

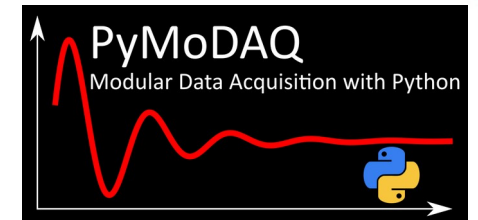


© Patri

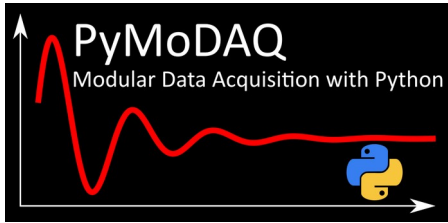
PyMoDAQ-Femto

Modular Data Acquisition with Python
For Femtosecond pulse characterization

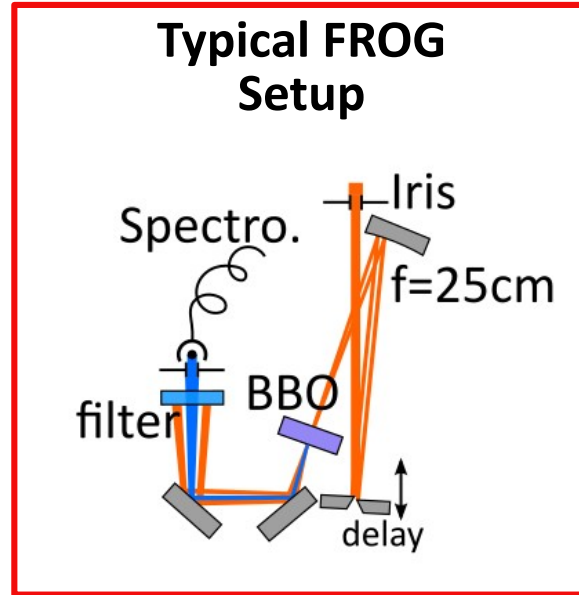
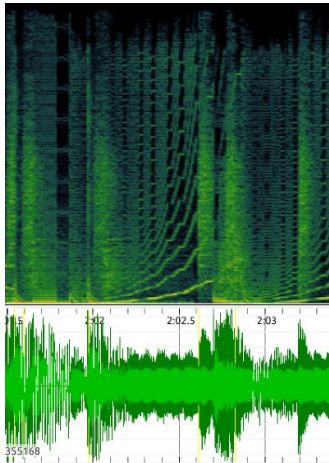
Sébastien Weber



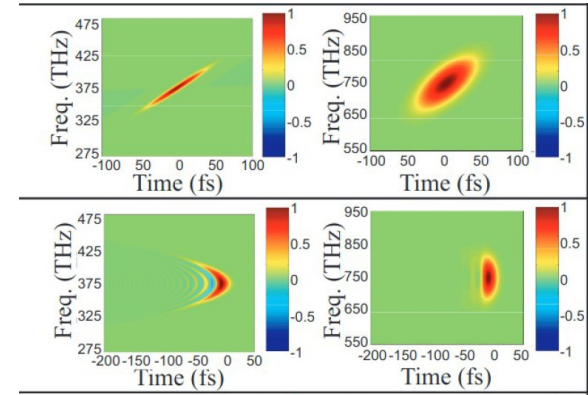
What for ?



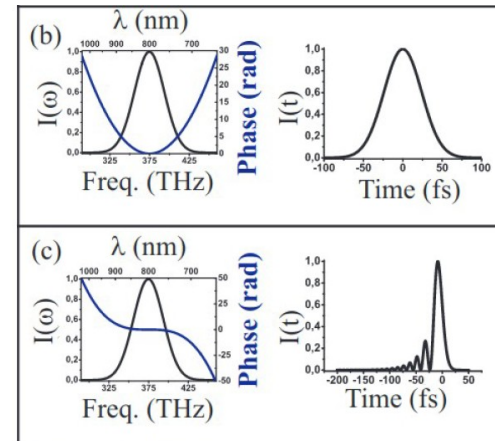
Acquisition of spectra as a function of the delay



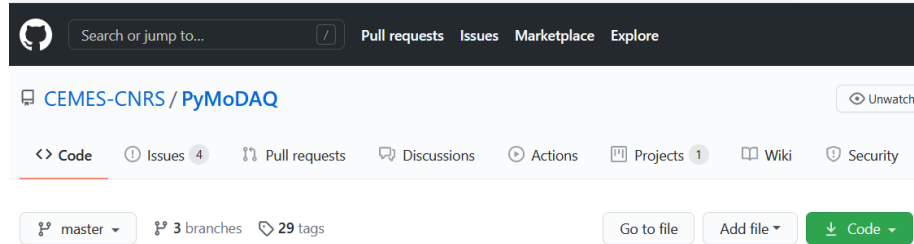
PyMoDAQ-Femto



Simulations and reconstructions

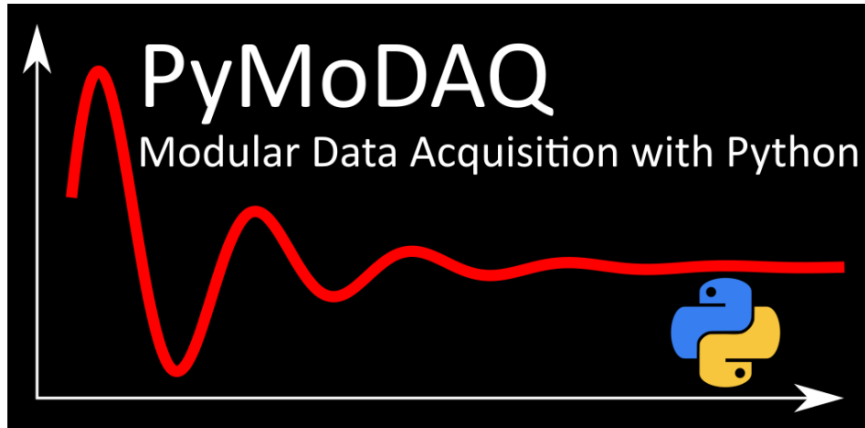


Based on open Source python codes !



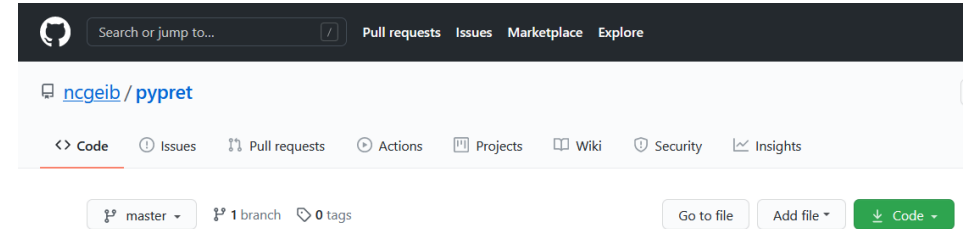
PyMoDAQ

pypi v3.0.4 docs passing codecov unknown Python package passing



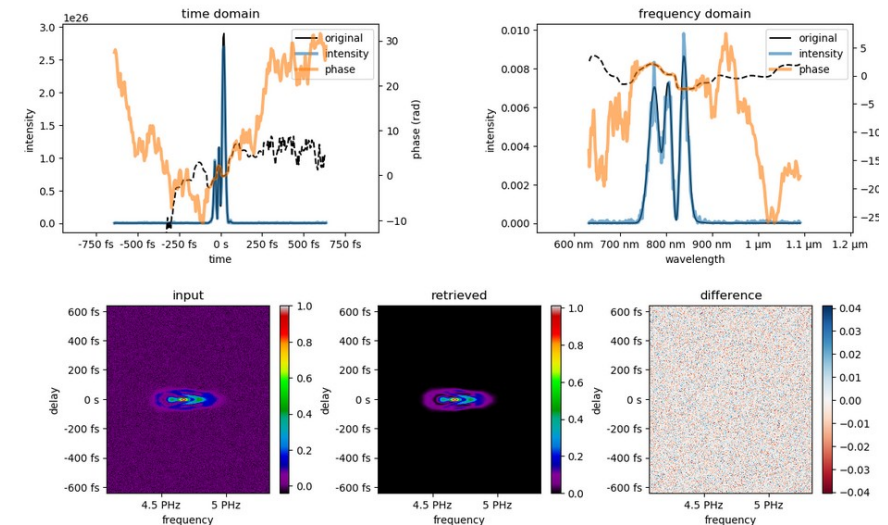
PyMoDAQ, Modular Data Acquisition with Python, is a set of **python** modules used to interface any kind of experiments. It simplifies the interaction with detector and actuator hardware to go straight to the data acquisition of interest.

<http://pymodaq.cnrs.fr>
PyMoDAQ Review of scientific Instrument (Submitted)



Python for Pulse Retrieval

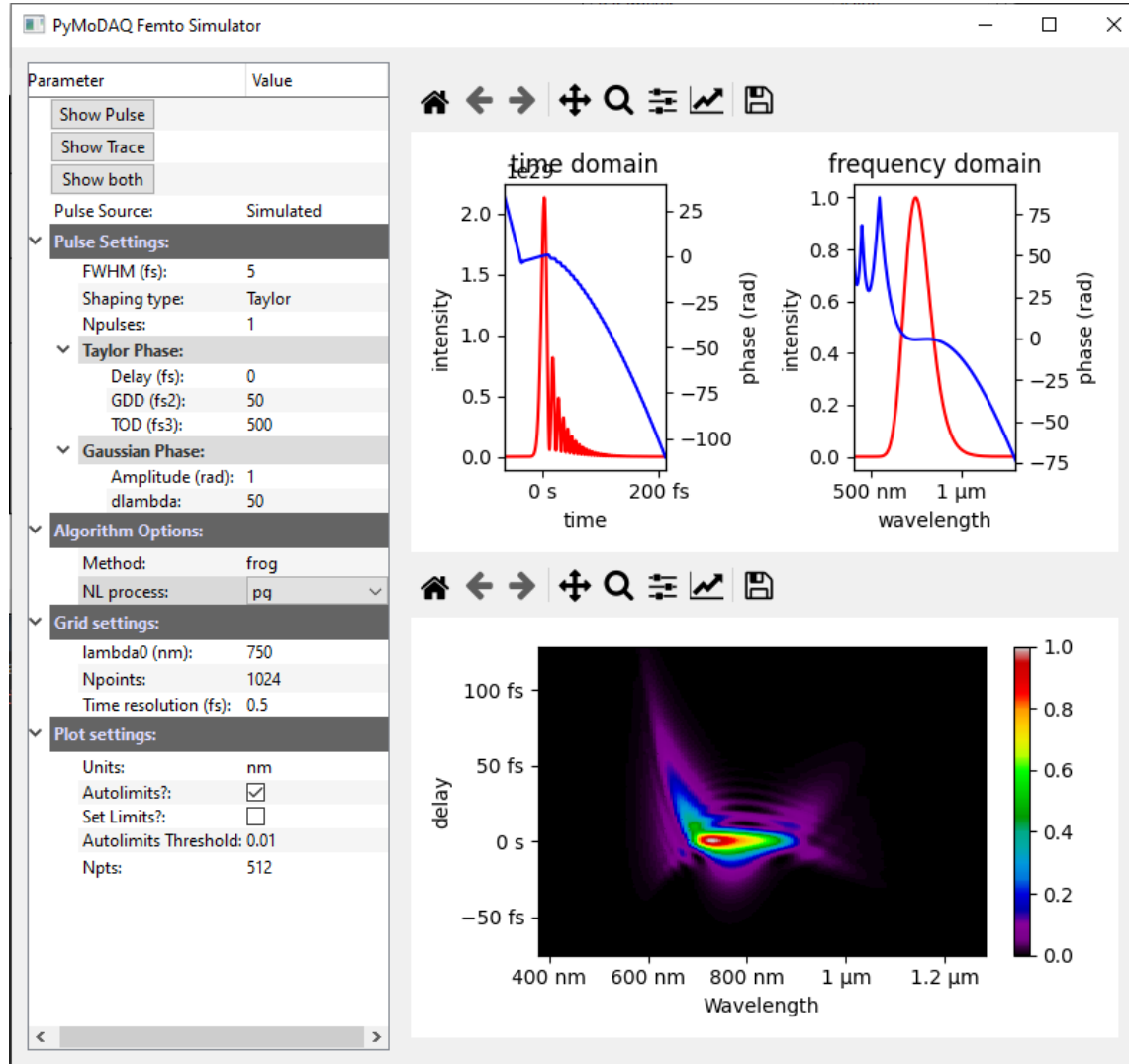
This project aims to provide numerical algorithms for ultrashort laser pulse measurement methods such as frequency resolved optical gating (FROG), dispersion scan (d-scan), or time-domain ptychography (TDP) and more. Specifically, provides a reference implementation of the algorithms presented in our paper "[Common pulse retrieval algorithm: a fast and universal method to retrieve ultrashort pulses](#)".



<https://pypret.readthedocs.io>
PyPret : Geib *et al.* Optica 6, 495-505 (2019)

Complete interface for :

1) Simulation





Complete interface for :

2) Acquisition

The image shows the PyMoDAQ Dashboard for 'femto'. It features a top menu bar with 'File', 'Settings', 'Preset Modes', 'Overshoot Modes', 'ROI Modes', 'Remote/Shortcuts Control', and 'Extensions'. Below the menu is a 'Loader' section with 'Remote controls' and a 'Parameter' table. The 'Delay' control module is active, showing a value of 10.5 and a 'Current value' of 10.500000. A 'Spectrometer Settings' panel is visible, including 'Main Settings', 'Detector Settings', and 'Pulse Settings'. A 'Spectrometer PNPS' plot shows an 'NL trace' of intensity versus 'Wavelength (nm)' from 0.5 to 1.1.

PyMoDAQ's Dashboard and its control modules

The image shows the DaqScan acquisition interface, a Python-based application. It includes a 'Scan' window with a 'Parameter' table and various settings panels like 'General Settings', 'Scanner Settings', and 'ScanID settings'. The 'ScanID settings' panel shows 'Scan subtype: Linear', 'Loss type', 'Start: -100', 'Stop: 200', and 'Step: 1'. A 2D plot displays a 'Delay (whatever)' heatmap with 'Spectrometer Wavelength (nm)' on the y-axis and 'Pxls' on the x-axis. A color scale on the right indicates intensity from 0 to 1.0.

PyMoDAQ's extension : DaqScan

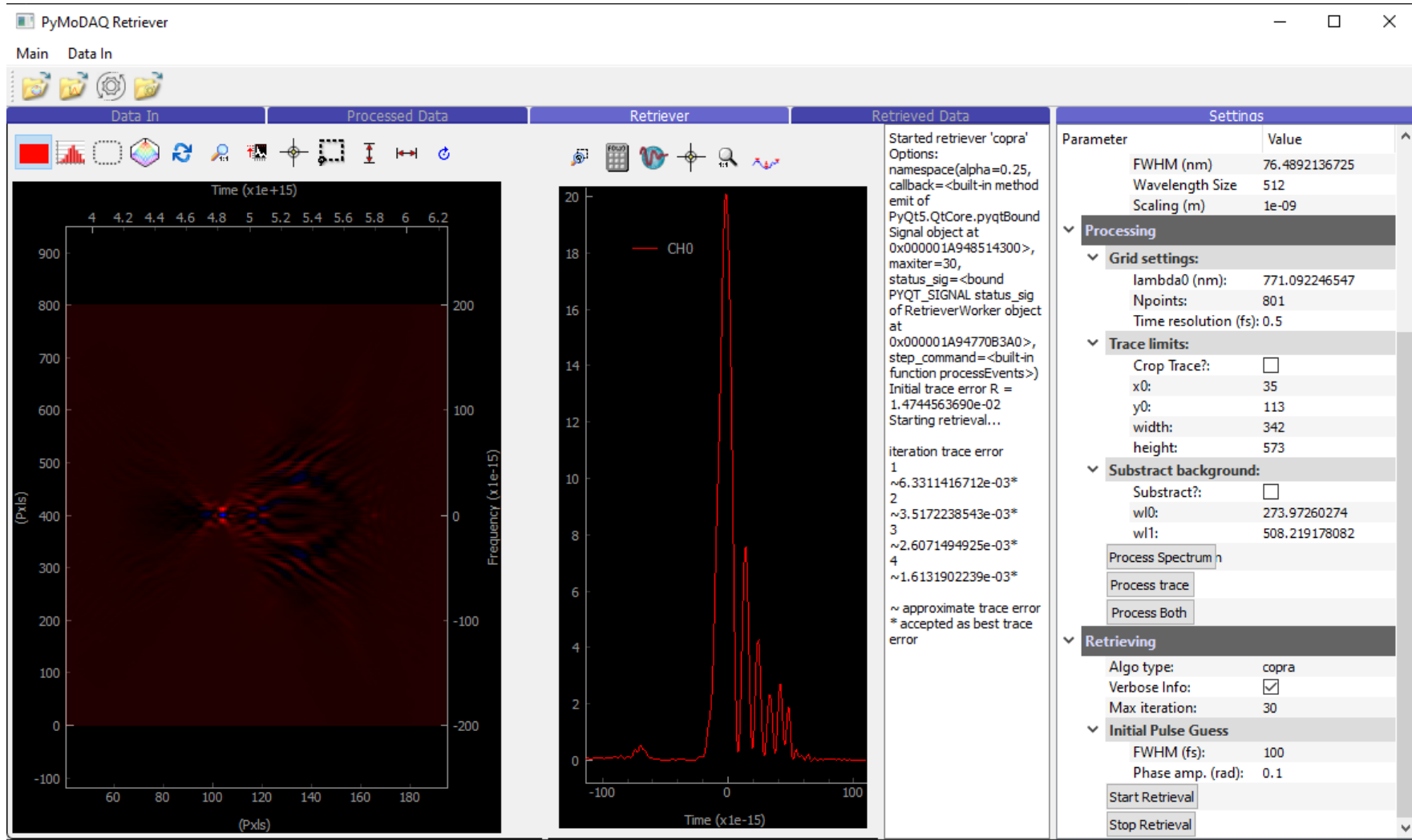
Complete interface for :

3) Pulse Shape Retrieval

PyMoDAQ Retriever

Main Data In

Data In Processed Data Retriever Retrieved Data Settings



The interface displays the following components:

- Data In:** A 2D spectrogram showing Frequency (x1e-15) vs Time (x1e+15). The x-axis ranges from 4 to 6.2, and the y-axis ranges from -100 to 900. A central pulse is visible at approximately Time = 5.2 x 10¹⁵ s.
- Processed Data:** A 1D pulse trace labeled 'CH0' showing Amplitude vs Time (x1e-15). The x-axis ranges from -100 to 100, and the y-axis ranges from 0 to 20. The trace shows a sharp peak at Time = 0.
- Retriever:** A text area displaying the status of the retrieval process, including the namespace, signal object, and iteration trace error.
- Retrieved Data:** A text area displaying the retrieved pulse trace error and its components.
- Settings:** A panel for configuring the retrieval process, including parameters like FWHM, Wavelength Size, and Scaling, as well as processing and retrieving options.

Retriever Status:

```

Started retriever 'copra'
Options:
namespace(alpha=0.25,
callback=<built-in method
emit of
PyQt5.QtCore.pyqtBound
Signal object at
0x000001A948514300>,
maxiter=30,
status_sig=<bound
PYQT_SIGNAL status_sig
of RetrieverWorker object
at
0x000001A94770B3A0>,
step_command=<built-in
function processEvents>)
Initial trace error R =
1.4744563690e-02
Starting retrieval...

iteration trace error
1
~6.3311416712e-03*
2
~3.5172238543e-03*
3
~2.6071494925e-03*
4
~1.6131902239e-03*

~ approximate trace error
* accepted as best trace
error
  
```

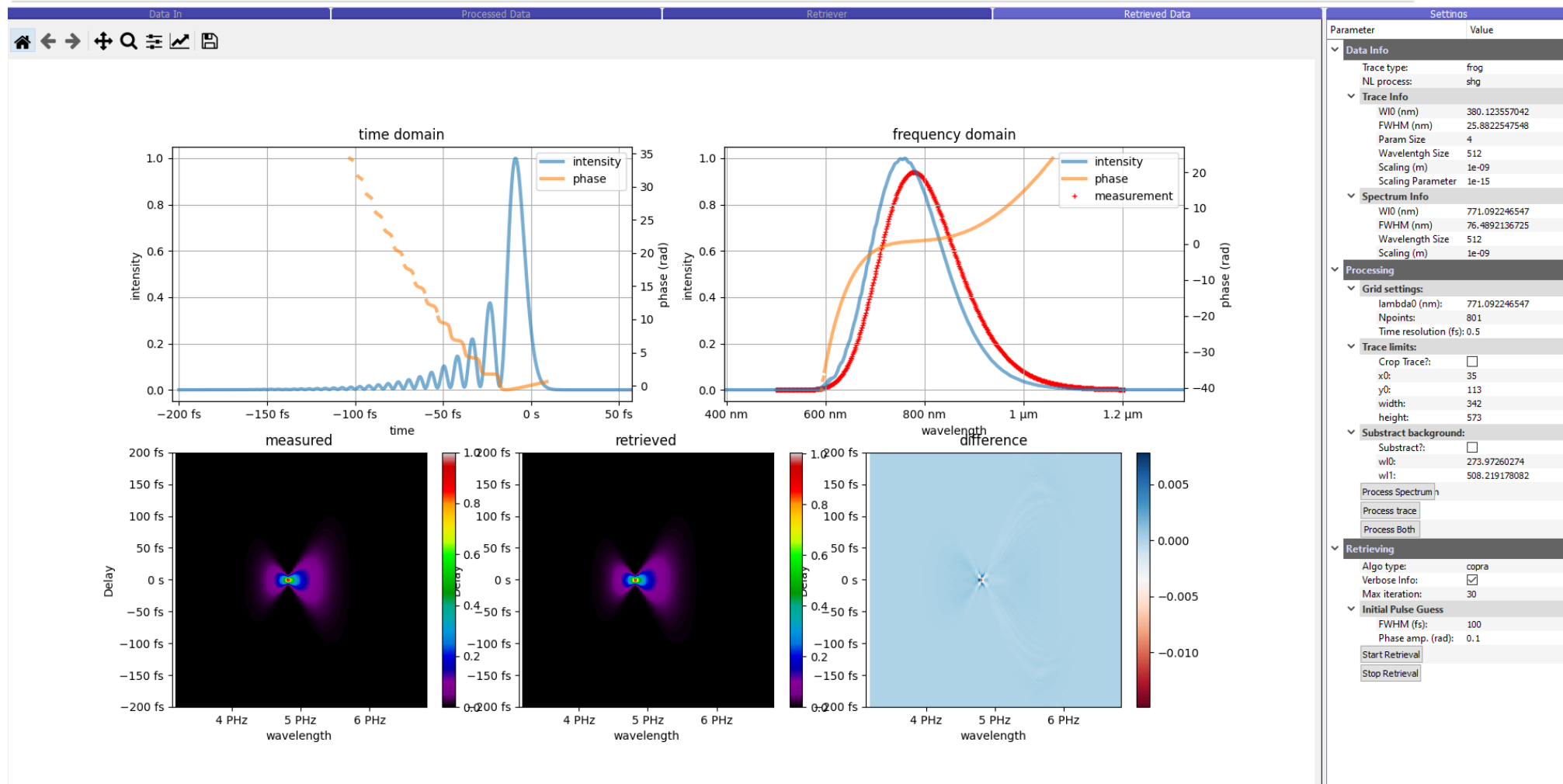
Settings:

Parameter	Value
FWHM (nm)	76.4892136725
Wavelength Size	512
Scaling (m)	1e-09
Processing	
Grid settings:	
lambda0 (nm):	771.092246547
Npoints:	801
Time resolution (fs):	0.5
Trace limits:	
Crop Trace?:	<input type="checkbox"/>
x0:	35
y0:	113
width:	342
height:	573
Subtract background:	
Subtract?:	<input type="checkbox"/>
wl0:	273.97260274
wl1:	508.219178082
Process Spectrum <input type="checkbox"/>	
Process trace <input type="checkbox"/>	
Process Both <input type="checkbox"/>	
Retrieving	
Algo type:	copra
Verbose Info:	<input checked="" type="checkbox"/>
Max iteration:	30
Initial Pulse Guess	
FWHM (fs):	100
Phase amp. (rad):	0.1
Start Retrieval <input type="button" value="Start Retrieval"/>	
Stop Retrieval <input type="button" value="Stop Retrieval"/>	



Complete interface:

4) With fine exportable graphs

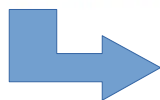


Who did what?



Nils C. Geib

Friedrich Schiller University Jena | FSU · Abbe Center of Photonics (ACP)



Developped the PyPret package for Non-linear Trace reconstruction



Sébastien J Weber

Centre d'Élaboration de Matériaux et d'Etudes Structurales
Research Engineer at CEMES-CNRS Toulouse

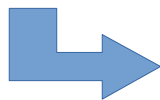


Developped PyMoDAQ and the interface on Pypret



Romain Généaux

Atomic Energy and Alternative Energies Commission | CEA · Laboratory Interactions, Dynamics and Lasers
PhD



Beta-testing and initial inpulse on PyMoDAQ-Femto

I want to measure data as a function of varying parameters !

Detectors

- Camera
- Analog signals
- Spectrum
- ...

Actuators

- Linear stage
- Rotation
- Temperature controller
- Current
- Voltage
- ...

Camera

XY stage

Laser wavelength

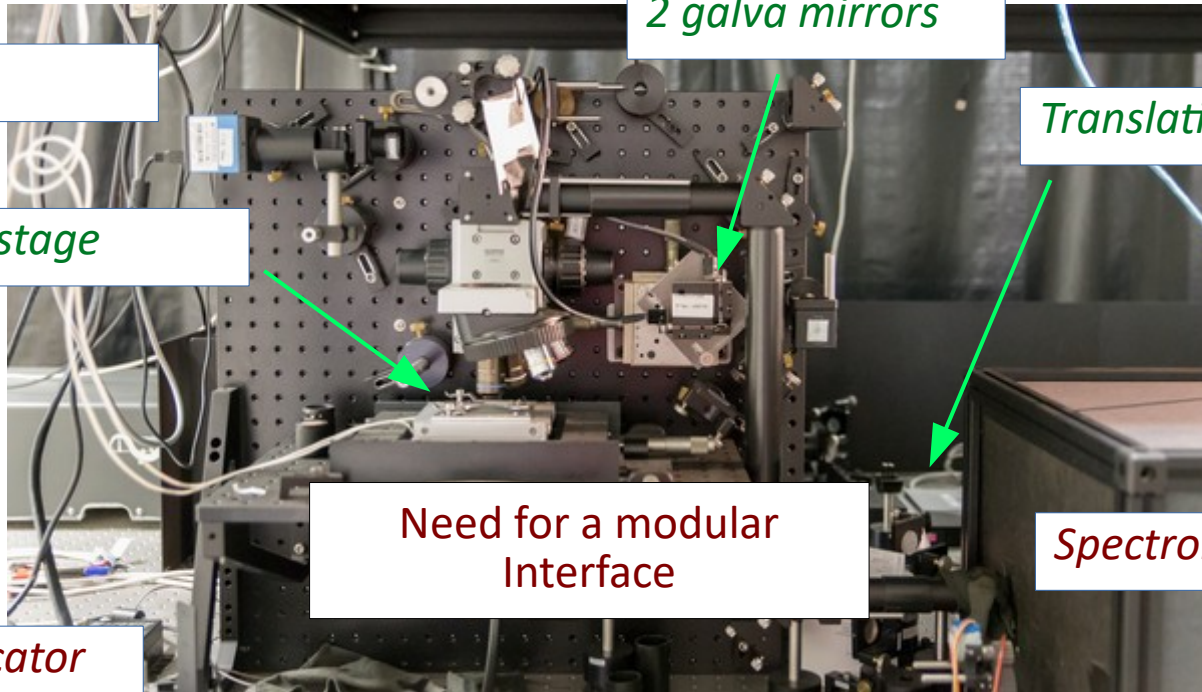
Photomultiplier

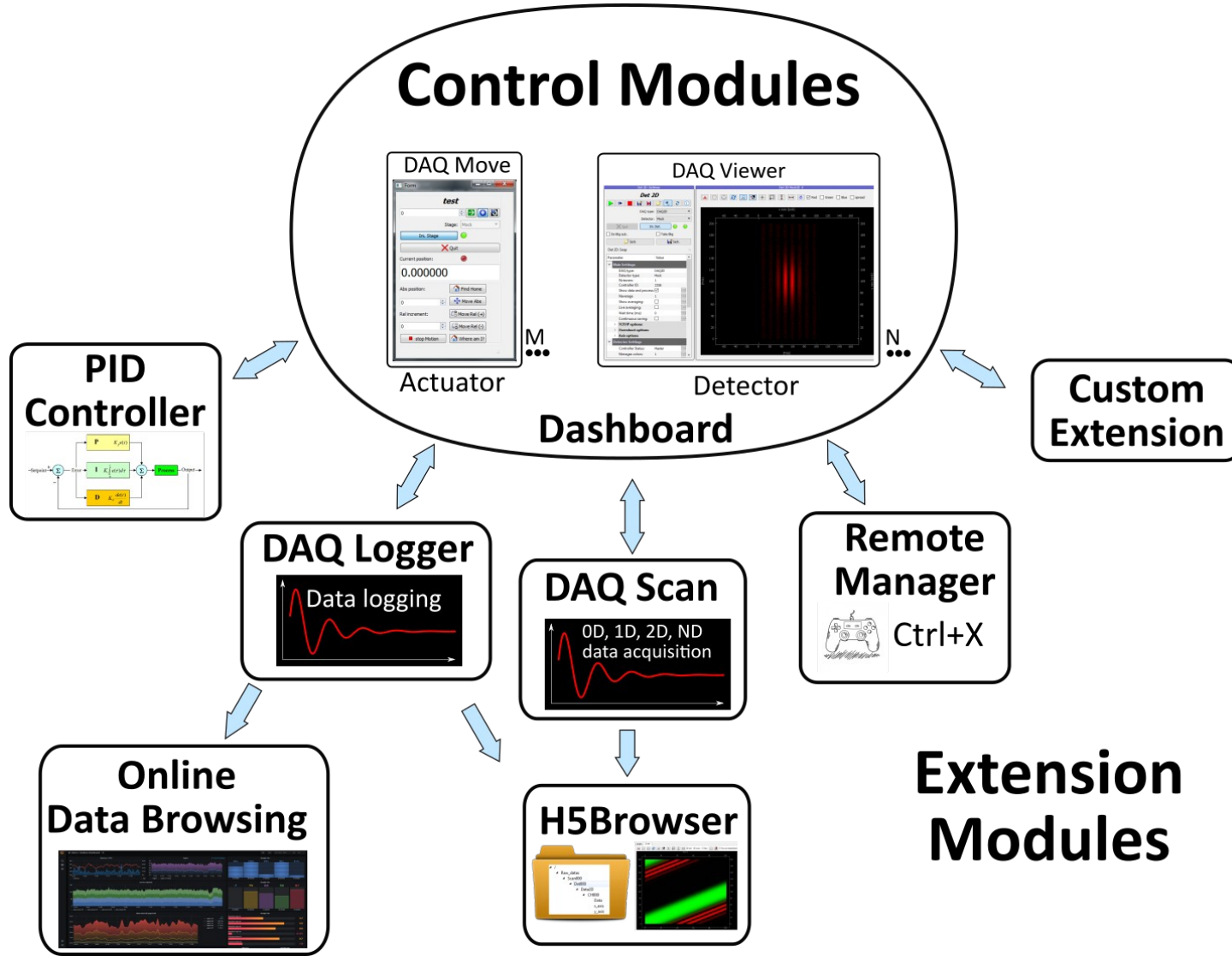
2 galva mirrors

Translation stage

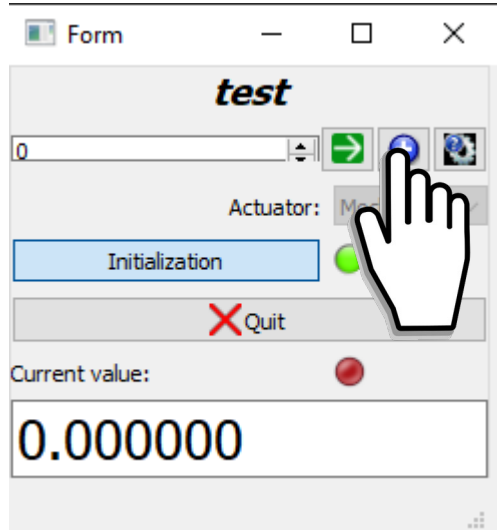
Need for a modular Interface

Spectrometer





DAQ Move : Actuators set/get values



Form

test

0

Actuator: Mock

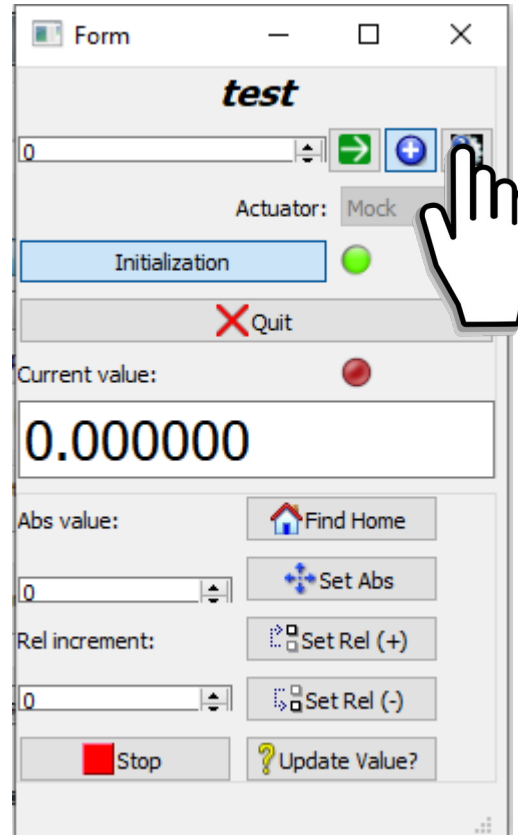
Initialization

Quit

Current value:

0.000000

A hand cursor is pointing to the green 'Initialization' button.



Form

test

0

Actuator: Mock

Initialization

Quit

Current value:

0.000000

Abs value: Find Home

Set Abs

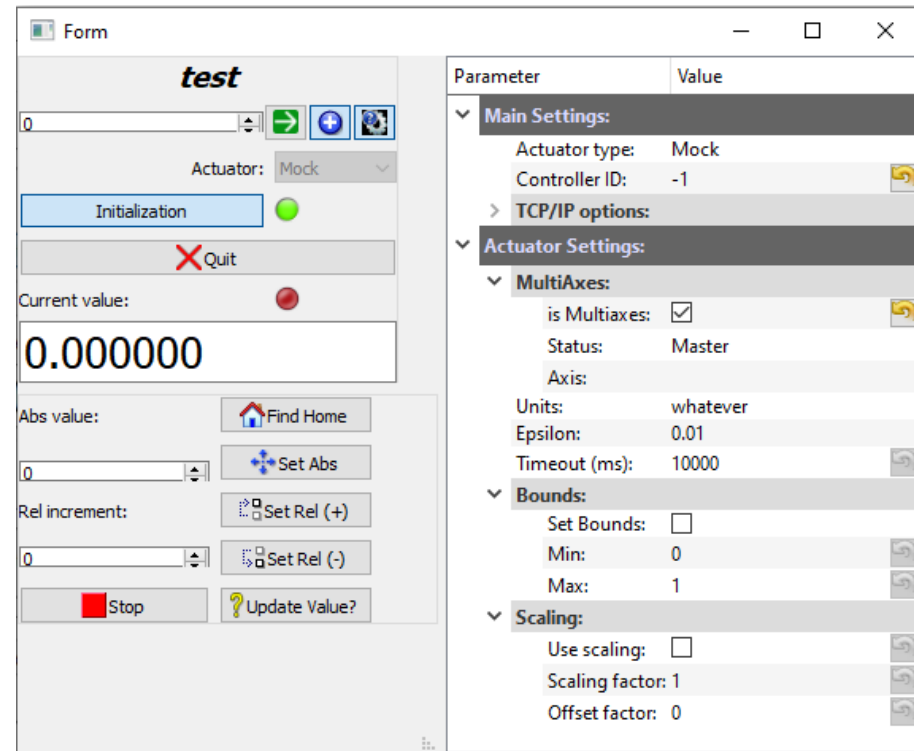
Rel increment: Set Rel (+)

Set Rel (-)

Stop

Update Value?

A hand cursor is pointing to the blue '+' button.



Form

test

0

Actuator: Mock

Initialization

Quit

Current value:

0.000000

Abs value: Find Home

Set Abs

Rel increment: Set Rel (+)

Set Rel (-)

Stop

Update Value?

Parameter	Value
Main Settings:	
Actuator type:	Mock
Controller ID:	-1
TCP/IP options:	
Actuator Settings:	
MultiAxes:	
is Multiaxes:	<input checked="" type="checkbox"/>
Status:	Master
Axis:	
Units:	whatever
Epsilon:	0.01
Timeout (ms):	10000
Bounds:	
Set Bounds:	<input type="checkbox"/>
Min:	0
Max:	1
Scaling:	
Use scaling:	<input type="checkbox"/>
Scaling factor:	1
Offset factor:	0

Dashboard example

PyMoDAQ Dashboard: preset_default

File Settings Preset Modes Overshoot Modes ROI Modes Actions ?

Parameter Value

- Loaded presets
 - Preset file preset_default.xml
 - Overshoot file
 - Layout file
 - ROI file preset_default.dock
- Actuators Init.
 - Xaxis
 - Yaxis
- Detectors Init.

2020/03/10 14:40:38: Preset mode (preset_default.xml) h

Yaxis

0

Stage: Mock

Ini. Stage

Quit

Current position: 0.000000

Det 2D Settings

Det 2D

Parameter Value

- Main Settings:
- Detector Settings
 - Controller Status: Master
 - Nimages colors: 1
 - Nimages panels: 1
 - Threshold: 1
 - Bool
 - rolling
 - Nx

Det 2D Mock2D_0

x axis (pxls)

y axis (pxls)

Det 1D Settings

Det 1D

Parameter Value

- Main Settings:
- Detector Settings
 - Controller Status: Master
 - rolling: 1
 - multi:
 - Mock1
 - Amp: 20
 - x0: 125
 - dx: 20
 - n: 1
 - amp_noise: 0.1
 - Mock2
 - Amp: 10
 - x0: 325

Det 0D Settings

Det 0D

Parameter Value

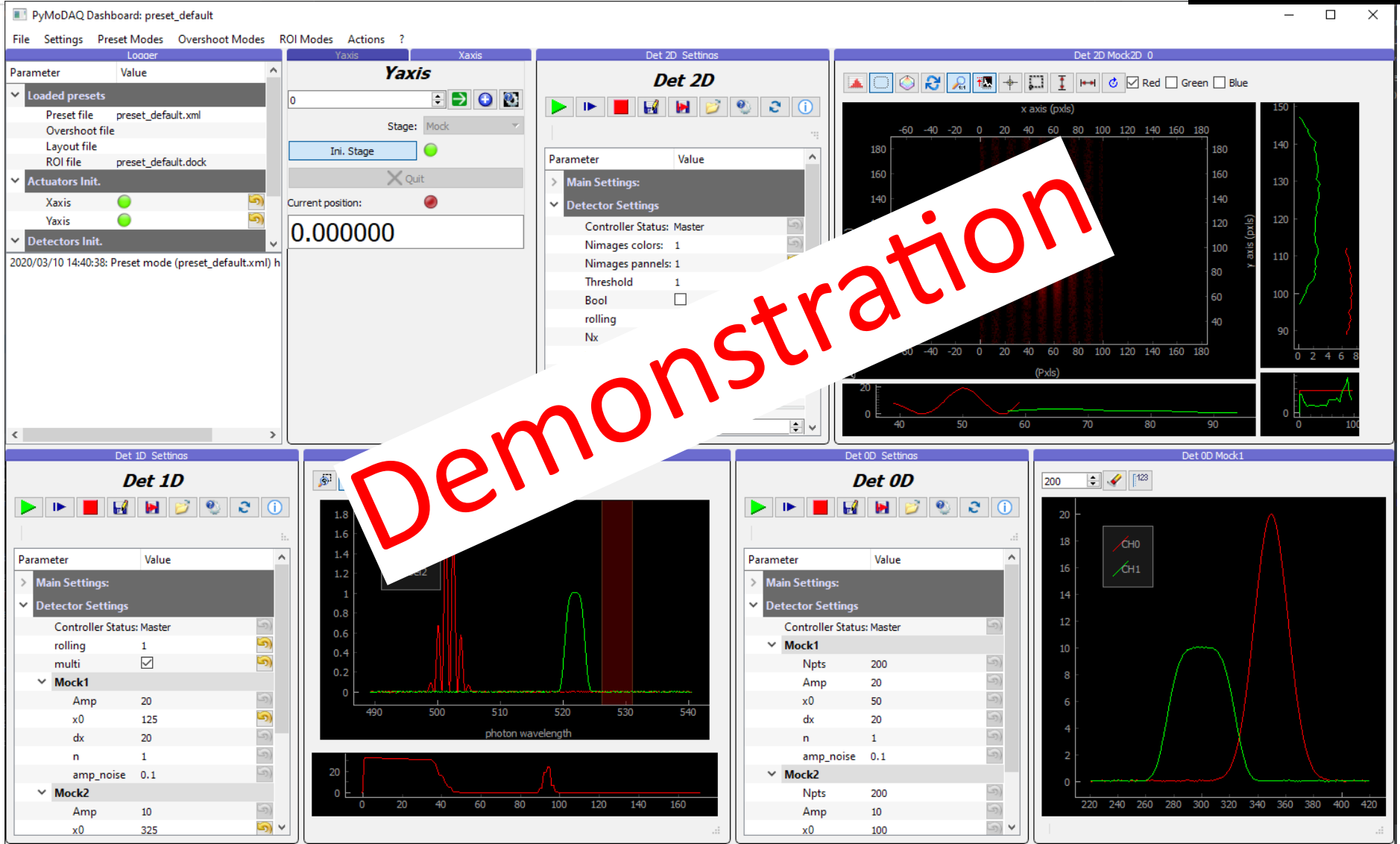
- Main Settings:
- Detector Settings
 - Controller Status: Master
 - Mock1
 - Npts: 200
 - Amp: 20
 - x0: 50
 - dx: 20
 - n: 1
 - amp_noise: 0.1
 - Mock2
 - Npts: 200
 - Amp: 10
 - x0: 100

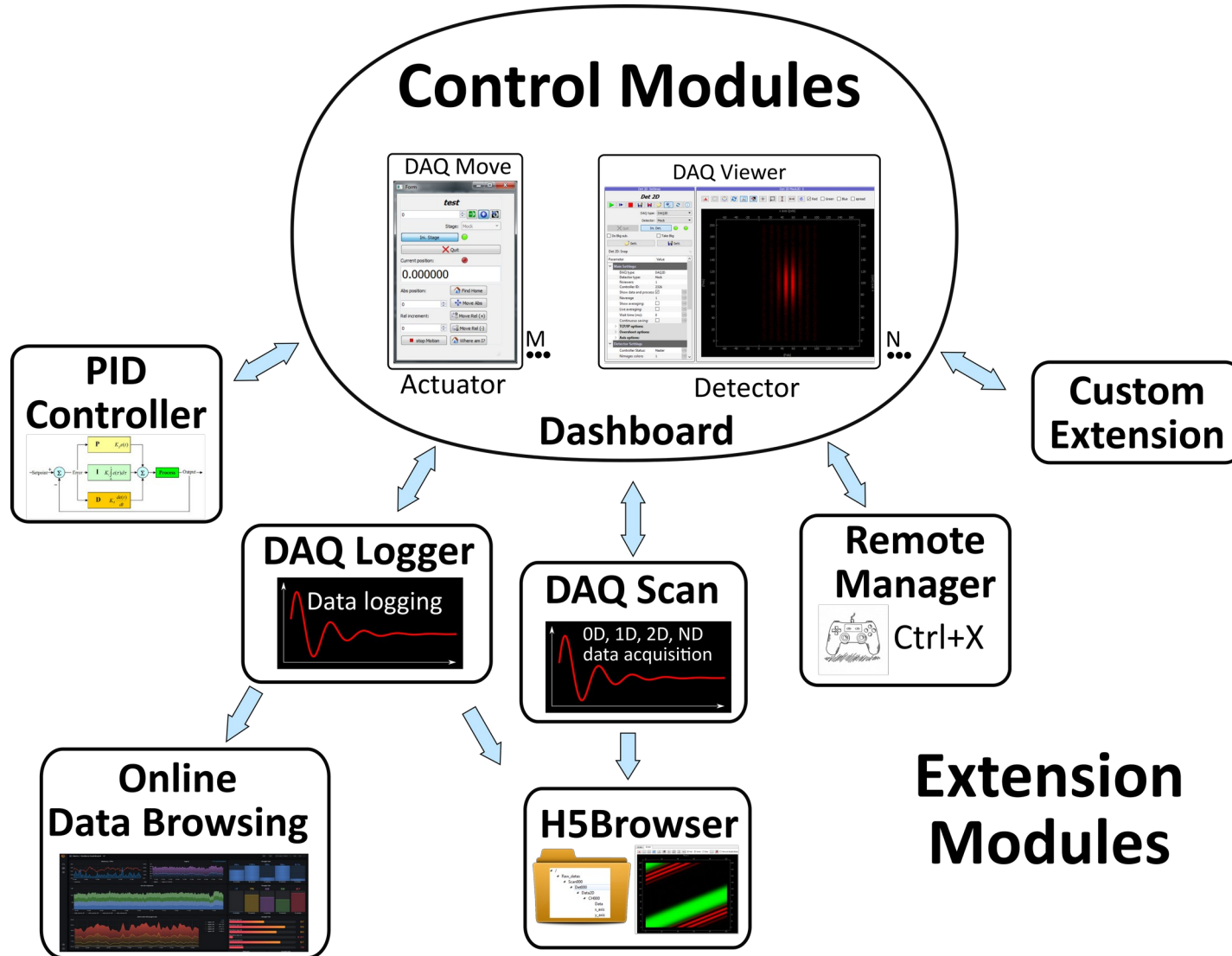
Det 0D Mock1

200

CH0

CH1





python Scan

File Settings

Parameter	Value
Actuators/Detectors Selection	
▼ detectors	
Det 1D	
Det 0D	
Det 2D	
▼ Actuators	
Theta Axis	
Yaxis	
Xaxis	

Moves done?

Detections done?

> Data dimensions

> Actuators positions

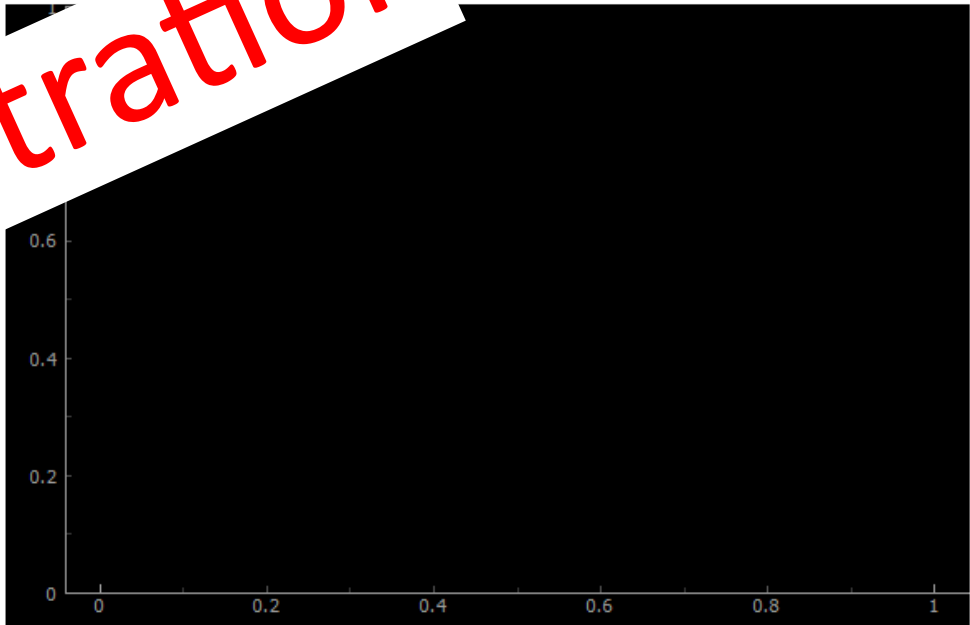
General Settings

Save Settings

Scanner Settings

Parameter	Value
▼ Scanner settings	
calculat	
	-2
stop:	3
Step:	0.5

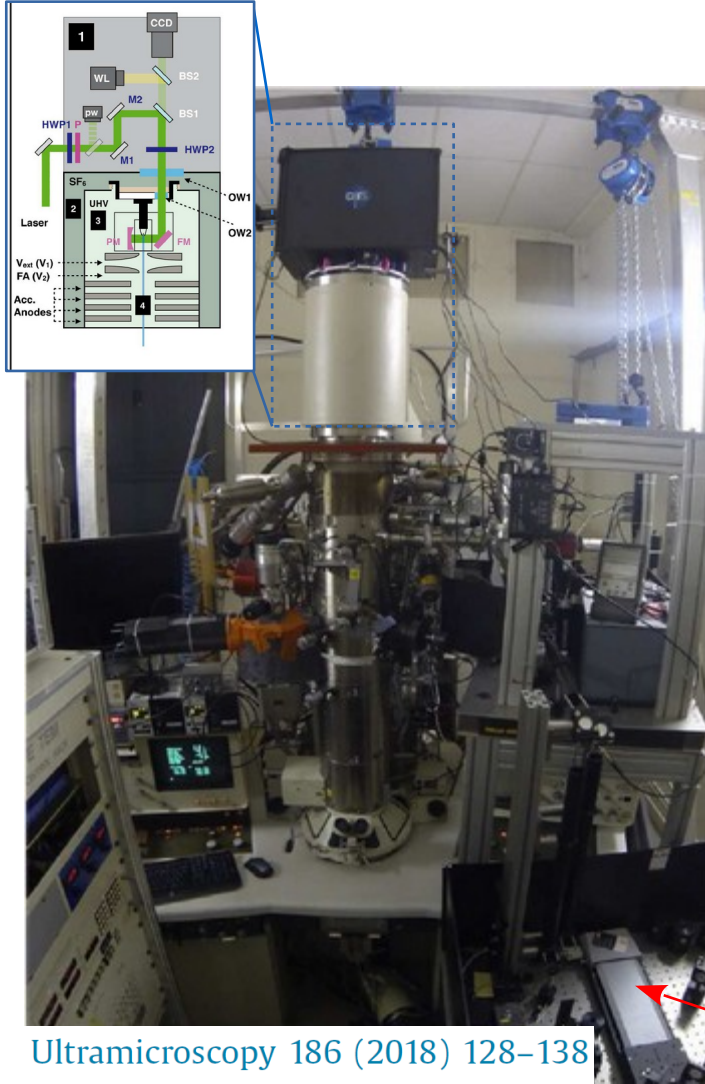
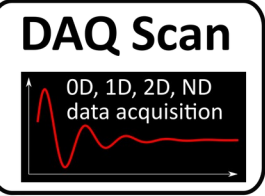
1D plot 2D plot



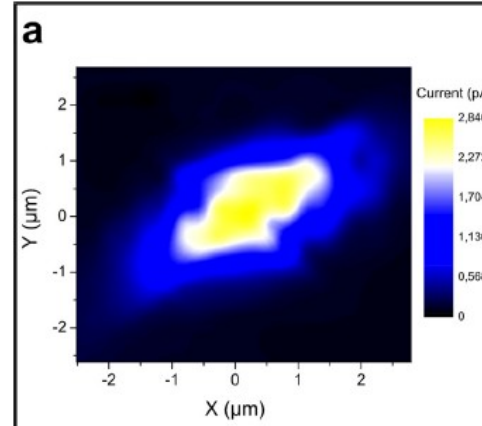
Initializing

Démonstration

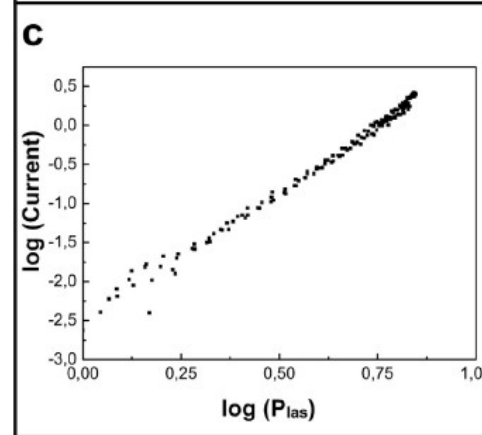
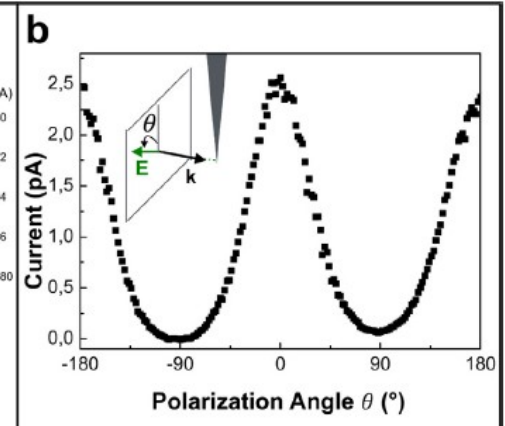
Scan Examples on the Ultrafast Electron Microscope: FemtoTEM



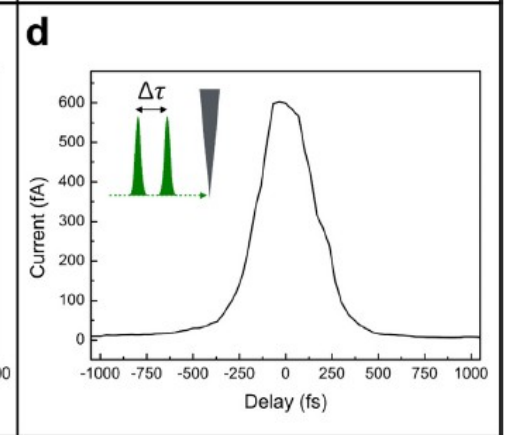
Probe current (Pico-amperemeter) as a function of laser pulse displacement, axes XY mirror M2



Probe current (Pico-amperemeter) as a function of laser polarisation (HWP2)



Probe current (Pico-amperemeter) as a function of laser intensity (HWP1)



Probe current (Pico-amperemeter) as a function of pump probe delay femtosecond

Pump-probe delay



Stay in touch and contribute

<https://github.com/CEMES-CNRS/PyMoDAQ>

The screenshot shows the GitHub repository page for CEMES-CNRS/PyMoDAQ. At the top, there is a search bar and navigation links for Pull requests, Issues, Marketplace, and Explore. The repository name is displayed as CEMES-CNRS / PyMoDAQ, with statistics for Unwatch (2), Star (0), and Fork (0). Below this, there are tabs for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. The repository description is "Modular Data Acquisition with Python" with an Edit button. A summary bar shows 3 commits, 1 branch, 0 releases, 1 contributor, and GPL-3.0 license. At the bottom, there are buttons for "Branch: master", "New pull request", "Create new file", "Upload files", "Find file", and "Clone or download".

<http://pymodaq.cnrs.fr/>

The image shows the PyMoDAQ logo, which includes a graph with a red line and the text "PyMoDAQ Modular Data Acquisition in Python". Below the logo is a search bar labeled "Search docs". To the right of the search bar is a "CONTENTS:" section with a list of links: Installation, Description, Synthesis Diagram, Class Diagram, and API documentation.

[Docs](#) » Welcome to PyMoDAQ's documentation!

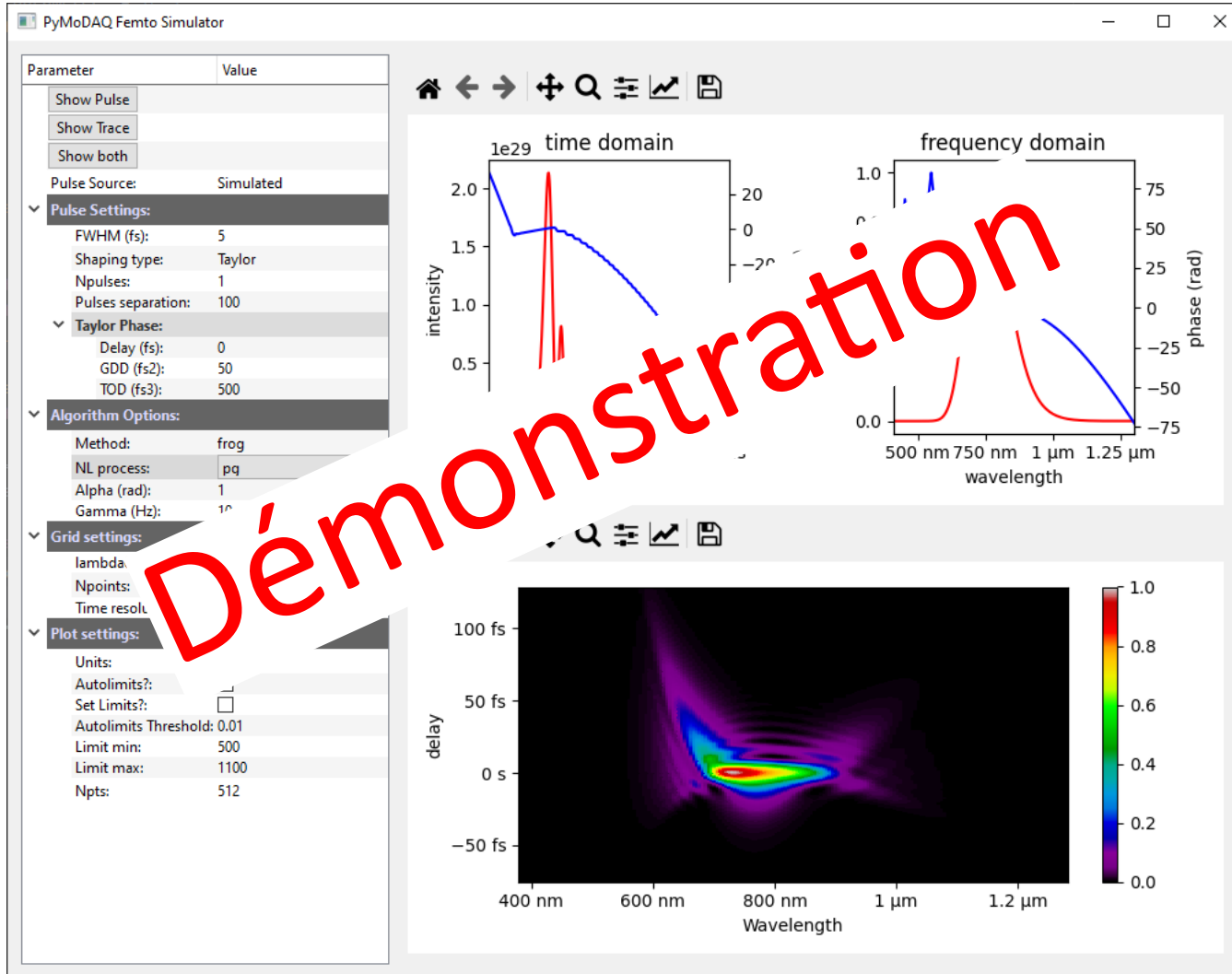
[View page source](#)

Welcome to PyMoDAQ's documentation!

Contents:

- [Installation](#)
 - [Automatic setup](#)
 - [Manual setup](#)
- [Description](#)
 - [Main Modules](#)
 - [DAQ_Move](#)
 - [Introduction](#)
 - [A paragraph](#)
 - [Another paragraph](#)
 - [DAQ_Scan](#)

PyMoDAQ-Femto: 1) Simulation





PyMoDAQ-Femto: 2) Acquisition

The PyMoDAQ Dashboard: femto interface is shown with the following components:

- Delay Control:** A slider set to 10.5, with an actuator set to 'Mock'. The current value is 10.500000.
- Spectrometer Settings:** Includes a parameter table and simulation settings.

Parameter	Value
Controller Status:	Master
Simulation settings:	
Show Spectrum:	<input type="checkbox"/>
Show Trace:	<input type="checkbox"/>
Pulse Source:	Simulated
Pulse Settings:	
FWHM (fs):	5
Shaping type:	Taylor
Npulses:	1
- Spectrometer PNPS Plot:** A line graph showing an 'NL trace' with a peak at approximately 0.95 μm .

The python Scan interface displays a 2D plot with the following details:

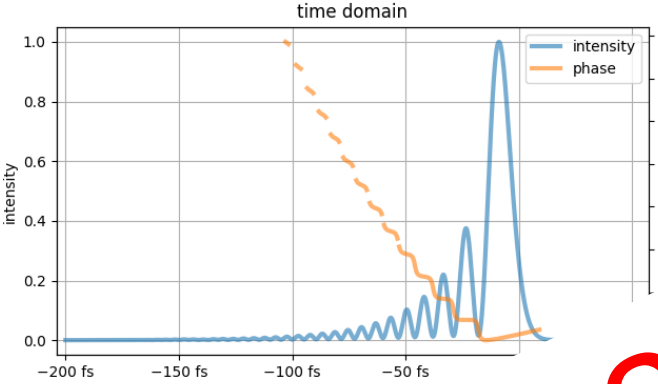
- Plot Title:** Delay (whatever)
- Axes:** X-axis is labeled 'Delay (whatever)' with values from -100 to 200. Y-axis is labeled 'Spectrometer Wavelength (nm)' with values from 0.5 to 1.1.
- Color Scale:** A vertical color bar on the right indicates intensity, ranging from 0 to 1.
- Plot Content:** A 2D plot showing a central vertical band of high intensity (red/yellow) with a horizontal spread of lower intensity (purple/blue) at the top and bottom.
- Buttons:** 'Set Scan', 'Init Positions', 'Quit', 'Load settings', and 'Save settings' are visible.

Démonstration

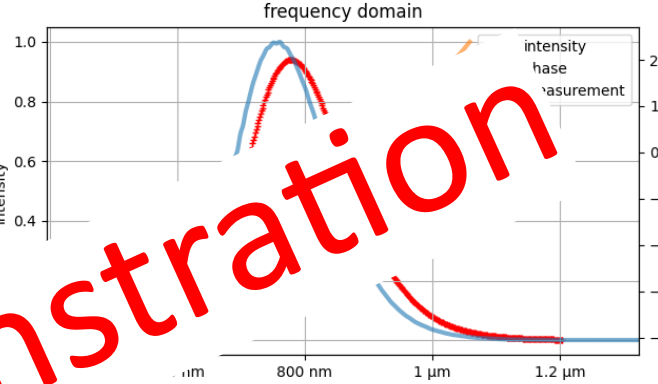
PyMoDAQ Femto: 3) ReTrieval

Data In
Processed Data
Retriever
Retrieved Data

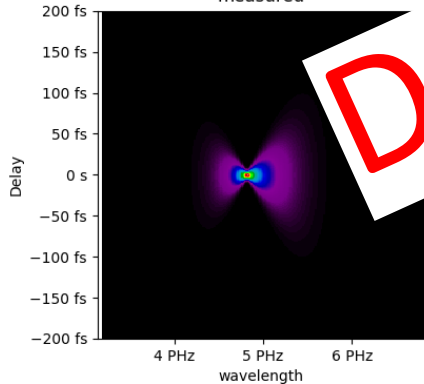
🏠
←
→
+
Q
📏
📐
📄



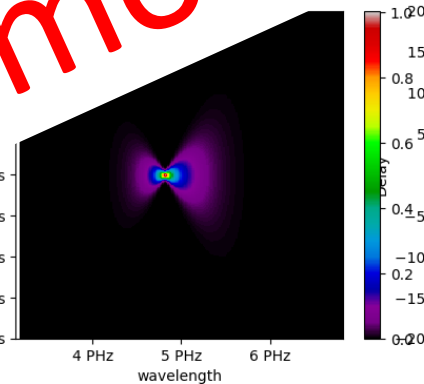
time domain



frequency domain



measured



difference

Démonstration

Settings	
Parameter	Value
Data Info	
Trace type:	frog
NL process:	shg
Trace Info	
W10 (nm)	380.123557042
FWHM (nm)	25.8822547548
Param Size	4
Wavelength Size	512
Scaling (m)	1e-09
Scaling Parameter	1e-15
Spectrum Info	
W10 (nm)	771.092246547
FWHM (nm)	76.4892136725
Wavelength Size	512
Scaling (m)	1e-09
Processing	
Grid settings:	
lambda0 (nm):	771.092246547
Npoints:	801
Time resolution (fs):	0.5
Trace limits:	
Crop Trace?:	<input type="checkbox"/>
x0:	35
y0:	113
width:	342
height:	573
Subtract background:	
Subtract?:	<input type="checkbox"/>
wl0:	273.97260274
wl1:	508.219178082
<input type="button" value="Process Spectrum h"/>	
<input type="button" value="Process trace"/>	
<input type="button" value="Process Both"/>	
Retrieving	
Algo type:	copra
Verbose info:	<input checked="" type="checkbox"/>
Max iteration:	30
Initial Pulse Guess	
FWHM (fs):	100
Phase amp. (rad):	0.1
<input type="button" value="Start Retrieval"/>	
<input type="button" value="Stop Retrieval"/>	