PyCorrFit - Generic cross-platform FCS fitting tool
Software Guide

Paul Müller
Biotechnology Center of the TU Dresden
November 30, 2013

Contents

1 Introduction

1.1 Preface

PyCorrFit emerged from my work in the Schwille Lab[1] at the Biotechnology Center of the TU Dresden in 2011/2012. The program source code is available at GitHub[2]. Please do not hesitate to sign up and add a feature request. If you you found a bug, please let me know via GitHub.

PyCorrFit was written to simplify the work with experimentally obtained correlation curves. These can be processed independently (operating system, location, time). PyCorrFit supports commonly used file formats and enables users to allocate and organize their data in a simple way.

What PyCorrFit can do

- Load correlation curves from numerous correlators
- Process these curves (Section 3.2)
- Fit a model function (many included) to an experimental curve
- Import user defined models for fitting
- Many batch processing features
- Save/load entire *PyCorrFit* sessions

What PyCorrFit is not

- A multiple-τ correlator
- A software to operate hardware correlators

1.2 System prerequisites

1.2.1 Hardware

This documentation addresses the processing of correlation curves with *PyCorrFit*. *PyCorrFit* was successfully used with the following setups:

1. APD: Photon Counting Device from PerkinElmer Optoelectronics, Model: `SPCM-CD3017`
   Correlator: Flex02-01D/C from correlator.com with the shipped software `flex02-1dc.exe`.
2. APD: Photon Counting Device from PerkinElmer Optoelectronics
   Correlator: ALV-6000
3. LSM Confocor2 or Confocor3 setups from Zeiss, Germany.

[1] http://www.biochem.mpg.de/en/rd/schwille/
[2] https://github.com/paulmueller/PyCorrFit
[3] http://www.gnu.org/licenses/gpl.html

1.2.2 Software

The latest version of *PyCorrFit* can be obtained from the internet at http://pycorrfit.craban.de.

- **MacOSx**. Binary files for MacOSx >10.6.8 are available from the download page but have not yet been fully tested for stability.
- **Windows**. For Windows XP or Windows 7, stand-alone binary executables are available from the download page.
- **Linux**. There are executable binaries for widely used distributions (e.g. Ubuntu).
- **Sources** The program was written in Python, keeping the concept of cross-platform programming in mind. To run *PyCorrFit* on any other operating system, the installation of Python v.2.7 is required. To obtain the latest source, visit *PyCorrFit* at GitHub (https://github.com/paulmueller/PyCorrFit). *PyCorrFit* depends on the following python modules:

  ```
  python-matplotlib (≥ 1.0.1)
  python-numpy (≥ 1.5.1)
  python-scipy (≥ 0.8.0)
  python-sympy (≥ 0.7.2)
  python-yaml
  python-wxtools
  python-wxgtk2.8-dbg
  ```

  For older versions of Ubuntu, some of the above package versions are not listed in the package repository. To enable the use of *PyCorrFit* on those systems, the following tasks have to be performed:

  matplotlib. The tukss-ppa includes version 1.0.1. After adding the repository (`apt-add-repository ppa:tukss/ppa`), matplotlib can be installed as usual.

  numpy. The package from a later version of Ubuntu can be installed: https://launchpad.net/ubuntu/+source/python-numpy/

  scipy. The package from a later version of Ubuntu can be installed: https://launchpad.net/ubuntu/+source/python-scipy/

  sympy. To enable importing external model functions, sympy is required. It is available from http://code.google.com/p/sympy/downloads/list. Unpacking the archive and executing `python setup.py install` within the unpacked directory will install sympy.

Alternatively `python-pip` (http://pypi.python.org/pypi/pip) can be used to install up-to-date python modules.

LaTeX. *PyCorrFit* can save correlation curves as images using matplotlib. It is also possible to utilize Latex to generate these plots. On Windows, installing MiKTeX with "automatic package download" will enable this feature. On MacOSx, the MacTeX distribution can be used. On other systems, the packages LaTeX, dvipng, Ghostscript and the scientific latex packages `texlive-science` and `texlive-math-extra` need to be installed.

1.3 Running PyCorrFit

Windows Download the executable file and double-click on the `PyCorrFit.exe` icon.

Linux/Ubuntu Make sure the binary has the executable bit set, then simply double-click on the binary `PyCorrFit`.

Mac OSx When downloading the archive `PyCorrFit.zip`, the binary should be extracted automatically (if not, extract the archive) and you can double-click it to run *PyCorrFit*.

from source Invoke `python PyCorrFit.py` from the command line.

2 Working with PyCorrFit

2.1 Workflow

The following chapter introduces the general idea of how to start and accomplish a fitting project. FCS experiments produce different sets of experimental correlation functions which must be interpreted with appropriate physical models. Each correlation function refers to a single contiguous signal trace or "run". In *PyCorrFit*, the user must assign a mathematical model function to each correlation function during the loading procedure. The assignment is irreversible in the sense that within an existing *PyCorrFit* session it cannot be changed. This feature assures the stability of the batch processing routine for automated fitting of large data sets. Nevertheless, the fit of different models to the same data can be explored by loading the data twice or simply by creating two different sessions.

Let's briefly discuss a typical example: To determine the diffusion coefficient of a fluorescently labeled protein in free solution, one has to deal with two sets of autocorrelation data: measurements of a diffusion standard (e.g. free dye for which a diffusion coefficient has been published) to calibrate the detection volume and measurements of the protein sample. The protein sample may contain small amounts of slowly diffusing aggregates. While the calibration measurements can be fitted with a one-component diffusion model (T-3D), the protein sample displays two mobility states, monomers and aggregates, which are taken into account by a two-component diffusion model (T-3D-3D). With *PyCorrFit* such a situation can be treated in three ways, having different pros and cons:

1. Create separate sessions for each type of sample and assign different model functions.
2. Assign a one-component model to the dye measurements and a two-component model to the protein measurements when loading consecutively into the same session.
3. Assign a two-component model for all data and, when appropriate, manually inactivate one component by fixing its contribution to 0%.

The first approach is straightforward, however, it requires homogeneous diffusion behavior for each data set. The second strategy has the advantage that the dye and the protein curves, as well as the obtained parameters can be visually compared during the fitting analysis within the same session. In this case, batch fitting is still possible because it discriminates data sets assigned to different models. In the third case, simultaneous batch fitting is also possible. However, for each dye measurement one has to eliminate the second, slow diffusion species manually, which might be laborious. Inactivating components by fixing parameters

is nevertheless a common way to evaluate heterogeneous data sets, for example, a protein sample for which only a subgroup of curves requires a second diffusion component due to occasional appearance of aggregates. Such situations are frequently encountered in intracellular measurements. In conclusion, all three strategies or combinations thereof may be suitable. In any case, the user must decide on model functions beforehand, therefore it is advisable to group the data accordingly.

The fitting itself is usually explored with a representative data set. Here, the user has to decide on starting parameters, the range in which they should be varied, corrections like background, and other fitting options. Once the fit looks good, the chosen settings can be transferred at once to all other pages assigned to the same model using the *Batch control* tool (Section 3.2.3). After flipping through the data for visual inspection one may check the parameters across all pages in the *Statistics view* tool and re-visit outliers (Section 3.2.7). From there, the numerical fit values and example correlation functions can be exported.

2.2 The main window

Together with a system's terminal of the platform on which PyCorrFit was installed (Windows, Linux, MacOS), the *main window* opens when starting the program as described in section 1.3. The window title bar contains the version of *PyCorrFit* and, if a session was re-opened or saved, the name of the fitting session. A menu bar provides access to many supporting tools and additional information as thoroughly described in Chapter 3.

There are three gateways for experimental data into a pre-existing or a new *PyCorrFit* session (*File / Load data*, *File / Open session*, and *Current page / Import data*). When a session has been opened or correlation data have been loaded, each correlation curve is displayed on a separate page of a notebook. For quick identification of the active data set, a tab specifies the page number, the correlated channels (AC/CC), and the run number in case there are multiple runs in one experimental data file. When clicking a little triangle to the far-right, one can use a drop-down list of all page titles to directly access a particular data set. Alternatively, the pages can be toggled by tapping the curser keys (left/right). There can be only one activated page for which the tab appears highlighted.

The page containing a correlation function is divided in two halves. At the left hand side the page shows a pile of boxes containing values or fitting options associated to the current model and data set:

- *Data set*, a unique identifier for each correlation curve which is automatically assembled from different fields during the loading procedure (Section 3.1.2). This window can also be manually edited, thereby allowing to re-name or flag certain data during the fitting analysis.

- *Model parameters* displays the values which determine the current shape of the assigned model function. Initially, starting values are loaded as they were defined in the model description (Section 3.1.1). Little buttons allow a stepwise increase or decrease in units of $1/10^{\text{th}}$. It is also possible to directly enter some numbers. A checkbox is used to set the parameter status to "varied" (checked) or "fixed" (unchecked) during the fitting. At the end, when saving the session, the current set of values together with their indicated names are stored in the *.yaml file (Section 3.1.6).

- *Amplitude corrections* applies additional rescaling to amplitude related parameters like the number of particles n or fractions thereof associated with different correlation times (n_1, n_2, etc.). Experimental values of non-correlated background intensity can

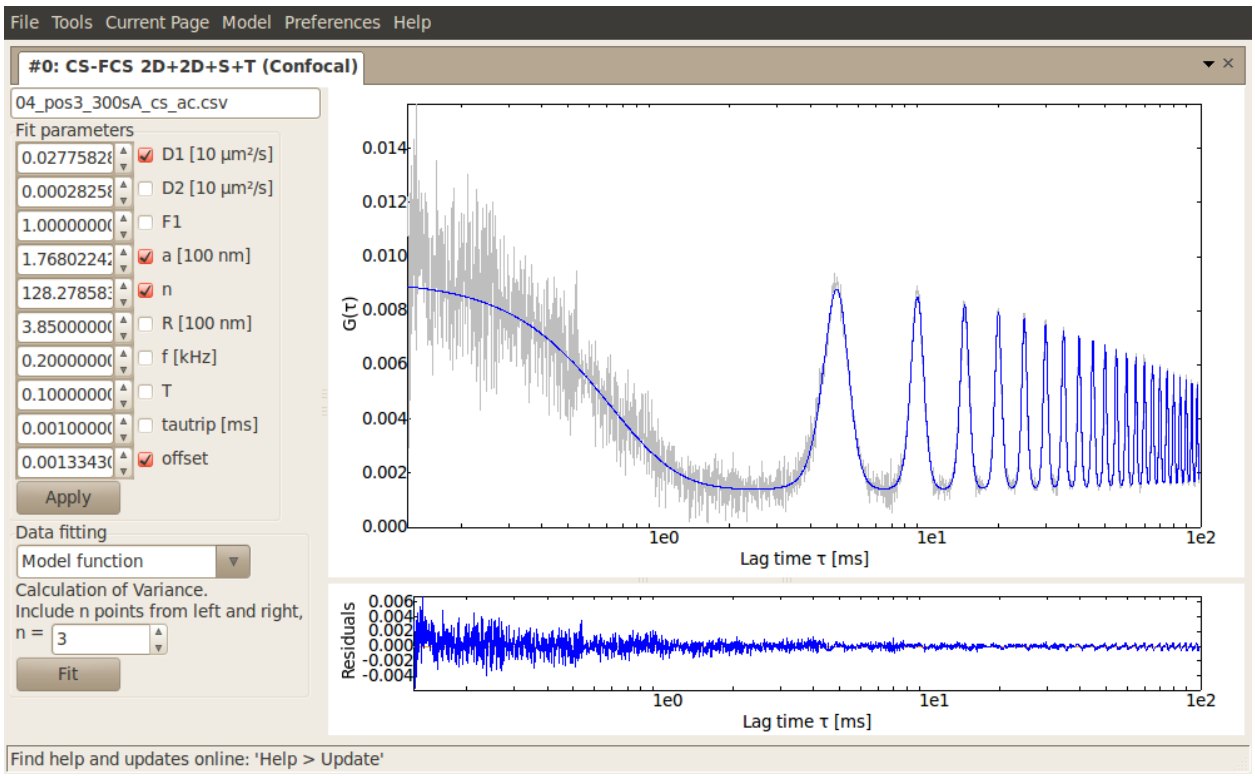

Figure 1, user interface of PyCorrFit: A circular scanning FCS (CS-FCS) curve of DiO on a supported lipid bilayer (glass substrate) is shown. The measurement yields a diffusion coefficient of 0.28 $\mu m^2 s^{-1}$ ($F1 = 1$, so only one component is fitted). Note that a 2D diffusion model is used and not a 3D model (as shown in figure 2).

be manually entered for each channel. In addition, the correlation curves can be normalized, to facilitate a visual comparison of their time dependence.

- *Fitting options* offers weighted fitting. The underlying idea is that data points with higher accuracy should also have a higher impact on model parameters. To derive weights, *PyCorrFit* calculates the variance of the difference between the actual data and a smooth, empiric representation of the curve for a certain neighborhood. The number of neighboring data points at each side ($j > 0$) can be set. For such a smooth representation a 5-knot spline function or the model function with the current parameter set can be used. The latter should improve when repeatedly fitting.

At the right hand side are two graphics windows. The dimensionless correlation functions $G(\tau)$ are plotted against the lag time (τ) in logarithmic scale. Below, a second window shows the residuals, the actual numerical difference between the correlation data and the model function. Fitting with appropriate models will scatter the residuals symmetrically around zero (x-axis). When weighted fitting was performed, the weighted residuals are shown. A good fit will not leave residuals systematically above or below the x-axis at any time scale.

The main window can be rescaled as a whole to improve data representation. In addition, to zoom in, one can drag a rectangle within the plot area; a double click then restores the initial scale. Experimental data points are linked by grey lines, the state of the model function is shown in blue. When a weighted fit was applied, the variance of the fit is calculated for each data point and displayed in cyan.

3 The menu bar

PyCorrFit is organized in panels which group certain functions. The menu organizes data management (File), data analysis (Tools), display of correlation functions (Current Page), numerical examples (Model), software settings (Preferences), and software metadata (Help).

3.1 File menu

The File menu organizes the import of theoretical models, experimental correlation data, and opening and saving of entire *PyCorrFit* fitting sessions. However, the numerical fit results are exported from the *Statistics view* panel which can be found under *Tools* (Section 3.2.7).

3.1.1 File / Import model

Correlation data must be fitted to models describing the underlying physical processes which give rise to a particular time dependence and magnitude of the recorded signal fluctuations. Models are mathematical expressions containing parameters with physical meaning, like the molecular brightness or the dwell time through an illuminated volume etc. While a number of standard functions are built-in, the user can define new expressions. Some examples can be found at GitHub in the *PyCorrFit* repository, e.g. circular scanning FCS [9] or a combination of diffusion and directed flow [2].

Model functions are imported as text files (*.txt) using certain syntax:

- **Encoding**: PyCorrFit can interpret the standard Unicode character set (UTF-8).
- **Comments**: Lines starting with a hash (`#`), empty lines, or lines containing only white space characters are ignored. The only exception is the first line starting with a hash followed by a white space and a short name of the model. This line is evaluated to complement the list of models in the dialogue *Choose model*, when loading the data.
- **Units**: PyCorrFit works with internal units for:
  - Time: 1 ms
  - Distance: 100 nm
  - Diffusion coefficient: $10\,\mu\mathrm{m}^2\mathrm{s}^{-1}$
  - Inverse time: $1000\,\mathrm{s}^{-1}$
  - Inverse area: $100\,\mu\mathrm{m}^{-2}$
  - Inverse volume: $1000\,\mu\mathrm{m}^{-3}$
- **Parameters:** To define a new model function new parameters can be introduced. Parameters are defined by a sequence of strings separated by white spaces containing name, the dimension in angular brackets, the equal sign, and a starting value which appears in the main window for fitting. For example: `D [10 µm²s⁻¹] = 5.0`. The parameter names contain only alphabetic (not numerical) characters. `G` and `g`, as well as the numbers `e` and `pi` are already mapped and cannot be used freely.
- **Placeholder:** When defining composite mathematical expressions for correlation functions one can use placeholders. Placeholders start with a lowercase 'g'. For example, the standard, Gaussian 3D diffusion in free solution may be written as
  - `gTrp = 1+ T/(1-T)*exp(-tau/tautrip)`

- `gTwoD = 1/(1+tau/taudiff)`
- `gThrD = 1/sqrt(1+tau/(taudiff*S**2))`

The individual parts are then combined in the last line of the *.txt file, where the correlation function is defined starting with uppercase 'G':

```
G = 1/n * gTrp * gTwoD * gThrD
```

For reference of mathematical operators check for example www.tutorialspoint.com / python / python_basic_operators.htm. To illustrate a more complex example see the model function for circular scanning FCS in figure 2.

3.1.2 File / Load data

Load data is the first way to import multiple correlation data sets into a *PyCorrFit* session. The supported file formats can be found in a drop-down list of supported file endings in the pop-up dialog *Open data files*:

(1) All supported files	default
(2) Confocor3 (*.fcs)	AIM 4.2, ZEN 2010, Zeiss, Germany
(3) Correlator ALV6000 (*.ASC)	ALV Laser GmbH, Langen, Germany
(4) Correlator.com (*.SIN)	www.correlator.com, USA
(5) Matlab 'Ries (*.mat)	EMBL Heidelberg, Germany
(6) PyCorrFit (*.csv)	Paul Müller, TU Dresden, Germany
(7) Zip files (*.zip)	Paul Müller, TU Dresden, Germany

While (2)-(4) are file formats associated with commercial hardware, (5) refers to a MATLAB based FCS evaluation software developed by Jonas Ries in the Schwille lab at TU Dresden, (6) is the txt-file containing comma-separated values (csv) generated with PyCorrFit via the command *Current Page / Save data*. Zip-files are automatically decompressed and can be imported when matching one of the above mentioned formats. In particular loading of zip files is a possibility to re-import correlation data from entire *PyCorrFit* sessions. However, these data are treated as raw, which means that all fitting parameters and model assignments are lost.

When loading data, the user is prompted to assign fit models in the *Choose Models* dialogue window. There, curves are sorted according to channel (for example AC1, AC2, CC12, and CC21, as a typical outcome of a dual-color cross-correlation experiment). For each channel a fit model must be selected from the list (see Section 3.4):

If a file format is not yet listed, the correlation data could be converted into a compatible text-file (*.csv) or bundles of *.csv files within a compressed archive *.zip. For reformatting the following points should be considered:

- **Encoding**: *PyCorrFit* uses the standard Unicode character set (UTF-8). However, since no special characters are needed to save experimental data, other encodings may also work. New line characters are `\r\n` (Windows).
- **Comments**: Lines starting with a hash (`#`), empty lines, or lines containing only white space characters are ignored. Exceptions are the keywords listed below.
- **Units**: PyCorrFit works with units/values for:

```
# CS-FCS 3D+S+T (Confocal)

# Circular Scanning FCS model function. 3D diffusion + Triplet.

## Definition of parameters:
# First, the parameters and their starting values for the model function
# need to be defined. If the parameter has a unit of measurement, then it
# may be added separated by a white space before the "=" sign. The starting
# value should be a floating point number. Floating point abbreviations
# like "1e-3" instead of "0.001" may be used.

# Diffusion coefficient
D [10 μm²/s] = 200.0
# Structural parameter
w = 5.0
# Waist of the lateral detection area
a [100 nm] = 1.0
# Particle number
n = 5.0
# Scan radius
R [100 nm] = 5.0
# Frequency
f [kHz] = 20.0
# Triplet fraction
T = 0.1
# Triplet time
tautrip [ms] = 0.001

# The user may wish to substitute certain parts of the correlation function
# with other values to keep the formula simple. This can be done by using the
# prefix "g". All common mathematical functions, such as "sqrt()" or "exp()"
# may be used. For convenience, "pi" and "e" are available as well.

gTrip = 1. + T/(1-T)*exp(-tau/tautrip)
gScan = exp(-(R*sin(pi*f*tau))**2/(a**2+D*tau))
gTwoD = 1./(1.+D*tau/a**2)
gOneD = 1./sqrt(1.+D*tau/(w*a)**2)
gThrD = gTwoD * gOneD

# The final line with the correlation function should start with a "G"
# before the "=" sign.

G = 1./n * gThrD * gScan * gTrip
```

Figure 2, user defined model function for PyCorrFit: The working example shows a model function for circular scanning FCS.

- Time: 1 ms
- Intensity: 1 kHz
- Amplitude offset: $G(0) = 0$ (not 1)

- **Keywords:**[4] *PyCorrFit* reads the first two columns containing numerical values. The first table (non-hashed) is recognized as the correlation data containing the lag times in the first and the correlation data in the second column. (In case the *.csv file has been generated with *PyCorrFit* up to three additional columns containing the fit function are ignored). The table ends, when the keyword `# BEGIN TRACE` appears. Below this line the time and the signal values should be contained in the first two columns. If cross-correlation data have to be imported a second trace can be entered after the keyword `# BEGIN SECOND TRACE`.

- **Tags:**[5] Channel information can be entered using defined syntax in a header. The keyword

  `# Type AC/CC Autocorrelation`

  assigns the tag `AC` and the keyword

  `# Type AC/CC Crosscorrelation`

  assigns the tag `CC` to the correlation curve. These strings are consistently displayed in the user interface of the respective data page in *PyCorrFit*. If no data type is specified, autocorrelation is assumed. Tags may be specified with additional information like channel numbers, e.g.

  `# Type AC/CC Autocorrelation _01.`

  In this case the tag `AC_01` is generated. This feature is useful to keep track of the type of curve during the fitting and when post-processing the numerical fit results.

3.1.3 File / Open session

This command is the second way to import data into PyCorrFit. In contrast to *Load data*, it opens an entire fitting project, which was previously saved with *PyCorrFit*. Sessions are bundles of files named *.fcsfit-session.zip. Sessions contain, comments, model assigned correlation data, and the current state of parameters for each data page (Section 3.1.6).

3.1.4 File / Comment session

This command opens a window to place text messages that can be used to annotate a fitting session.

3.1.5 File / Clear session

This command closes all pages while the PyCorrFit.exe keeps running. The user is prompted to save the session under the same or a different name. At this stage both options *No* or *Cancel* lead to clearance and a potential loss of recent modifications.

[4] Keywords are case-insensitive.
[5] Tags are case-insensitive.

3.1.6 File / Save session

In addition to display and fit individual curves, a strong feature of PyCorrFit is to save an entire fitting project as a single session. Sessions allow the user to revisit and explore different models, fitting strategies, and data sets. Importantly the work can be saved at any stage.

The number of files bundled in a session varies depending on the number of data sets (pages), the number of used models, and what was done during the fitting. A detailed description can be found in the Readme.txt file attached to each session. For example, the numerical correlation and intensity data are saved separately as *.csv text files. However, in contrast to the *Save data (*.csv)* command of the *Current Page* menu, there are no metadata in the header, just tables containing the numerical values. In sessions, the fitting parameters are stored separately in the human-readable data serialization format, *.yaml.

3.1.7 File / Exit

This command closes down *PyCorrFit*. The user is prompted to save the session under the same or a different name. At this stage *No* leads to the loss of recent changes, while *Cancel* keeps *PyCorrFit* running.

3.2 Tools menu

The *Tools* menu provides access to a series of accessory panels which extent the capability of the main window. These accessory panels can stay open during the entire analysis. Open panels appear checked in the menu. Most operations can be executed across the entire data set with a single mouse click.

3.2.1 Tools / Data range

This panel limits the range of lag times which are displayed in the main window panel. At the same time it defines the range of points which are used for fitting. For example, this feature can be applied to remove dominant after-pulsing of the avalanche photo diodes (APDs) which may interfere with Triplet blinking at short lag times. The user has the options to *Apply* the channel settings only to the current page or he can *Apply to all pages*. In contrast to *Batch control*, this operation ignores whether the data are assigned to different models.

Power user, who frequently load and remove data sets, may take advantage of a checkbox to fix the channel selection for all newly loaded data sets.

3.2.2 Tools / Overlay curves

This window displays the correlation data (not the fit curves) of all pages in a single plot. The curves can be discriminated by color. If only one curve is selected it appears in red. Curves with ambiguous shape can easily be identified, selected, and removed by clicking *Apply*. A warning dialogue lists the pages which will be kept.

Data representation is synchronized with the page display in the *Main window*. For example, narrowing the range of lag times by *Data range* is immediately updated in the *Overlay curves* tool. Likewise, their normalization of the amplitudes to unity.

The other way round, some tools directly respond to the selections made in the *Overlay curves* tool: *Global fitting*, *Average curves*, and *Statistics view* allow to perform operations on an arbitrary selection of pages which can be specified by page number. Instead of manually

typing their numbers, the curves may be selected within the *Overlay curves* tool. The respective input fields are immediately updated.

The tool is closed by the button *Cancel*. All the listed data sets will be kept. However, the selections transferred to the *Global fitting*, *Average curves*, and *Statistics view* tools are kept as well.

3.2.3 Tools / Batch control

By default the current page is taken as a reference to perform automated fitting. A batch is defined as the ensemble of correlation data sets (pages) assigned to the same model function within a session. A session can therefore have several batches, even for the same data.

For fitting it is crucial to carefully define the starting parameters, whether parameters should be fixed or varied, the range of values which make physically sense, and other options offered within the *Main window*. By executing *Apply to applicable pages*, these settings are transferred to all other pages assigned to the same fit model. Note that this includes the range of lag times (lag time channels) which may have been changed with the *Data range* tool for individual pages.

The button *Fit applicable pages* then performs several cycles of fitting [how many cycles?] on all pages of the same batch. Alternatively, the user can define an external source of parameters as a reference, i.e. the first page of some *Other session* (*.fcsfit-session.zip). However, this assumes a consistent assignment of model functions.

3.2.4 Tools / Global fitting

Global fitting is useful when experimental curves share the same values for certain physical parameters. For example, due to physical constraints in two-focus FCS both autocorrelation curves and the cross-correlation curves should adopt the same values for the diffusion time *taudiff* and the number of particles *n*. A global fit can be applied such that *n* and *taudiff* are identical for all data sets. All curves are added to a single array. In contrast to fixing the shared parameters across a batch, in *Global fitting* Chi-square is minimized for all data sets simultaneously [please check!]. To perform *Global fitting*, a subset of curves has to be selected by typing the numbers into the input field or by highlighting the pages via the *Overlay* tool.

3.2.5 Tools / Average data

Often in FCS, the measurement time at a particular spot is divided in several runs. This approach is taken when occasional, global intensity changes are superimposed on the molecular fluctuations of interest. Then the user has to sort out the bad runs. After fitting, one may want to re-combine the data, to export a cleaned, average correlation function. This can be done with the tool *Average data*, for which a subset of curves has to be selected by typing the numbers into the input field or by highlighting the pages via the *Overlay curves* tool.

For averaging, there are constraints:

1. Since the correlation curves are averaged point by point this requires the same number of lag time channels. Runs of different length cannot be averaged.

2. The tool can only average data sets which are exclusively autocorrelation or cross-correlation.

3. The user can check a box to enforce the program to ask for data sets with the same model as the current page. This may help to avoid mistakes when selecting pages.

The averaged curve is shown on a separate page. The new *Filename/title* receives the entry *Average [numbers of pages]*. The assigned model is by default the same as for the individual pages. However, while averaging, the user can choose a different model from a drop-down list.

3.2.6 Tools / Trace view

FCS theory makes assumptions about the thermodynamic state of the system. Signal fluctuations can only be analyzed when the system is at equilibrium or at a sufficiently stable steady state. Global instabilities on the time scale of the measurement itself, e.g. photobleaching, have dramatic effect on the shape of the measured correlation curve. Therefore it is common practice to check the correlated intensity trace for each curve. Trace view simply displays the signal trace for each correlation function. The window stays open during the session and can be used to revisit and flag ambiguous data sets.

3.2.7 Tools / Statistics view

The goal of a correlation analysis is to determine experimental parameter values with sufficient statistical significance. However, especially for large data sets, it can get quite laborious to check all of the individual values on each page. We designed the *Statistics view* panel to review the state of parameters across the experimental batch (pages assigned to the same model) in a single plot, thereby facilitating to the identification of outliers.

The current page is taken as a reference for the type of model parameters which can be displayed. The user can choose different *Plot parameters* from a drop-down list. A subset of pages within the batch can be explicitly defined by typing the page numbers into the input field or by highlighting in the *Overlay curves* tool. Note that page numbers which refer to different models than the current page are ignored.

The *Statistics view* panel contains a separate *Export* box, where parameters can be selected (checked) and saved as a comma separated text file (*.csv). Only selected page numbers are included.

3.2.8 Tools / Page info

Page info is a most verbose summary of a data set. The panel *Page info* is synchronized with the current page. The following fields are listed:

1. Version of PyCorrFit
2. Field values from the main window (filename/title, model specifications, page number, type of correlation, normalizations)
3. Actual parameter values (as contained in the model function)
4. Supplementary parameters (intensity, counts per particle, duration, etc.)
5. Fitting related information (Chi-square, channel selection, varied fit parameters) .
6. Model doc string (Section 3.4)

The content of Page info is saved as a header when exporting correlation functions via the command *Current page / Save data (*.csv)* (Section 3.3.2).

3.2.9 Tools / Slider simulation

This tool visualizes the impact of model parameters on the shape of the model function of a current page. Such insight may be useful to choose proper starting values for fitting or to develop new model functions. For example, in the case two of the parameters trade during the fitting one may explore to which extent a change in both values produces similar trends.

Two variables (A and B) have to be assigned from a drop-down list of parameters associated with the current model function. For each of these, the *Slider simulation* panel shows initially the starting value (x) as a middle position of a certain range (from 0.1*x to 1.9*x). The accessible range can be manually edited and the actual value of the slider position is displayed at the right hand side of the panel. Dragging the slider to lower (left) or higher (right) values changes the entry in the box *Model parameters* of the *Main window* and accordingly the shape opt the model function in the plot. By default the checkbox *Vary A and B* is active meaning that both variables during *Slider simulation* can be varied independently.

In addition, the variables A and B can be linked by a mathematical relation. For this a mathematical operator can be selected from a small list and the option *Fix relation* must be checked. Then, the variable B appears inactivated (greyed out) and the new variable combining values for A and B can be explored by dragging.

3.3 Current Page

This menu compiles import and export operations referring exclusively to the active page in the main window.

3.3.1 Current Page / Import Data

This command is the third way to import data into a pre-existing session. Single files containing correlation data can be imported as long as they have the right format (Section 3.1.2). In contrast to *Load data* from the *File* menu, the model assignment and the state of the parameters remains. The purpose of this command is to compare different data sets to the very same model function for a given parameter values. After successful import, the previous correlation data of this page are lost.

To avoid this loss, one could first generate a new page via the menu (Section **??**), select a model function and import data there. This is also a possibility to assign the very same data to different models within the same session.

3.3.2 Current Page / Save data (*.csv)

For the documentation with graphics software of choice, correlation curves can be exported as a comma-separated table. A saved *PyCorrFit* text-file (*.csv) will contain a hashed header with metadata from the *Page info* tool (Section 3.2.8), followed by the correlation and fitting values in tab-separated columns: *Channel (tau [s])*, *Experimental correlation*, *Fitted correlation*, *Residuals*, and *Weights (fit)*.

Below the columns, there are again 5 rows of hashed comments followed by the intensity data in two columns: *Time [s]* and *Intensity trace [kHz]*. Note that there are no assemblies of "multiple runs", since *PyCorrFit* treats these as individual correlation functions. A *.csv file therefore contains only a single fitted correlation curve and one intensity trace for autocorrelation or two intensity traces for cross-correlation.

3.3.3 Current Page / Save correlation as image

For a quick documentation, the correlation curve can be exported as a compressed bitmap (*.png). The plot contains a legend and the actual values and errors of the varied parameters, however, not the fixed parameters. Note that the variable tau cannot be displayed using Unicode with Windows.

3.3.4 Current Page / Save trace view as image

For a quick documentation the intensity from the *Trace view* panel can be exported as a compressed bitmap (*.png).

3.3.5 Current Page / Close page

Closes the page; the data set is removed from the session. The page numbers of all other pages remain the same. The command is equivalent with the closer (x) in the tab.

3.4 Models

When choosing a model from the *Models* menu a new page opens and the model function is plotted according to the set of starting values for parameters as they were defined in the model description. The lists contains all of the implemented model functions, which can be selected during *File / Load data*. The parameters can be manipulated to explore different shapes; the tool *Slider simulation* can also be used. Via *Current page / Import data*, the model may then be fitted to an experimental data set. Standard model functions for a confocal setup are:

- Confocal (Gaussian): T+3D Triplet blinking and 3D diffusion
- Confocal (Gaussian): T+3D+3D Triplet with two diffusive components
- Confocal (Gaussian): T-2D Triplet blinking and 2D diffusion
- Confocal (Gaussian): T-2D-2D Triplet with two diffusive components
- Confocal (Gaussian): T-3D-2D Triplet with mixed 3D and 2D diffusion

There is also a collection of models for FCS setups with TIR excitation:

- TIR (Gaussian/Exp.): 3D 3D diffusion
- TIR (Gaussian/Exp.): T+3D+3D Triplet with two diffusive components
- TIR (Gaussian/Exp.): T+3D+2D Triplet with mixed 3D and 2D diffusion

In addition, there are may be user defined model functions which have been uploaded previously via File / Import model (Section 3.1.1).

3.5 Preferences

Latex If the user has a Tex distribution (e.g. MikTex for Windows) installed, checking the "Latex" option will open a separate, TeX formatted panel (*Figure1*) via the *Current page / Save [...] as image* commands. The *Figure1* contains some interactive options for display. From there, in a second step, the image can be exported as *.png or *.svg.

Verbose If checked, this will cause the *PyCorrFit* to display graphs that would be hidden otherwise. In weighted fitting with a spline, the spline function used for calculating the weights for each data points is displayed[6]. When saving the correlation curve as an image (Section 3.3.3), the plot will be displayed instead of saved. If "Latex" is active these plots will also be TeX-formatted. The advantage in displaying plots is the ability to zoom or rescale the plot from within *PyCorrFit*.

Show weights Checking the option *Show weights* will produce two lines showing the weights for each data point of the correlation function in the plot, as well as in the exported image. Note that the weights are always exported when using the *Save data (*.csv)* command from the *Current page* menu.

3.6 Help

Documentation This entry displays this documentation using the systems default PDF viewer.

Wiki This entry displays the wiki of *PyCorrFit* on *GitHub*. Everyone who registers with *GitHub* will be able to make additions and modifications. The wiki is intended for end-users of *PyCorrFit* to share protocols or to add other useful information.

Update establishes a link to the GitHub website to check for a new release; it also provides a few web links associated with PyCorrFit

Shell This gives Shell-access to the functions of *PyCorrFit*. It is particularly useful for trouble-shooting.

Software This lists the exact version of *Python* and the corresponding modules with which PyCorrFit is currently running.

About Information of the participating developers, the license, and documentation writers.

4 4 Hacker's corner

New internal model functions Additionally, new file formats can be implemented by programming of the readfiles module of *PyCorrFit*. First, edit the code for `__init__.py` and then add the script `read_FileFormat.py`.

External models will be imported with internal model function IDs starting at 7000. Models are checked upon import by the Python module sympy. If the import fails it might be a syntax error or just an error of sympy, since this module is still under development.

[6]For obvious reasons, such a plot is not generated when using the iteratively improved *Model function* or the actual *Average* correlation curve for weighted fitting.

5 Theoretical background

5.1 Derivation of FCS model functions

This section introduces the calculation of FCS model functions. It supplies some background information and points out general properties of correlation functions.

5.1.1 General Autocorrelation function for a single species

FCS model functions describe how the signal $F(t)$, emitted from a certain observation volume, is temporally dependent on its own past (autocorrelation) or on some other signal (cross-correlation). The autocorrelation $G(\tau)$ of a signal $F(t)$ is computed as follows:

$$G(\tau) = \frac{\langle \delta F(t) \delta F(t+\tau) \rangle}{\langle F(t) \rangle^2} = \frac{g(\tau)}{\langle F(t) \rangle^2}. \tag{1}$$

$G(\tau)$ normalized autocorrelation curve

τ lag time

$\langle F \rangle$ the expectation value of $F(t)$. Applying the ergodic theorem, this can be rewritten as the time average

$$\langle F(t) \rangle = \lim_{T \to \text{inf}} \frac{1}{T} \int_0^T F(t) \mathrm{d}t.$$

$\delta F(t)$ $= F(t) - \langle F(t) \rangle$ fluctuation of the fluorescence signal

$g(\tau)$ non normalized autocorrelation curve

The fluorescence signal is dependent on the size and shape of the detection volume (e.g. Gaussian shaped for confocal setups or exponential decaying for TIRF setups), on the propagator of the diffusing dye (free diffusion, diffusion with flow, etc.), and the brightness and concentration of the dye under observation [3].

$$G(\tau) = \frac{q^2 C \int \mathrm{d}^3 r \int \mathrm{d}^3 r' \, \Omega(\mathbf{r}) \Phi(\mathbf{r}, \mathbf{r}', \tau) \Omega(\mathbf{r}')}{\langle F(t) \rangle^2} \tag{2}$$

q molecular brightness, dependent on excitation intensity, quantum yield, i.e. emission properties and absorption cross sections of the dye, and the detection efficiency of the instrument.

Ω 3D molecule detection function, dependent on the shape of the pinholes used for detection and the excitation laser profile, i.e. the point spread function (PSF).

Φ diffusion propagator. The distribution of dyes in a liquid follows Fick's laws of diffusion. For free diffusion, this is a simple Gaussian distribution.

F fluorescence signal of the sample. It is defined as

$$F(t) = q \int \mathrm{d}^3 r \, \Omega(\mathbf{r}) c(\mathbf{r}, t)$$

with $c(\mathbf{r}, t)$ being dye distribution (particle concentration) inside the detection volume.

C average concentration of the dye following the dynamics of the propagator Φ. Using the ergodic hypothesis and assuming a normalized molecule detection function ($V_{\text{eff}} = \int d^3 r \, \Omega(\mathbf{r}) = 1$), the concentration computes to $C = \langle F(t) \rangle / q$.

5.1.2 General Autocorrelation function for multiple species

Most experiments include particles with more than one dynamic property. Labeled particles may have different size or the temporal dynamics may include a triplet term. For n different species inside the detection volume, the autocorrelation function becomes:

$$G(\tau) = \frac{g(\tau)}{\langle F(t) \rangle^2} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} g_{ij}(\tau)}{\langle F(t) \rangle^2} \tag{3}$$

$$g_{ij}(\tau) = q_i q_j \int \mathrm{d}^3 r \int \mathrm{d}^3 r' \, \Omega(\mathbf{r}) \Phi_{ij}(\mathbf{r}, \mathbf{r}', \tau) \Omega(\mathbf{r}') \tag{4}$$

$g(\tau)$ non normalized correlation function

$g_{ij}(\tau)$ non normalized cross correlation between two species i and j. For n species, $i, j \in [1, ..., n]$.

q_i molecular brightness of species i

Ω 3D molecule detection function

Φ_{ij} diffusion propagator computed from species i with species j. If species i and j are independently diffusing, then Φ_{ij} is zero. $C_{ij} \Phi_{ij}(\mathbf{r}, \mathbf{r}', \tau) = \langle \delta c_i(\mathbf{r}, 0) \delta c_j(\mathbf{r}', \tau) \rangle$

C_{ij} average concentration of objects following the dynamics of Φ_{ij}. If $i = j$, $C_{ii} = C_i$ is the concentration of the dye i.

If the propagators $\Phi_{ij}(x, y, z; x', y', z'; \tau)$ and the molecule detection function $\Omega(x, y, z)$ factorize into an axial (z) and a lateral (x, y) part, so will $g_{ij}(\tau)$:

$$g_{ij}(\tau) = q_i q_j \cdot g_{ij,z}(\tau) \cdot g_{ij,xy}(\tau) \tag{5}$$

Following the example with a freely diffusing species A and a laterally diffusing species B inside a membrane at $z = z_0$, it can be concluded:

$$\begin{aligned} g_{AA}(\tau) = &\quad q_A^2 \cdot g_{AA,z}(\tau) \cdot g_{AA,xy}(\tau) \\ g_{BB}(\tau) = &\quad q_B^2 \cdot g_{BB,z_0}(\tau) \cdot g_{BB,xy}(\tau) \\ g_{AB}(\tau) = g_{BA}(\tau) = &\quad q_A q_B \cdot g_{AB,z}(\tau) \cdot g_{AB,xy}(\tau) \\ g(\tau) = &\quad g_{AA}(\tau) + 2g_{AB}(\tau) + g_{BB}(\tau) \end{aligned}$$

To obtain the normalized autocorrelation function, the average $\langle F(t)\rangle$ has to be calculated:

$$\begin{aligned} F(t) = &\quad \sum_{i=1}^{n} F_i(t) \\ F_A(t) = &\quad q_A \int \mathrm{d}^3 r \, \Omega(\mathbf{r}) C_A(\mathbf{r}, t) \\ F_B(t) = &\quad q_B \int \mathrm{d}x \int \mathrm{d}y \, \Omega(x, y, z = z_0) C_B(x, y, t) \\ \langle F(t)\rangle = &\quad \langle F_A(t)\rangle + \langle F_B(t)\rangle \end{aligned}$$

It is noticeable, that C_B is a 2D concentration, whereas C_A is a 3D concentration. Since there is no correlation between the two freely diffusing species A and B, $g_{AB}(\tau)$ is zero. The normalized autocorrelation curve may now be calculated like this:

$$\begin{aligned} G(\tau) = &\quad \frac{g(\tau)}{\langle F(t)\rangle^2} \\ G(\tau) = &\quad \frac{g_{AA}(\tau) + g_{BB}(\tau)}{(\langle F_A(t)\rangle + \langle F_B(t)\rangle)^2} \end{aligned}$$

5.1.3 Cross-correlation

Cross-correlation is a generalization of autocorrelation. Cross-correlation functions are derived in the same manner as autocorrelation functions. Here, signals recorded in two detection channels are cross-correlated to obtain the correlation function.

$$G_{XY}(\tau) = \frac{\langle \delta F_X(t) \delta F_Y(t+\tau)\rangle}{\langle F_X(t)\rangle \langle F_Y(t)\rangle} \tag{6}$$

A cross-correlation analysis of two species labeled by two types of dyes observed in two corresponding detection channels can be used for binding assays. Only complexes giving simultaneous signal in both channels contribute to the cross-correlation amplitude. Thus a finite cross-correlation indicates co-diffusion.

5.1.4 Extension of the theory

By modifying the propagator Φ and the detection volume Ω, other effects, like triplet blinking or binding reactions can be quantified. In many cases, analytical solutions to the above integrals are not straightforward and approximations have to be made. For example, the Gaussian shaped detection profile in confocal FCS is already an approximation. However, deviations from the true results are considered to be small [18]. Section 5.4 introduces several model functions with various detection symmetries and particle dynamics.

5.2 Non-linear least-squares fit

PyCorrFit uses the non-linear least-squares fitting capabilities from `scipy.optimize`. This package utilizes the Levenberg–Marquardt algorithm to minimize the sum of the squares. More information on this topic can be obtained from the online documentation of `leastsq`[7]. One can define a distance $d(G, H)$ between two discrete functions G and H with the discrete domain of definition $\tau_1 \dots \tau_n$ as the sum of squares:

$$d(G, H) = \sum_{i=1}^{n} [G(\tau_i) - H(\tau_i)]^2 \tag{7}$$

The least-squares method minimizes this distance between the model function G and the experimental values H by modifying k additional fitting parameters $\alpha_1, \dots, \alpha_k$:

$$\chi^2 = \min_{\alpha_1,\dots,\alpha_k} \sum_{i=1}^{n} [G(\tau_i, \alpha_1, \dots, \alpha_k) - H(\tau_i)]^2 \tag{8}$$

The minimum distance χ^2 is used to characterize the success of a fit. Note, that if the number of fitting parameters k becomes too large, multiple values for χ^2 can be found, depending on the starting values of the k parameters.

5.3 Weighted fitting

In certain cases, it is useful to implement weights (standard deviation) σ_i for the calculation of χ^2. For example, very noisy parts of a correlation curve can falsify the resulting fit. In PyCorrFit, weighting is implemented as follows:

$$\chi^2_{\text{weighted}} = \min_{\alpha_1,\dots,\alpha_k} \sum_{i=1}^{n} \frac{[G(\tau_i, \alpha_1, \dots, \alpha_k) - H(\tau_i)]^2}{\sigma_i^2} \tag{9}$$

PyCorrFit is able to calculate the weights σ_i from the experimental data. The different approaches of this calculation of weights implemented in PyCorrFit are explained in section 2.1.

5.4 Implemented model functions

This is an overview of all the model functions that are currently[8] implemented in PyCorrFit. To each model a unique model ID is assigned by PyCorrFit. Most of the following information is also accessible from within PyCorrFit using the **Page info** tool.

5.4.1 Confocal FCS

The confocal detection volume with the structural parameter

$$SP = \frac{z_0}{r_0} \tag{10}$$

has an effective size of

$$V = \pi^{3/2} r_0^2 z_0 \tag{11}$$

[7] http://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.leastsq.html#scipy.optimize.leastsq

[8] November 30, 2013

where r_0 is its lateral and z_0 its axial (in case of 3D diffusion) extension. Thus, the effective number of particles is defined as

$$N = CV \tag{12}$$

with the concentration C given implicitly in the model functions. The diffusion coefficient is calculated from the diffusion time τ_{diff} using

$$D = \frac{1}{4\tau_{\text{diff}}} \left(\frac{z_0}{SP}\right)^2 = \frac{r_0^2}{4\tau_{\text{diff}}}. \tag{13}$$

The parameters in the equation above need to be calibrated to obtain the diffusion coefficient. Usually a reference dye with a known diffusion coefficient is used to determine the lateral extension of the detection volume r_0 with a fixed structural parameter of e.g. $SP = 4$.

Name **2D (Gauß)**
ID **6001**
Descr. Two-dimensional diffusion with a Gaussian laser profile [1,10,12].

$$G(\tau) = A_0 + \frac{1}{N} \frac{1}{(1 + \tau/\tau_{\text{diff}})} \tag{14}$$

A_0	Offset
N	Effective number of particles in confocal area
τ_{diff}	Characteristic residence time in confocal area

Name **2D+T (Gauß)**
ID **6002**
Descr. Two-dimensional diffusion with a Gaussian laser profile, including a triplet component [1, 6, 10, 12, 15, 16].

$$G(\tau) = A_0 + \frac{1}{N} \frac{1}{(1 + \tau/\tau_{\text{diff}})} \left(1 + \frac{T e^{-\tau/\tau_{\text{trip}}}}{1 - T}\right) \tag{15}$$

A_0	Offset
N	Effective number of particles in confocal area
τ_{diff}	Characteristic residence time in confocal area
T	Fraction of particles in triplet (non-fluorescent) state
τ_{trip}	Characteristic residence time in triplet state

Name **3D (Gauß)**
ID **6012**
Descr. Three-dimensional free diffusion with a Gaussian laser profile (eliptical) [1, 10, 12].

$$G(\tau) = A_0 + \frac{1}{N} \frac{1}{(1 + \tau/\tau_{\text{diff}})} \frac{1}{\sqrt{1 + \tau/(SP^2\tau_{\text{diff}})}} \tag{16}$$

A_0	Offset
N	Effective number of particles in confocal volume
τ_{diff}	Characteristic residence time in confocal volume
SP	Structural parameter, describes elongation of the confocal volume

Name **3D+T (Gauß)**
ID **6011**
Descr. Three-dimensional free diffusion with a Gaussian laser profile (eliptical), including a triplet component [6, 15, 16].

$$G(\tau) = A_0 + \frac{1}{N}\frac{1}{(1+\tau/\tau_{\text{diff}})}\frac{1}{\sqrt{1+\tau/(SP^2\tau_{\text{diff}})}}\left(1+\frac{Te^{-\tau/\tau_{\text{trip}}}}{1-T}\right) \tag{17}$$

A_0	Offset
N	Effective number of particles in confocal volume
τ_{diff}	Characteristic residence time in confocal volume
SP	Structural parameter, describes elongation of the confocal volume
T	Fraction of particles in triplet (non-fluorescent) state
τ_{trip}	Characteristic residence time in triplet

Name **2D+2D+T (Gauß)**
ID **6031**
Descr. Two-component, two-dimensional diffusion with a Gaussian laser profile, including a triplet component [1, 4, 8, 14].

$$G(\tau) = A_0 + \frac{1}{N(F+\alpha(1-F))^2}\left[\frac{F}{1+\tau/\tau_1}+\alpha^2\frac{1-F}{1+\tau/\tau_2}\right]\left(1+\frac{Te^{-\tau/\tau_{\text{trip}}}}{1-T}\right) \tag{18}$$

A_0	Offset
N	Effective number of particles in confocal area ($N = N_1 + N_2$)
τ_1	Diffusion time of particle species 1
τ_2	Diffusion time of particle species 2
F	Fraction of molecules of species 1 ($N_1 = FN$)
α	Relative molecular brightness of particles 1 and 2 ($\alpha = q_2/q_1$)
T	Fraction of particles in triplet (non-fluorescent) state
τ_{trip}	Characteristic residence time in triplet state

Name **3D+2D+T (Gauß)**
ID **6032**
Descr. Two-component, two- and three-dimensional diffusion with a Gaussian laser profile, including a triplet component [1, 4, 8, 14].

$$G(\tau) = A_0 + \frac{1}{N(1-F+\alpha F)^2}\left[\frac{1-F}{1+\tau/\tau_{2\text{D}}}+\frac{\alpha^2 F}{(1+\tau/\tau_{3\text{D}})}\frac{1}{\sqrt{1+\tau/(SP^2\tau_{3\text{D}})}}\right]\left(1+\frac{Te^{-\tau/\tau_{\text{trip}}}}{1-T}\right) \tag{19}$$

A_0	Offset
N	Effective number of particles in confocal volume ($N = N_{2D} + N_{3D}$)
τ_{2D}	Diffusion time of surface bound particles
τ_{3D}	Diffusion time of freely diffusing particles
F	Fraction of molecules of the freely diffusing species ($N_{3D} = FN$)
α	Relative molecular brightness of particle species ($\alpha = q_{3D}/q_{2D}$)
SP	Structural parameter, describes elongation of the confocal volume
T	Fraction of particles in triplet (non-fluorescent) state
τ_{trip}	Characteristic residence time in triplet state

Name **3D+3D+T (Gauß)**
ID **6030**
Descr. Two-component three-dimensional free diffusion with a Gaussian laser profile, including a triplet component [1, 4, 8, 14].

$$G(\tau) = A_0 + \frac{1}{N(F+\alpha(1-F))^2}\left(1+\frac{Te^{-\tau/\tau_{trip}}}{1-T}\right)\times \times\left[\frac{F}{(1+\tau/\tau_1)}\frac{1}{\sqrt{1+\tau/(SP^2\tau_1)}}+\alpha^2\frac{1-F}{(1+\tau/\tau_2)}\frac{1}{\sqrt{1+\tau/(SP^2\tau_2)}}\right] \tag{20}$$

A_0	Offset
N	Effective number of particles in confocal volume ($N = N_1 + N_2$)
τ_1	Diffusion time of particle species 1
τ_2	Diffusion time of particle species 2
F	Fraction of molecules of species 1 ($N_1 = FN$)
α	Relative molecular brightness of particles 1 and 2 ($\alpha = q_2/q_1$)
SP	Structural parameter, describes elongation of the confocal volume
T	Fraction of particles in triplet (non-fluorescent) state
τ_{trip}	Characteristic residence time in triplet state

5.4.2 Confocal TIR-FCS

The detection volume is axially confined by an evanescent field and has an effective size of

$$V = \pi R_0^2 d_{eva} \tag{21}$$

where R_0 is the lateral extent of the detection volume and d_{eva} is the evanescent field depth[9]. From the concentration C, the effective number of particles is $N = CV$. The decay constant κ is the inverse of the depth d_{eva} :

$$d_{eva} = \frac{1}{\kappa} \tag{22}$$

[9]Where the field has decayed to $1/e$

The model functions make use of the Faddeeva function (complex error function)[10]:

$$\begin{aligned} w(i\xi) &= e^{\xi^2}\operatorname{erfc}(\xi) \\ &= e^{\xi^2}\cdot\frac{2}{\sqrt{\pi}}\int_{\xi}^{\infty}\mathrm{e}^{-\alpha^2}\mathrm{d}\alpha \end{aligned} \tag{23}$$

The lateral detection area has the same shape as in confocal FCS. Thus, correlation functions for two-dimensional diffusion of the confocal case apply and are not mentioned here.

Name **3D (Gauß/exp)**
ID **6013**
Descr. Three-dimensional free diffusion with a Gaussian lateral detection profile and an exponentially decaying profile in axial direction [5, 7, 13].

$$G(\tau) = \frac{1}{C}\frac{\kappa^2}{\pi(R_0^2+4D\tau)}\left(\sqrt{\frac{D\tau}{\pi}}+\frac{1-2D\tau\kappa^2}{2\kappa}w\left(i\sqrt{D\tau}\kappa\right)\right) \tag{24}$$

C	Particle concentration in confocal volume
κ	Evanescent decay constant ($\kappa = 1/d_{\mathrm{eva}}$)
R_0	Lateral extent of the detection volume
D	Diffusion coefficient

Name **3D+2D+T (Gauß/exp)**
ID **6033**
Descr. Two-component, two- and three-dimensional diffusion with a Gaussian lateral detection profile and an exponentially decaying profile in axial direction, including a triplet component [1, 4, 5, 7, 8, 13, 14].

$$\begin{aligned} G(\tau) = A_0 &+ \frac{1}{N(1-F+\alpha F)^2}\left(1+\frac{Te^{-\tau/\tau_{\mathrm{trip}}}}{1-T}\right)\times \\ &\times\left[\frac{1-F}{1+4D_{2\mathrm{D}}\tau/R_0^2}+\frac{\alpha^2 F\kappa}{1+4D_{3\mathrm{D}}\tau/R_0^2}\left(\sqrt{\frac{D_{3\mathrm{D}}\tau}{\pi}}+\frac{1-2D_{3\mathrm{D}}\tau\kappa^2}{2\kappa}w\left(i\sqrt{D_{3\mathrm{D}}\tau}\kappa\right)\right)\right] \end{aligned} \tag{25}$$

A_0	Offset
N	Effective number of particles in confocal volume ($N = N_{2\mathrm{D}} + N_{3\mathrm{D}}$)
$D_{2\mathrm{D}}$	Diffusion coefficient of surface bound particles
$D_{3\mathrm{D}}$	Diffusion coefficient of freely diffusing particles
F	Fraction of molecules of the freely diffusing species ($N_{3\mathrm{D}} = FN$)
α	Relative molecular brightness of particle species ($\alpha = q_{3\mathrm{D}}/q_{2\mathrm{D}}$)
R_0	Lateral extent of the detection volume
κ	Evanescent decay constant ($\kappa = 1/d_{\mathrm{eva}}$)
T	Fraction of particles in triplet (non-fluorescent) state
τ_{trip}	Characteristic residence time in triplet state

[10] In user-defined model functions, the Faddeeva function is accessible through `wofz()`. For convenience, the function `wixi()` can be used which only takes ξ as an argument and the imaginary i can be omitted.

Name **3D+3D+T (Gauß/exp)**
ID **6034**
Descr. Two-component three-dimensional diffusion with a Gaussian lateral detection profile and an exponentially decaying profile in axial direction, including a triplet component [1, 4, 5, 7, 8, 13, 14].

$$
\begin{aligned}
G(\tau) =& A_0 + \frac{1}{N(1-F+\alpha F)^2}\left(1+\frac{Te^{-\tau/\tau_{\text{trip}}}}{1-T}\right)\times \\
&\times\left[\frac{F\kappa}{1+4D_1\tau/R_0^2}\left(\sqrt{\frac{D_1\tau}{\pi}}+\frac{1-2D_1\tau\kappa^2}{2\kappa}w\left(i\sqrt{D_1\tau}\kappa\right)\right)+\right. \\
&\left.+\frac{(1-F)\alpha^2\kappa}{1+4D_2\tau/R_0^2}\left(\sqrt{\frac{D_2\tau}{\pi}}+\frac{1-2D_2\tau\kappa^2}{2\kappa}w\left(i\sqrt{D_2\tau}\kappa\right)\right)\right]
\end{aligned}
\tag{26}
$$

A_0	Offset
N	Effective number of particles in confocal volume ($N = N_1 + N_2$)
D_1	Diffusion coefficient of species 1
D_2	Diffusion coefficient of species 2
F	Fraction of molecules of species 1 ($N_1 = FN$)
α	Relative molecular brightness of particle species ($\alpha = q_2/q_1$)
R_0	Lateral extent of the detection volume
κ	Evanescent decay constant ($\kappa = 1/d_{\text{eva}}$)
T	Fraction of particles in triplet (non-fluorescent) state
τ_{trip}	Characteristic residence time in triplet state

5.4.3 TIR-FCS with a square-shaped lateral detection volume

The detection volume is axially confined by an evanescent field of depth[11] $d_{\text{eva}} = 1/\kappa$. The lateral detection area is a convolution of the point spread function of the microscope of size σ,

$$
\sigma = \sigma_0 \frac{\lambda}{NA}, \tag{27}
$$

with a square of side length a. The model functions make use of the Faddeeva function (complex error function)[12]:

$$
\begin{aligned}
w(i\xi) &= e^{\xi^2}\operatorname{erfc}(\xi) \\
&= e^{\xi^2}\cdot\frac{2}{\sqrt{\pi}}\int_{\xi}^{\infty}\mathrm{e}^{-\alpha^2}\mathrm{d}\alpha
\end{aligned}
\tag{28}
$$

[11] Where the field has decayed to $1/e$

[12] In user-defined model functions, the Faddeeva function is accessible through `wofz()`. For convenience, the function `wixi()` can be used which only takes ξ as an argument and the imaginary i can be omitted.

Name **2D (□xσ)**
ID **6000**
Descr. Two-dimensional diffusion with a square-shaped lateral detection area taking into account the size of the point spread function [11, 17][13].

$$G(\tau) = \frac{1}{C}\left[\frac{2\sqrt{\sigma^2 + D\tau}}{\sqrt{\pi}a^2}\left(\exp\left(-\frac{a^2}{4(\sigma^2 + D\tau)}\right) - 1\right) + \frac{1}{a}\operatorname{erf}\left(\frac{a}{2\sqrt{\sigma^2 + D\tau}}\right)\right]^2 \tag{29}$$

C	Particle concentration in detection area
σ	Lateral size of the point spread function
a	Side size of the square-shaped detection area
D	Diffusion coefficient

Name **3D (□xσ/exp)**
ID **6010**
Descr. Three-dimensional diffusion with a square-shaped lateral detection area taking into account the size of the point spread function; and an exponential decaying profile in axial direction [11, 17].

$$G(\tau) = \frac{\kappa^2}{C}\left(\sqrt{\frac{D\tau}{\pi}} + \frac{1 - 2D\tau\kappa^2)}{2\kappa}w\left(i\sqrt{D\tau}\kappa\right)\right) \times \tag{30}$$
$$\times\left[\frac{2\sqrt{\sigma^2 + D\tau}}{\sqrt{\pi}a^2}\left(\exp\left(-\frac{a^2}{4(\sigma^2 + D\tau)}\right) - 1\right) + \frac{1}{a}\operatorname{erf}\left(\frac{a}{2\sqrt{\sigma^2 + D\tau}}\right)\right]^2$$

C	Particle concentration in detection volume
σ	Lateral size of the point spread function
a	Side size of the square-shaped detection area
κ	Evanescent decay constant ($\kappa = 1/d_{\text{eva}}$)
D	Diffusion coefficient

Name **2D+2D (□xσ/exp)**
ID **6022**
Descr. Two-component two-dimensional diffusion with a square-shaped lateral detection area taking into account the size of the point spread function.
The correlation function is a superposition of two-dimensional model functions of the type **2D (□xσ)** (6000) [11, 17].

Name **3D+2D (□xσ/exp)**
ID **6020**
Descr. Two-component two- and three-dimensional diffusion with a square-shaped lateral detection area taking into account the size of the point spread function; and an exponential decaying profile in axial direction.
The correlation function is a superposition of the two-dimensional model function **2D (□xσ)** (6000) and the three-dimensional model function **3D (□xσ)** (6010) [11, 17].

Name **3D+3D (□xσ/exp)**
ID **6023**
Descr. Two-component three-dimensional free diffusion with a square-shaped lateral detection area taking into account the size of the point spread function; and an exponential decaying profile in axial direction.
The correlation function is a superposition of three-dimensional model functions of the type **3D (□xσ)** (6010) [11, 17].

Name **3D+2D+kin (□xσ/exp)**
ID **6021**
Descr. Two-component two- and three-dimensional diffusion with a square-shaped lateral detection area taking into account the size of the point spread function; and an exponential decaying profile in axial direction. This model covers binding and unbinding kintetics.
The correlation function for this model was introduced in [11]. Because approximations are made in the derivation, please verify if this model is applicable to your problem before using it.

Acknowledgements

I thank André Scholich (TU Dresden, Germany) for initial proof reading of the manuscript and Grzegorz Chwastek, Franziska Thomas, and Thomas Weidemann (Biotec, TU Dresden, Germany) for critical feedback on PyCorrFit.

References

[1] S. R. Aragon and R. Pecora. Fluorescence correlation spectroscopy as a probe of molecular dynamics. *The Journal of Chemical Physics*, 64(4):1791–1803, 1976. doi:10.1063/1.432357.

[2] M. Brinkmeier, K. Dörre, J. Stephan, and M. Eigen. Two-beam cross-correlation: a method to characterize transport phenomena in micrometer-sized structures. *Analytical Chemistry*, 71(3):609–616, Feb 1999. doi:10.1021/ac980820i.

[3] Markus Burkhardt. *Electron multiplying CCD – based detection in Fluorescence Correlation Spectroscopy and measurements in living zebrafish embryos*. PhD thesis, Biophysics, BIOTEC, Technische Universität Dresden, Tatzberg 47–51, 01307 Dresden, Germany, 2010. http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-61021.

[4] Elliot L. Elson and Douglas Magde. Fluorescence correlation spectroscopy. i. conceptual basis and theory. *Biopolymers*, 13(1):1–27, 1974. doi:10.1002/bip.1974.360130102.

[5] Kai Hassler, Tiemo Anhut, Rudolf Rigler, Michael Gösch, and Theo Lasser. High count rates with total internal reflection fluorescence correlation spectroscopy. *Biophysical Journal*, 88(1):L01–L03, January 2005. doi:10.1529/biophysj.104.053884.

[6] Ulrich Haupts, Sudipta Maiti, Petra Schwille, and Watt W. Webb. Dynamics of fluorescence fluctuations in green fluorescent protein observed by fluorescence correlation spectroscopy. *Proceedings of the National Academy of Sciences*, 95(23):13573–13578, 1998. doi:10.1073/pnas.95.23.13573.

[7] Yu Ohsugi, Kenta Saito, Mamoru Tamura, and Masataka Kinjo. Lateral mobility of membrane-binding proteins in living cells measured by total internal reflection fluorescence correlation spectroscopy. *Biophysical Journal*, 91(9):3456–3464, 2006. doi:10.1529/biophysj.105.074625.

[8] A. G. Palmer and N. L. Thompson. Theory of sample translation in fluorescence correlation spectroscopy. *Biophysical Journal*, 51(2):339–343, Feb 1987. doi:10.1016/S0006-3495(87)83340-4.

[9] Zdeněk Petrášek and Petra Schwille. Precise measurement of diffusion coefficients using scanning fluorescence correlation spectroscopy. *Biophysical Journal*, 94(4):1437–1448, February 2008. doi:10.1529/biophysj.107.108811.

[10] Hong Qian and Elliot L. Elson. Analysis of confocal laser-microscope optics for 3-d fluorescence correlation spectroscopy. *Applied Optics*, 30(10):1185–1195, Apr 1991. doi:10.1364/AO.30.001185.

[11] Jonas Ries, Eugene P. Petrov, and Petra Schwille. Total internal reflection fluorescence correlation spectroscopy: Effects of lateral diffusion and surface-generated fluorescence. *Biophysical Journal*, 95(1):390 – 399, 2008. doi:10.1529/biophysj.107.126193.

[12] R. Rigler, Ü. Mets, J. Widengren, and P. Kask. Fluorescence correlation spectroscopy with high count rate and low background: analysis of translational diffusion. *European Biophysics Journal*, 22:169–175, 1993. doi:10.1007/BF00185777.

[13] Tammy E. Starr and Nancy L. Thompson. Total internal reflection with fluorescence correlation spectroscopy: Combined surface reaction and solution diffusion. *Biophysical Journal*, 80(3):1575 – 1584, 2001. doi:10.1016/S0006-3495(01)76130-9.

[14] Nancy Thompson. Fluorescence correlation spectroscopy. In Joseph Lakowicz, Chris D. Geddes, and Joseph R. Lakowicz, editors, *Topics in Fluorescence Spectroscopy*, volume 1 of *Topics in Fluorescence Spectroscopy*, pages 337–378. Springer US, 2002. doi:10.1007/0-306-47057-8_6.

[15] Jerker Widengren, Ülo Mets, and Rudolf Rigler. Fluorescence correlation spectroscopy of triplet states in solution: a theoretical and experimental study. *The Journal of Physical Chemistry*, 99(36):13368–13379, 1995. doi:10.1021/j100036a009.

[16] Jerker Widengren, Rudolf Rigler, and Ülo Mets. Triplet-state monitoring by fluorescence correlation spectroscopy. *Journal of Fluorescence*, 4:255–258, 1994. doi:10.1007/BF01878460.

[17] Stoyan Yordanov, Andreas Best, Klaus Weisshart, and Kaloian Koynov. Note: An easy way to enable total internal reflection-fluorescence correlation spectroscopy (tir-fcs) by combining commercial devices for fcs and tir microscopy. *Review of Scientific Instruments*, 82(3):036105, 2011. doi:10.1063/1.3557412.

[18] Bo Zhang, Josiane Zerubia, and Jean-Christophe Olivo-Marin. Gaussian approximations of fluorescence microscope point-spread function models. *Applied Optics*, 46(10):1819–1829, Apr 2007. doi:10.1364/AO.46.001819.