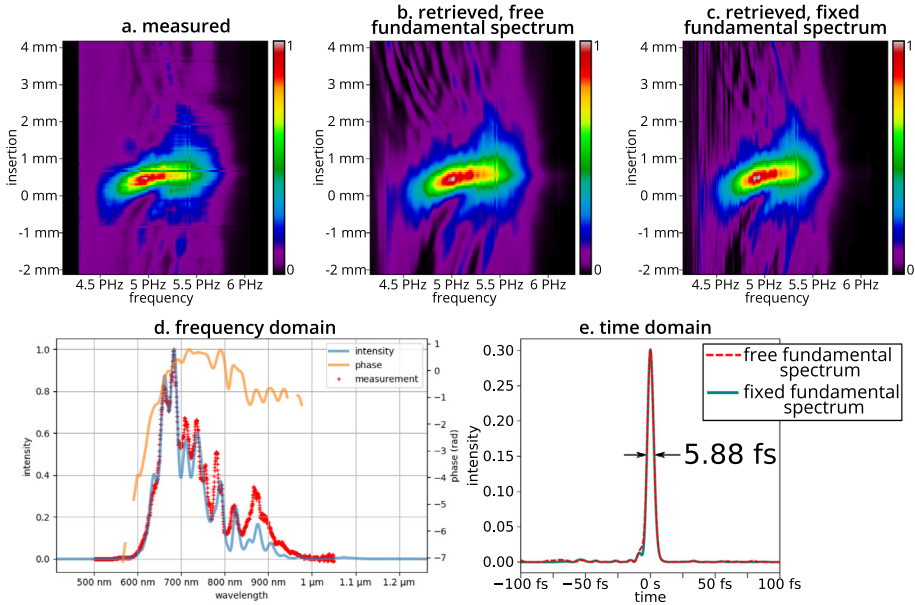


FIG. 5.11 – Retrieval of a dispersion scan trace. (a) Measured trace. Retrieved traces using either an unconstrained fundamental spectrum (b) or a spectrum fixed to the experimentally measured one (c). (d) Retrieved spectral intensity and phase, together with the measured fundamental spectrum (red crosses); (e) Propagated pulse after 0.5 mm of fused silica, obtained using free (dashed red) or fixed (filled blue) fundamental spectra.

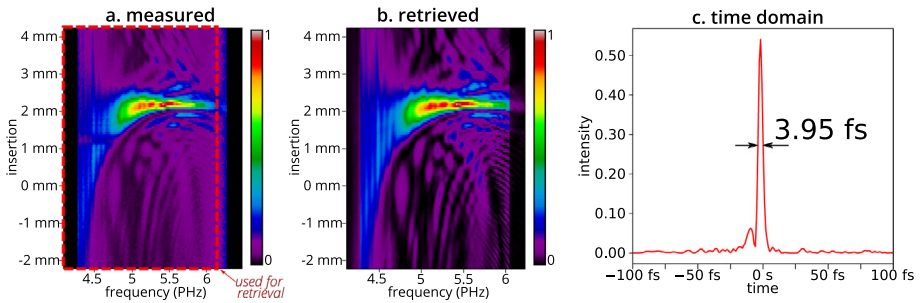


FIG. 5.12 – Retrieval of a dispersion scan trace with a sub-4 fs pulse. (a) Measured and (b) Retrieved traces; (c) Propagated pulse after 2.23 mm of fused silica.

in the insertion dimension, and shows more complex structures, which do arise due to higher order phase terms. Yet, the COPRA retriever manages to reconstruct the trace with a 1.59% error. Propagating the pulse through 2.23 mm of fused silica yields a 3.95 fs-long pulse. Note that less glass was present in the beam path compared to the previous measurement, explaining the large difference in optimal insertion.

5.4 Conclusion

We have presented the capabilities of the PyMoDAQ-Femto module, which allows to simulate and retrieve PNPS traces. The possibility of quickly visualizing the shape of various traces is particularly helpful to understand measurement techniques for short femtosecond pulses. Besides its educational value, this open-source software can also be used in real conditions with experimental data, as we demonstrate in a hollow-core fiber post-compression experiment. When set up in combination with PyMoDAQ, it allows to measure femtosecond pulse reliably and transparently to the user, regardless of the used hardware. We hope that the large flexibility in pre-processing and retrieving options encourages its use in diverse situations, and stimulates collaborative contributions to the software. Code modification or issues can be made using the public PyMoDAQ-Femto GitHub repository.

References

- [1] Geib N.C., Zilk M., Pertsch T., Eilenberger F. (2019) Common pulse retrieval algorithm: A fast and universal method to retrieve ultrashort pulses, *Optica* **6**, 495.
- [2] Geib N. Pypret GitHub repository. <https://github.com/ncgeib/pypret>.
- [3] Weber S.J. (2021) PyMoDAQ: An open-source Python-based software for modular data acquisition, *Rev. Sci. Instrum.* **92**, 045104.
- [4] https://github.com/CEMES-CNRS/pymodaq_femto.
- [5] PyMoDAQ-Femto contains a utility python script to convert your data (in *numpy* format) to a valid *hdf5* PyMoDAQ file.
- [6] <https://pymodaq.cnrs.fr>.
- [7] <https://pymodaq-femto.readthedocs.io>.
- [8] Trebino R., Kane D.J. (1993) Using phase retrieval to measure the intensity and phase of ultrashort pulses: Frequency-resolved optical gating, *J. Opt. Soc. Am. A* **10**, 1101.
- [9] Miranda M., Fordell T., Arnold C., L'Huillier A., Crespo H. (2012) Simultaneous compression and characterization of ultrashort laser pulses using chirped mirrors and glass wedges, *Opt. Exp.* **20**, 688.
- [10] The clearest non-linear trace would be one mimicking the Gabor transform, but for this, one must shear the tested pulse by a real function. This is why some third order non-linearities give clearer shape. In PG-FROG the pulse field is first mixed with its conjugate, giving a real quantity that will shear the third field of this third order non-linearity.
- [11] A chirped mirror is a multi-layer dielectric mirror reflecting spectral components at different depth within the mirror effectively inducing GDD. It is tailored to produce negative GDD therefore compensating for the natural positive dispersion in gas or materials (See chapter 2, figure 9).

- [12] https://github.com/CEMES-CNRS/pymodaq_femto.
- [13] The Simulation module excludes more theoretical time-frequency representations such as Gabor or Wigner functions. Their peculiarities are however discussed in the previous chapter.
- [14] Nisoli M., De Silvestri S., Svelto O. (1996) Generation of high energy 10 fs pulses by a new pulse compression technique, *Appl. Phys. Lett.* **68**, 2793.
- [15] Nagy T., Simon P., Veisz L. (2021) High-energy few-cycle pulses: Post-compression techniques, *Adv. Phys.: X* **6**, 1845795.
- [16] Böhle F. (2017) *Near-single-cycle laser for driving relativistic plasma mirrors at kHz repetition rate - Development and application*. Ph.D. thesis, Université Paris Saclay (COMUE). <https://pastel.archives-ouvertes.fr/tel-01806334>.
- [17] Chen X. *et al.* (2009) Generation of 4.3 fs, 1 mJ laser pulses via compression of circularly polarized pulses in a gas-filled hollow-core fiber, *Opt. Lett.* **34**, 1588.
- [18] Baltuska A., Pshenichnikov M., Wiersma D. (1999) Second-harmonic generation frequency-resolved optical gating in the single-cycle regime, *IEEE J. Quantum Electron.* **35**, 459.
- [19] If your instrument is not available yet, please watch <https://youtu.be/9O6pqz89UT8> for an introduction on how to write your own plugin.
- [20] For the list of actually available plugins, see: https://github.com/CEMES-CNRS/pymodaq_plugin_manager.
- [21] Silva F. *et al.* (2014) Simultaneous compression, characterization and phase stabilization of GW-level 14 cycle VIS-NIR femtosecond pulses using a single dispersion-scan setup, *Opt. Exp.*
- [22] Miranda M. *et al.* (2017) Fast iterative retrieval algorithm for ultrashort pulse characterization using dispersion scans, *J. Opt. Soc. Am. B* **34**, 190.