TimeTagger Driver Documentation

Henri Morin
henri.p.morin@gmail.com

October 2022

Contents

1 Introduction

The goal of this document is to serve as documentation for the `swabian_timetagger.py` code for the CERC group. The TimeTagger does have an associated webapp but the gui was created as it will be more customisable.

1.1 Required Software

To have access to the `TimeTagger` python library, you must install the timetagger software suite, as it is not installable by `pip` or `conda`. Find the latest software at `https://www.swabianinstruments.com/support/downloads/`.

See TimeTagger documentation at: `https://www.swabianinstruments.com/static/documentation/TimeTagger/index.html`

TimeTagger

The code creating the TimeTagger instance only needs to be modified if more than one tagger is connected to one computer at the same time. This seems unlikely and so it is not taken into account. If ever it needs to be considered, you must specify the serial number of the tagger in question in the `createTimeTagger` call. Note that you need to install the TimeTagger Software (`https://www.swabianinstruments.com/support/downloads/`) to install the `TimeTagger Python` package. It does not seem available through the normal means of installing `Python` packages.

2 `swabian_timetagger.py`

This is the GUI which interfaces with the TimeTagger.

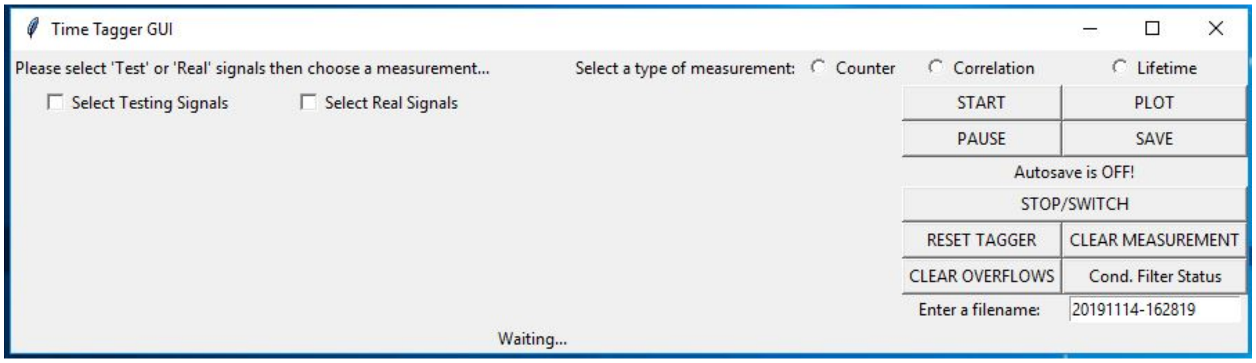


Figure 1: Tagger GUI at startup after choosing the status of the filter

2.1 Controlling the GUI

There is a certain “order of operations” required such that the gui does not break. If ever anything breaks, closing and restarting it will fix it. Remember not to do things in the order which caused the errors. The biggest one to mention is that the radio buttons for the selection of the type of measurement do not act properly as radio buttons. As such, to change measurement or to change the settings on the same type of measurement, you must `STOP/SWITCH`, then select the new measurement. If anytime you see some widgets are superimposed on top of each other, quit and restart since the gui will not function properly at this point. One of those sets of widgets will remain on screen even when no measurement is selected.

Default values for any parameter used, whether direct tagger or a measurement setting should have the default value be obvious in the code, making it easier to change if necessary. Furthermore, functionality was added such that when you change a default value then change measurement, when you return to that measurement, the inserted value will now be what you previously selected, not the initial default value.

The general order of operations to do a measurement is:

1. Run `swabian_timetagger.py`
2. Select whether or not you wish to turn on the conditional filter for this instance of the program (see sec. 2.2 for details)
    - If `yes` is selected, you will first enter the channel(s) to use as a trigger (usually the APD), then the channel(s) to filter (usually the pump)
    - If `no` is selected, the startup continues as normal
3. Select type of signal: Test or Real ("test" being the internal test signal and "real" being a real signal from an APD)
4. Input channel trigger voltages to desired channel(s) if the default values need changing
5. Input the channels you wish to use
    - Note you must separate each channel used by a space.
    - Ex. To use channels 1 and 2, you input: "1 2"
6. Select `Set Channel Triggers` to set the triggers on the desired channels.
    - Note that simply entering the new values will not change the trigger on the TimeTagger. You must click the button.
7. OPTIONAL: If you wish to set non-default delay and dead time, select `More Tagger Parameters` and enter the desired values. You may select `See All Set Tagger Parameters` at any time to see what is set, if you forget.
    - Note that you must select `See All Set Tagger Parameters` to see the current values set for these parameters. Clicking on `More Tagger Parameters` will always insert default values, not current values.
    - Note that the channels involved in `More Tagger Parameters` are the same as the trigger voltages. Thus define the selected channels there *first* before setting their extra parameters.

8. Select the measurement type you wish to use (Counter, Correlation, Lifetime)
9. Fill out the measurement parameters
   - Note that measurement parameters now persist through multiple measurements. Thus you do not need to re-enter ALL parameters per measurement, only the ones you want to change.
10. Click `START`. The measurement is now running and "Running..." is printed in the console as proof that data is being acquired.
11. OPTIONAL: Click `PLOT` to see an animated plot of the measurement. Note that some important numbers (measurement depending) are updating on the gui itself such that plotting is not strictly necessary to see valuable information. Also note once you decide on a measurement, plot may be selected at anytime. You may even close it and reopen it.
12. OPTIONAL: At any time, you may specify the filename for this measurement. The default is the day and time which the gui was started.
13. Once you have acquired enough data, while it is running, click `SAVE`. This will save the data from the measurement under `\<script-directory>\Tagger Code\<measurement-type>` `\<current-day\>`.
    - Note that an autosave feature is implemented with an arbitrary number of minutes (currently set at 5). It will save data under the current file name after that time expires. Furthermore, it is only turned off once a measurement is over. If you **pause** a measurement, the autosave will still be working.
    - A indicator is set in the `Main Controls` section of the GUI to explicitly show when this feature is activated
14. After a mesurement, there are multiple possible actions:
    - To quit the gui, click the "X"
    - To clear the measurement to start another of the same type with the same settings, click `CLEAR MEASUREMENT`
    - To stop acquisition of the current data set, click `PAUSE`. To restart this measurement, click `START`
    - To change measurement type, click `STOP/SWITCH`, then the new measurement type
    - To change the settings on the tagger itself, you may change them on the fly or reset to defaults using `RESET TAGGER`
    - If the tagger's status light is continuously red or that plots are not updating, overflows are occurring. Click `CLEAR OVERFLOWS` to clear them. Check the console to see the number of overflows, which are printed there. Note that 13 is essentially no overflows.

2.2 Conditional Filtering

In order to get the lifetime measurements, one of the channels is connected to a pulsed source. However some frequencies at which one wants to take measurements at is larger than the allowable data transfer rate between the TimeTagger and the PC (usually 8-9 million counts/s). To reduce this data stream, one can employ *conditional filtering*. The documentation provides a supplementary guide on the filtering (`https://www.swabianinstruments.com/static/documentation/TimeTagger/sections/inDepthGuides.html`). The following plot is taken from this guide since it shows a nice visualisation of what is happening.

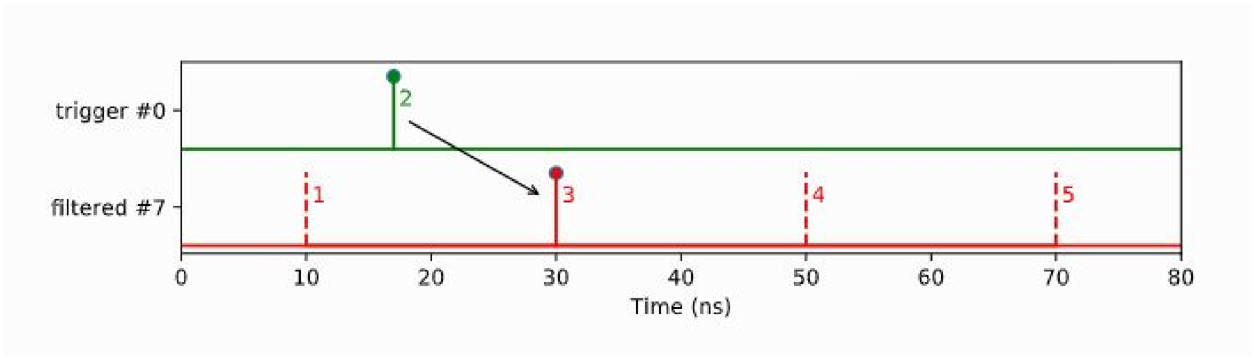


Figure 2: Schematic of Conditional Filtering Process

This allows the reduction of the data stream, allowing high pulse rates to be used. Note that in the current implementation of the filter, it is either on or off, and this is decided at startup. There is no current way to change the status of the filter on the fly. At any time however, you may choose `Cond. Filter Status`, which will print the list of channel(s) which are used as trigger, then the list of channel(s) which are filtered. Note that the choice for each set is not limited to one channel. If more than one channel is desired, the same "multi-channel" convention as the channel triggers apply ie write "1 2", with the space. Also note that the filter is applied per *channel*, not per *measurement*. Thus it will still be active for all measurements, not just the lifetime where it's required for the measurement to work. The filter does not seem to affect the other measurements. Finally, note that there is only one command in the code to *either* turn on the filter *or* turn it off. This is why the filter is called exactly only once.

2.2.1 Problems with the Filter

There are some issues with the filter and its implementation. In all measurements, there are updating counters on the gui updating the counts for 2 channels. However, sometimes, when the filter is activated, we can still preform a measurement but these counts do not update past their default 0 value. It is unknown at this time the cause of this bug.

2.3 Type of Signal

Two types of signals are used: "Testing" and "Real". Note that the use of `RESET TAGGER` will clear the settings for *both* testing and real signals.

2.3.1 Testing

The testing signal is the built-in test signal (~ 0.8 to 0.9 MHz) used in order to test the functionality of the gui when there is no dot source available (like when testing at a desk). In real operation, this would almost never be used.

Click the check box for each signal you want to activate. You may even do this while the measurement is running and see the impact of the channel being activated/deactivated.

If you reset the tagger with the test signal running, the tagger will properly reset and the test signals will be turned off. However, the check buttons will not deactivate. If you choose to continue using the test signal, you'll need to reactivate them.

2.3.2 Real

Real signal is a signal from the APD's, caused by a real photon from the dot, hence the name.

Enter the desired voltages for each desired channel. Note it needs to be $0 \leq V \leq 2.5$. Enter the desired channels to set voltages. Note each channel number must be separated by a space. Hence setting channels 1 and 2, the input is: "1 2". Then click `Set Channel Triggers`.

If you wish to set non-default delay and dead time, select the `More Tagger Parameters` button and enter the desired values. Note only do this once the all the channels have been inputted since the pop-ups will use the same channel numbers as the trigger voltages.

Finally, the `See All Set Tagger Parameters` button allows one to see the settings which have been set. For all channels selected this means: trigger voltage, delay, and dead time.

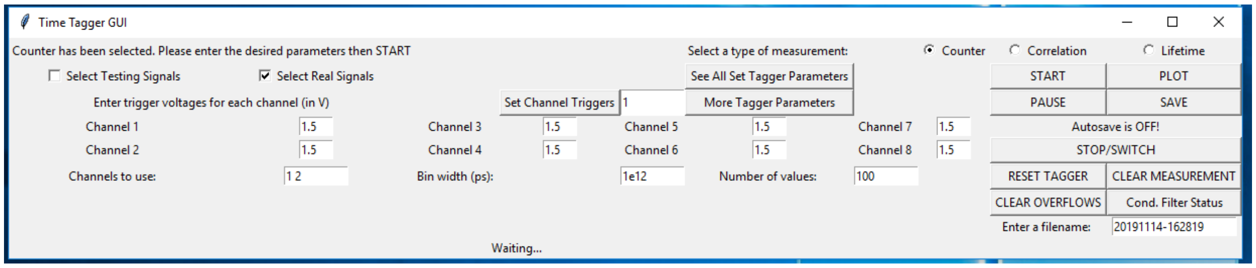
2.4 Type of Measurements

From the multitude of measurements the tagger is able to do, three were targeted specifically: Counts, Correlation, and Lifetime (Histogram).

The tagger is able to run many measurements at the same time, simply by creating the objects for each desired measurement sequentially. The difference between them is simply what do you do with each tag. However, the nature of the gui is only one measurement at a time. Ideally, this means that once you select a measurement, only that measurement is created and you **can not** choose another to add dynamically. However, since it is useful to see the counts at the same time, there is an additional counter object which is created at the same time as the correlation and lifetime measurements.

All defaults may be adjusted manually in the code. Look at the beginning of the `start_measure` function to change *measurement* defaults.

2.4.1 Counter


Time Tagger GUI

Counter has been selected. Please enter the desired parameters then START

Select a type of measurement: [✓ Counter] Correlation Lifetime

Select Testing Signals [✓ Select Real Signals]

See All Set Tagger Parameters | START | PLOT

Enter trigger voltages for each channel (in V) | Set Channel Triggers: 1 | More Tagger Parameters | PAUSE | SAVE

Channel 1	1.5	Channel 3	1.5	Channel 5	1.5	Channel 7	1.5
Channel 2	1.5	Channel 4	1.5	Channel 6	1.5	Channel 8	1.5

Channels to use: 1 2 | Bin width (ps): 1e12 | Number of values: 100

Autosave is OFF! | STOP/SWITCH | RESET TAGGER | CLEAR MEASUREMENT | CLEAR OVERFLOWS | Cond. Filter Status

Enter a filename: 20191114-162819

Waiting...

Figure 3: GUI set for Real Counter Signal

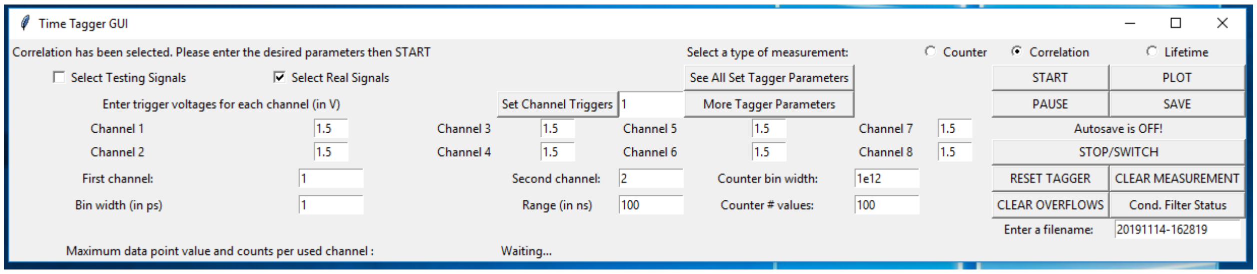
Three settings exist for the Counter: Channels to use (same syntax as the channel triggers), Bin width (self explanatory), and Number of values (total number of bins kept in the buffer).

Settings	Defaults
Channels to use	1 2
Bin width (ps)	1e12
Number of values	100

Table 1: Counter Settings and Defaults

The way the measurement is setup in the code is that there will be a list of size "Number of values", for each channel, where all elements are initialized at 0. Once a time period of size "Bin width" has passed, the number of counts in that bin is added to the *last* index of the list. This gets shifted every new iteration, as a rotating buffer. Thus the last value in the list is the most recent value. While the measurement is running, the `Waiting...` turns into this most recent value, which updates every second.

2.4.2 Correlation


Time Tagger GUI

Correlation has been selected. Please enter the desired parameters then START

Select a type of measurement: Counter [✓ Correlation] Lifetime

Select Testing Signals [✓ Select Real Signals]

See All Set Tagger Parameters | START | PLOT

Enter trigger voltages for each channel (in V) | Set Channel Triggers: 1 | More Tagger Parameters | PAUSE | SAVE

Channel 1	1.5	Channel 3	1.5	Channel 5	1.5	Channel 7	1.5
Channel 2	1.5	Channel 4	1.5	Channel 6	1.5	Channel 8	1.5

First channel: 1 | Second channel: 2 | Counter bin width: 1e12

Bin width (in ps): 1 | Range (in ns): 100 | Counter # values: 100

Autosave is OFF! | STOP/SWITCH | RESET TAGGER | CLEAR MEASUREMENT | CLEAR OVERFLOWS | Cond. Filter Status

Enter a filename: 20191114-162819

Maximum data point value and counts per used channel : Waiting...

Figure 4: GUI set for Real Correlation Signal

Six settings exist for the Correlation. Four directly for the correlation measurement and two for the associated counter.

Channels 1 and 2, and Bin width are self explanatory. The two counter related settings are the same as for the Counter measurement (see sec. 2.4.1), minus the channel selection since it is obvious which one you want (the ones inputted for the correlation). The last setting is the range, that is the range of the x-axis you wish to see.

For this measurement, the `Waiting...` turns into 3 continuously updating numbers: the maximum for the data stream and the counts in each channel. The maximum is also updating directly on the plot.

If anyone ever needs to adjust bin width and/or range, there is some subtlety involved that needs some explanation.

The programmer wants to determine how big the data structure needs to be and thus the correlation measurement objects takes in bin width and number of values as parameters. The experimentalist wants to input the bin width and the range for the correlation as parameters. Thus some math needs to be done behind the scenes such that the user does not have to input directly the required number of bins. Luckily, the following formula (1) helps:

$$binwidth * bins = total\ \ x\text{-}axis \tag{1}$$

Settings	Defaults
Channel 1	1
Channel 2	2
Bin width (ps)	1
Range (ns)	100
Counter bin width (ps)	1e12
Counter # values	100

Table 2: Correlation Settings and Defaults

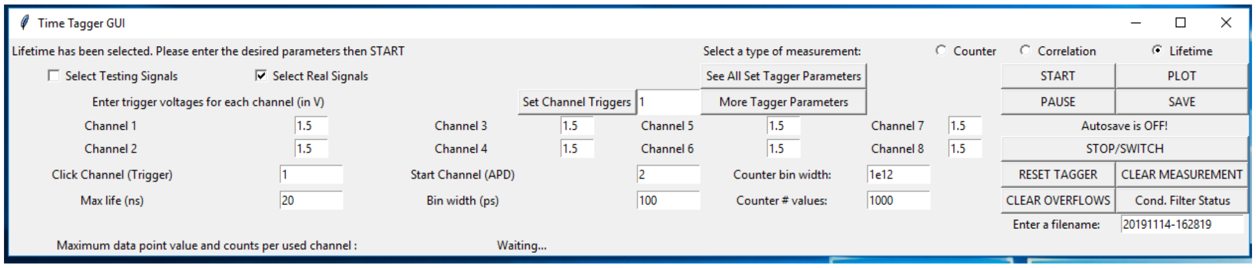
Note on normalization functionality of correlation measurement

The tagger has the ability to return the correlation data under two forms, what I call "Raw" and "Normalized". Raw is exactly what the tagger receives, no post-processing, and is what is normally plotted for the correlation measurement. Normalized is that data but normalized according to:

$$g^{(2)}(\tau) = \frac{\Delta t}{binwidth * N_1 * N_2} \cdot histogram(\tau) \tag{2}$$

where Δt is the capture duration, N_1 and N_2 are the number of events in each channel. This equation is found in the TimeTagger documentation for the Correlation measurement. Note that the normalization does NOT make the maximum value in the normalized plot 1. We are unsure of the use of the normalized so this functionality is no longer included in the gui. This is only a note of its existence. During testing, the look of both kinds of plots remained identical, simply the scaling differed. In the "normalized" plot, the average horizontal part of a CW $g^{(2)}$ was kept to ~ 1.

2.4.3 Lifetime



Lifetime has been selected. Please enter the desired parameters then START

Select a type of measurement: [✓ Lifetime]

[ ] Select Testing Signals [✓] Select Real Signals

Enter trigger voltages for each channel (in V) — Set Channel Triggers: 1

Channel 1	1.5	Channel 3	1.5	Channel 5	1.5	Channel 7	1.5
Channel 2	1.5	Channel 4	1.5	Channel 6	1.5	Channel 8	1.5
Click Channel (Trigger)	1	Start Channel (APD)	2	Counter bin width:	1e12		
Max life (ns)	20	Bin width (ps)	100	Counter # values:	1000		

Autosave is OFF!

Enter a filename: 20191114-162819

Maximum data point value and counts per used channel : Waiting...

Figure 5: GUI set for Real Lifetime Signal

This measurement is called “Lifetime” in the gui and in the code but there is not a “Lifetime” measurement in the tagger, if you look closely in the documentation. It is instead a “Histogram”. The documentation states that this is the measurement to use for a lifetime measurement. Four settings exist for this measurement and the two extra counter settings also persist.

Settings	Defaults
Click Channel	1
Start Channel	2
Max life (ns)	20
Bin width (ps)	100
Counter bin width (ps)	1e12
Counter # values	100

Table 3: Lifetime Settings and Defaults

Eq. 1 is used again here since the Histogram object requires that value and not Max life. Max life is the total x-axis, analogous to the “range” for the correlation measurement. For this measurement, the `Waiting...` text is the maximum value of the measurement. Count values also update on the gui (note the problem with the updating counts mentioned in sec. 2.2).

Issues with Lifetime As mentioned before, using the filter is necessary in order to do a lifetime measurement if the pump is set to repeat faster than $\sim$ 10MHz. Ideally, this would mean that `Start Channel` is the pump and the `Click Channel` is the APD. This would mean that there is a large “spike” in the signal before exponentially decaying, then repeating after the repetition rate of the pump (this is the lifetime signal). However, either in this configuration or the ones with the channels reversed, ie `Start Channel` is the APD and the `Click Channel` is the pump, the counts *do not* update while the measurement is running. This simply results with the x-axis being flipped but not necessarily with respect the to middle of the plot as previously plotted, requiring some “exploration” to find the signal. Unfortunately, this is not a problem with the webapp, which means this is a problem with my code, somehow. Good news is that one can re-flip the data to the signal. But the bug is still there (as of **2019-11-15**). Playing with the settings will show what I mean.

2.5 Main Controls

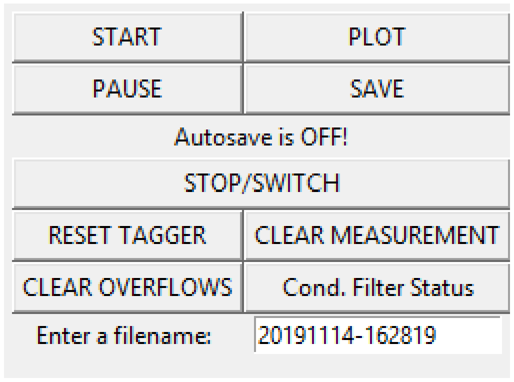


Figure 6: Main Controls

These controls are always present. Some may have dedicated sections with more details but will cover them generally in the following lists.

- `START`: Starts the desired measurement either for the first time or restarting from a `Pause`. `Running...` will continuously appear in the console to show that the measurement is still acquiring data.
- `PLOT`: Plots the acquired data. See sec. 2.5.1.
- `SAVE`: Saves data. See sec. 2.5.2
- `PAUSE`: Pauses the acquisition of the current measurement. Restart it using `START`.
- Autosave Status: Shows the status of the autosave functionality
- `STOP/SWITCH`: Stops the current measurement and clears it from the radio buttons. Use it to change measurements or to change the settings for the same type of measurement.
- `RESET TAGGER`: Resets ALL tagger settings to default.
- `CLEAR MEASUREMENT`: Clears the current data buffer for the current measurement. Use this to restart acquiring the same measurement with same settings.
- `Cond. Filter Status`: Prints the status of the conditional filter in the console. See sec. 2.2 for details
- `CLEAR OVERFLOWS`: Clears overflows if tagger receives too much signal at once. LED is red in this case. Resetting also clears them but this will print in the console the exact number. Note the any live-updating number or plot will no longer update while there are too many overflows.
- `Filename`: The file name under which data will be saved. Default is date and time the tagger was launched. The files themselves will be saved with suffixes attached to this filename. See sec. 2.5.2

2.5.1 Plotting

To get the animated plots, you must have the graphics backend of Spyder be “automatic”. To check go to:

1. Tools
2. Preferences
3. IPython Console
4. Graphics Tab
5. For the graphics backend, choose “automatic”

The plots will not be animated if this is not the case. The default is “inline” which will show the plots in the `IPython Console` window. This was done since embedding `matplotlib.pyplot` plots into `tkinter` is very difficult. Changing the backend takes care of this. Note to save the plots, you must use the pop-up directly. However this may not be necessary since you will most likely replot the data to get the plots looking like you want. Note that the plotting was not tested outside of Spyder. Thus, I do not know how the default IDE would plot the animation.

The default plot for Counts is counts vs. time (s). This means the x-axis data is hard-coded from ps to s. For Correlation, coincidences vs. time (ns). x-axis data is now hard-coded from ps to ns. For lifetime, it is the same as correlation, ie ps to ns.

To save the plot, you must use the button on the pop-up window, unless you wish to replot the data later, to better suit your needs.

2.5.2 Saving

Saving the data generated evolved over time but is now better structured. The data is saved under the following directory: `\<script-directory>\Tagger Code\<measurement-type>` `\<current-day\>`

The name of the file is: `<filename>+-INDEX.csv` for x-axis data and `<filename>+-DATA.csv` for y-axis data. Further distinction is made for the correlation and lifetime measurements to distinguish the data sets for those measurements and their associated counter measurement.

The data was selected to be saved as a `csv` but it is possible to save it as another type. Simply change the file extension in the `save_data` function.

The autosave features variable is found in the `__init__` function of the GUI. It is in minutes, currently set to 5. Thus, every 5 minutes, the current measurement will save data under the filename currently in the entry, overwriting *any* file with the same exact name. Thus if someone runs an experiment, saves, changes settings and starts again, if the name is not changed then there is a risk of losing the original data. Keep in mind to change the name in the entry *immediately* after you save and want to change settings, such as not to disturb the saved data. Note that at every save, a notice of successful saving is printed in the console.