# Instructions about the code

## Instructions

The basic idea is to click the button from left to right and from top to bottom.

**Reset**: resetting the FROG trace back to the state when you finish loading the FROG trace and the measured spectrum/background if they’re loaded. It can be clicked anytime.

1. **FROG trace:**

**Load FT**: load the FROG trace file. Currently, two types of files are supported: the raw FROG trace (with an extension “.RAW”) and the VideoFROG’s exported txt files (the one named “FROGtrace …txt”). After loading the FROG trace, automatic threshold value will be found for “cleaning” in “Resize” tab (This will be explained below).

A close-up of a computer screen

AI-generated content may be incorrect.

**Load spectrum**: load the measured spectrum from an optical spectrum analyzer (OSA). This can be a good candidate of an “initial condition” for the pulse retrieval algorithm and it’s useful for checking the quality of the retrieval by comparing the spectrum measured from two devices, FROG retrieval and OSA. (Check “Marginals” description below.)

It supports measurements from several devices. However, there are cases where a spectrum file has a non-supported file format, so this code also supports “user-defined” format, that provides a linear-scale spectrum to this FROG-retrieval code. Please find “user\_read\_spectrum.m” that comes with this code for details.

A screenshot of a computer

AI-generated content may be incorrect.

**Load background spectrum**: load the measured background of the spectrum above.

**Estimate the background**: This open “backcor” GUI for background estimation. You can play with “demo\_for\_backcor.m” to see how it works. It has a GUI, so it’s straightforward.

1. **Resize:**

**Crop**: If the pulse is short, get rid of useless information of extra delay window; or if the pulse is narrowband, remove useless frequencies. Just click two points on the popping figure and their positions will be used to crop the delay and the frequency windows.

**Clean**: The FROG trace retrieval needs only the central part of the FROG trace. The background needs to be cleaned; therefore, a threshold value needs to be defined. Adjust the threshold value until it shows only the central part of the FROG trace, and the background is cleaned appropriately. Sometimes it’s worth playing with this several times for retrieving. Automatic threshold value will be found after loading the FROG trace. It’s found by increasing the threshold value from a noise floor until the D4Sigma of the trace stops changing significantly.

Cleaning the background noise is crucial for pulse retrieval. Those tiny noise at the edges of the trace makes the retrieved pulse noisy around the edges too and generates a weird retrieval result or prevents convergence of the retrieval process.

After clicking “Clean”, check MATLAB command window. It shows D4Sigma X and Y values. They represent the size of your central FROG trace, so they should be smaller than the delay and frequency windows. If they’re too large, click “**Reset**” and find the threshold again.

**Resize**: Resize your FROG trace window to include only the important/central part for pulse retrieval.

1. **Pulse Retrieval:**

**BW (frequency)** and **BW (wavelength)** are used when the button of “using the measured spectrum as the initial guess” of pulse retrieval algorithm isn’t clicked and the Gaussian spectrum is used instead.

**Duration**: the pulse duration. With the spectrum/bandwidth and the duration, “calc\_chirp.m” is used to find the chirp.

**chirp sign:** the sign of the chirp to lengthen the transform-limited pulse computed from the spectrum guess to the user-input duration above. 1 is the positive chirp; -1 is the negative chirp.

**Iterations** and **Threshold**: The retrieval will only stop if it reaches the maximum iterations or convergence threshold. You can also stop it by clicking “stop” in the retrieving figure when retrieving or closing the figure. Typically, a good retrieval needs at least 100 iterations, so 200 is a good candidate.

**Time window**: It’s default to twice the delay window after resizing. It controls the frequency resolution because . You can increase it to increase the frequency resolution of the retrieval result.

Since the guessed bandwidth and time window affects the retrieval sampling, “Resampling” is required to run again whenever they are changed.

**Resample**: Based on the time window, this finds the correct number of sampling points () until it covers the entire frequency window of the FROG trace after resizing because .

Note that during resampling, calibration of the FROG trace with the measured spectrum, if provided, is applied. Therefore, if the user loads a different spectrum, “Resampling” is required.

**Retrieve**: Run the pulse retrieval.

**Pulse energy**: The retrieved pulse will be normalized to this input pulse energy. If it’s zero, it’ll temporally be normalized to one.

1. **Analysis:**

**FROG trace:** Plot measured and retrieved FROG traces.

**Marginals**: Plot the delay marginals and the frequency marginals. If the measured spectrum is provided, it also plots the frequency marginals from this spectrum. If the retrieved frequency marginal can match well with the one from the measured spectrum, the pulse retrieval is excellent. I highly recommend you use the measured spectrum for the comparison of the frequency marginals.

**Pulse**: The values of the pulse duration is shown in the MATLAB command window. The pulse figure is plotted. If “Strehl ratio” box is checked, the Strehl ratio and the transform-limited pulse duration are shown in the command window, as well as plotting with the transform-limited pulse.

**Compare spectra**: It compares the retrieved spectrum with the measured spectrum, which is also a useful information representing whether your retrieval is good or not. If there is no provided measured spectrum, it plots only the retrieved spectrum.

**Save results**: It saves the “pulse in time”, “time and frequency vectors”, “center frequency”, “delay of the FROG trace”, “measured and reconstructed FROG traces”, “transform-limited information”, and “spectral information.”

## Notice

I found out the retrieval algorithm is sometimes strongly dependent on the initial condition, especially for a chirped pulse. If possible, measure the spectrum experimentally and use this as an initial guess for the algorithm. By doing this, the only degree of freedom is the pulse duration that determines the chirp of a pulse. By playing with this parameter, it’s easy to obtain good pulse retrieval. You may even avoid the ambiguity of FROG time/frequency reversal with this measured spectrum.

If you don’t have a separately measured spectrum, a Gaussian spectrum is set as an initial condition instead, but the parameters of the bandwidth and the pulse duration are necessary.

Because the algorithm uses some random indexing, sometimes it can be helpful just to run the “Retrieve” again with the same parameters. But you should be able to get a good result for at least two runs. If you can’t, change the parameters for the algorithm.

“Time window” is for controlling the frequency resolution. For a narrowband pulse, you might want to use a larger number than the default one. You need to re-sample it again after changing the time window.

**Extra detail:**

When using the measured spectrum, the chirp is calculated by “calc\_chirp.m” which uses an optimization algorithm to add the chirp to the spectrum until it gets the user-specified duration. If you specify it correctly, it should be already close to the real pulse if it’s a dechirpable pulse.

# Important check for the correctness of a FROG measurement

In this code, the retrieval can use the spectrum measured independently by another optical spectral analyzer (OSA) as an initial guess of the retrieval algorithm. Beyond this, the known spectrum can be used for several crucial purposes:

1. **Calibrate the frequency position of the measured FROG trace.**This might be the problem with our FROG only. Our FROG from Mesaphotonics will have a wrongly-offset frequency, compared to the spectrum measured by other OSAs. Therefore, this code uses OSA’s spectrum to calibrate the FROG trace by shifting it up and down in frequency.
2. **Verify the correctness of the FROG measurement**
   1. With a known spectrum, we can compute its **frequency marginal** (the spectral version of temporal autocorrelation). By comparing this frequency marginal with the one derived by the FROG trace, we can check if our FROG measurement is correct.
   2. The code can **compare the retrieved spectrum with the measured spectrum** (independently from an OSA). This can also check if a FROG measurement is correct.

Potential reasons for wrong measurements are (1) non-optimal beam alignment of the FROG that induces asymmetry in FROG trace and (2) existence of spatial chirp. Typically, we use a FROG to measure the dechirped pulse from a laser. If the dechirping stage, usually from a grating pair, isn’t correctly set, the dechirped beam might exhibit a spatial chirp (different spatial position has different frequency components). The beam into a FROG device might generate SHG signal of only a part of spectrum due to spatial chirp, creating a FROG trace with a narrower bandwidth than a true spectrum.

# Instructions for running examples

#### ANDi (all-normal-dispersion) laser pulse:

Some parameters I’ve tried:

The final output is a , pulse.

I found out for this ANDi pulse, the initial condition doesn’t matter at all.

#### Linear-Mamyshev-oscillator pulse:

This FROG was experimentally taken from the (un-dechirped) chirped pulse directly coming from a linear Mamyshev oscillator. I found out that the pulse retrieval is sensitive to the initial condition.

1. Use the measured spectrum as an initial guess:

Retrieval of a chirped pulse can be very sensitive to the initial guess (spectral shape and phase information [controlling the chirp and the pulse temporal duration]). Therefore, you can see that the retrieved pulse becomes structured with durations with a bad guess, other than 6 ps in this example.

1. Use the Gaussian spectrum as an initial guess:

and gives the best result.

The results are approximately the same for these parameters. Using extremely small BW gives an unrealistic structured pulse.