The Triple Correlation Toolbox Manual

William K. Ridgeway, David P. Millar & James R. Williamson

Contents

1 Installation (make all)

The programs are for a unix / OS X environment and are compiled from the command line by using *make*.

1.1 Editing the Makefile

Open the file Makefile and verify that CC points to the correct location of gcc, e.g.
CC = /usr/local/bin/gcc
-or-
CC = /usr/bin/gcc
Note: You can find the path to GCC by typing

```
which gcc
```

If nothing was found, install the GCC package from http://gcc.gnu.org/ before proceeding. Triple Correlation Toolbox has been tested with GCC versions:
4.4.1, 4.6.1 (compiled from source, $\geq$ 4.6.1 recommended)
4.4.3 (Ubuntu 4.4.3-4ubuntu5)
4.2.1 (i686-apple-darwin11-llvm-gcc-4.2)

1.2 Compiling the easy stuff first: Installation for the impatient!

The impatient will probably want to compile something, anything, now, so they can change to the F3CS directory and type:

```
make no_libraries_reqd
```

This should compile several programs that do not have external dependencies. If you were to try to compile the rest of the programs by typing:

```
make all
```

this second command would be expected to fail if you did not have the appropriate libraries installed. Failure will generate errors such as, but not limited to:
`F3CS_LocalFit.c:19:25: gsl/gsl_rng.h: No such file or directory.` ($\Rightarrow$ GSL missing)
`F3CS_DAQ.c:17:21: error: NIDAQmx.h: No such file or directory.` ($\Rightarrow$ NIDAQmx missing)
`F3CS_GlobalFit_3D_Plot.c:39:27: error: plplot/plplot.h: No such file or directory.` ($\Rightarrow$ PLplot missing)
`/bin/sh: line 1: pkg-config: command not found.` ($\Rightarrow$ GTK+-2.0 missing)
The errors should disappear after installing the libraries listed in the next section.

1.3 Installing the required libraries

1.3.1 Gnu Scientific Library

Provides robust fitting routines. This library can be downloaded from:

http://www.gnu.org/s/gsl/

Tested with Versions 1.14 and 1.15

1.3.2 NIDAQmx

This is the library of drivers for the National Instruments (NI) data acquisition card. Different versions of NIDAQmx match different version of Linux, and at present this is denoted in the NI documentation "NI-DAQmx and NI-DAQ Driver Support: PCI/PXI Devices"

http://zone.ni.com/devzone/cda/tut/p/id/6910

Tested with NI-DAQmx v. 8.0.0f1 for Linux/x86 32-bit on Suse 9.3.

1.3.3 PLplot

Graphics routines for shaded plots. Can be downloaded from:

http://plplot.sourceforge.net/

Tested with v. 5.9.9. (linux only)

1.3.4 GTK+-2.0

This is used to create F3CS_AlignTool's graphical user interface. Depending on OS, a development version can be readily be installed via a package manager, but failing that it can be downloaded and built from:

http://www.gtk.org/

Tested with v. 2.6.4

1.4 Compiling the rest of the package

Now that the libraries are installed, all programs can be compiled by typing:

```
make all
```

Alternately, there are several options available for those who only want to compile specific programs, such as typing the specific name of a program, e.g. F3CS_AxBxG:

```
make F3CS_AxBxG
```

or you can compile various programs by library dependency if you choose to only install particular libraries:

```
make make no_libraries_reqd
make gsl_reqd
make nidaqmx_reqd
make plplot_reqd
make nidaqmx_AMD_reqd
```

Should the compilation of one program fail, in general the compilation of the others will not be directly effected. Even if errors are displayed, chances are you will find a few compiled executables by typing "ls F3CS*".

1.5 Installing the compiled executables in your path

Once the programs have compiled, they can be placed in one of the directories in your path, or simply sourced from the build directory. This is entirely up to the user's preferences. For example, this will put them in the /usr/local/bin directory and then run F3CS_StochasticData:

```
sudo cp F3CS_* /usr/local/bin/
rehash
F3CS_StochasticData _test1 70
```

2 Program invocation, usage and notes

Following the more scientific discussions of each program in the main manuscript, detailed instructions for running each program are presented here along with assorted technical details.

2.1 F3CS_AlignTool

Usage: F3CS_AlignTool &

There are no options when invoking this program. F3CS_AlignTool can be run remotely with X11 windowing and a low-latency, high-bandwidth internet connection.

2.2 F3CS_DAQ

Usage: F3CS_DAQ tag_string acquisition_time shutter_threshold
Example: F3CS_DAQ _rxn18 6000 125

Option	Explanation
tag_string	A string identifier that becomes part of the data file name.
acquisition_time	The number of seconds of data to take.
shutter_threshold	The APD intensity (in kHz) at which an optional electromechanical shutter will close to block incoming laser light.

This program needs reliable access to CPU resources at frequent intervals to prevent data overruns in the FIFO buffer of the data acquisition card. As such, it is not recommended to have other processes running during the data acquisition process. To some extent, the use of a larger

binning factor (16× in this work) can reduce the demands on the acquisition computer (binning factor is changed by selecting a different vectorized data processing function by altering the code and then recompiling, e.g. bits_2048_16_3 provides 16× binning, bits_2048_8_3 8× binning etc.). While our 2007-era hardware is capable of supporting 1× binning for 50 ns time resolution, the program overruns approximately every 100 s and data are not continuous; this problem should be reduced in more modern and higher-performance computer hardware.

The binary file format of .w3c files writes data as interleaved unsigned short integers as follows:

$$\left\{i_\alpha(t_0),i_\beta(t_0),i_\gamma(t_0),i_\alpha(t_1),i_\beta(t_1),i_\gamma(t_1),...,i_\alpha(t_{T-1}),i_\beta(t_{T-1}),i_\gamma(t_{T-1})\right\}$$

Both F3CS_AlignTool and F3CS_DAQ read data from the 8-channel waveform acquisition port of the data acquisition card. For three channel data acquisition, data are read from the following channels: 0.0, 0.2, and 0.4. Wire the output from the counter-timer circuits to these exact inputs. For model AQRH APDs (Perkin Elmer), AC-termination was necessary to detect TTL pulses that are apparently weaker than the AQR model shown in the circuit in the main manuscript.

Assembly code was tested on AMD Athlon 64 X2 4200+ and AMD Phenom II X6 1055T processors.

2.3 F3CS_DAQ_ASM

While usage and function are identical to F3CS_DAQ, F3CS_DAQ_ASM contains hand-tuned assembly code to speed up the inner loops of the processor-intensive data acquisition process. As the assembly-code version is less well-tested, it is advisable to only use F3CS_DAQ_ASM when data acquisition with F3CS_DAQ overruns too frequently. Given recent advances in GCC vectorization loops the difference in speed between the compiled and hand-coded versions is presently only about two- to three-fold, depending on machine. The assembly code is specific for AMD processors, not Intel, and was tested on AMD Athlon 64 X2 4200+ and AMD Phenom II X6 1055T processors.

2.4 F3CS_StochasticData

Usage: F3CS_StochasticData tag_string time
Example: F3CS_StochasticData _test1 70

Option	Explanation
tag_string	A string identifier that becomes part of the data file name.
time	Seconds of simulated data; the program will round this value up in order to create a complete datafile.

F3CS_StochasticData is a didactic substitute for F3CS_DAQ and is described only in the tutorial below (Section 3.1). It allows the user to generate test data and play with triple correlation integrals without a microscope or data acquisition hardware card. Run time is about a minute per 10 s of simulated data.

2.5 F3CS_Reverser

Usage: F3CS_Reverser tag_string first_file_num last_file_num
Example: F3CS_Reverser _rxn18 1 13

Option	Explanation
tag_string	A string identifier that becomes part of the data file name.
first_file_num	The number of the first rxn_*_tag.w3c file output by F3CS_DAQ, typically 1.
last_file_num	The number of the last rxn_*_tag.w3c file.

The explicit rearrangement of raw photon data files is:

$$\{i_\alpha(t_0),i_\beta(t_0),i_\gamma(t_0),i_\alpha(t_1),i_\beta(t_1),i_\gamma(t_1),...,i_\alpha(t_{T-1}),i_\beta(t_{T-1}),i_\gamma(t_{T-1})\} \rightarrow$$
$$\{i_\alpha(t_{T-1}),i_\beta(t_{T-1}),i_\gamma(t_{T-1}),i_\alpha(t_{T-2}),i_\beta(t_{T-2}),i_\gamma(t_{T-2}),...,i_\alpha(t_0),i_\beta(t_0),i_\gamma(t_0)\}$$

2.6 F3CS_TimeTrace

Usage: F3CS_TimeTrace filename time_quantum_ns output_hz tag
Example: F3CS_TimeTrace rxn_1_test1.w3c 800 2000 test1

Option	Explanation
filename	The name of the .w3c file to analyze
time_quantum	The time-resolution of the .w3c file, in nanoseconds. With a 20 MHz clock and 16× binning, this is 800 ns.
output_hz	The desired time resolution of the output file, 1 Hz minimum.
tag	A simple text identifier to add to the output filename (TimeTrace_tag.dat) and column headings.

F3CS_TimeTrace down-samples raw data and writes it to a tab-delimited text file so that data can be easily inspected and plotted. One .w3c file is processed at a time.

2.7 F3CS_2FCS

Usage: F3CS_2FCS tag_string first_file_num last_file_num num_cores
Example: F3CS_2FCS _rxn18 1 13 2

Option	Explanation
tag_string	A string identifier that becomes part of the data file name.
first_file_num	The number of the first rxn_*_tag.w3c file output by F3CS_DAQ, typically 1.
last_file_num	The number of the last rxn_*_tag.w3c file.
num_cores	The number of threads to run as. Keep at or below the max. number of cores in the computer. If data sets are small, this will default to the number of files divided by 2.

2.8 F3CS_AxBxG, F3CS_AxAxA, F3CS_AxAxB

Usage of the triple-correlation programs largely follows that of the double-correlation programs:

Usage: F3CS_AxBxG tag_string first_file_num last_file_num num_cores
Example: F3CS_AxBxG _rxn18 1 13 2

Option	Explanation
tag_string	A string identifier that becomes part of the data file name.
first_file_num	The number of the first rxn_*_tag.w3c file output by F3CS_DAQ, typically 1.
last_file_num	The number of the last rxn_*_tag.w3c file.
num_cores	The number of threads to run as. Keep at or below the max. number of cores in the computer. If data sets are small, this will default to the number of files over 4.

2.9 F3CS_Outlier2, F3CS_Outlier3

Both F3CS_Outlier2 and F3CS_Outlier3 are invoked as:

Usage: F3CS_Outlier2 tag_string first_file_num χ bin_size
Example: F3CS_Outlier2 _rxn18 1 2.8 128

Option	Explanation
tag_string	The string identifier that is part of the data file name (see F3CS_DAQ).
first_file	The number (#) of the first G_#_tag.bin or G_#_tag.bin2 file to consider.
χ	The number of standard deviations above which data are considered outliers. Higher numbers reject less data.
bin_size	The number of curves to bin together. (-1) defaults to the total number of curves in the data set.

2.10 F3CS_Difference

The usage of F3CS_Difference largely follows that of F3CS_Outlier2:

Usage: F3CS _Difference tag_string first_file_num χ bin_size
Example: F3CS _Difference _rxn18 1 2.8 128

Option	Explanation
tag_string	The string identifier that is part of the data file name (see F3CS_DAQ).
first_file	The number (#) of the first G_#_tag.bin or G_#_tag.bin2 file to consider.
χ	The number of standard deviations above which data are considered outliers. Higher numbers reject less data.
bin_size	The number of curves to bin together. (-1) defaults to the total number of curves in the data set.

2.11 F3CS_GlobalFit

Global fitting of data according to Eqs 14-17 is performed by F3CS_GlobalFit. The program takes an input from a text file, where each line specifies an initial guess of the fitting parameter and states whether this parameters should be held. The last bit, which determines fitting behaviour, is written as $\mathbb{F}$ in the list of parameters below, ($\mathbb{F} = 0 \Rightarrow$ hold, $\mathbb{F} = 1 \Rightarrow$ vary in fit). For example, if the diffusion time τ_D for species 3 is guessed to be 280 μs and you want to vary it in the fit, write: tauD_3=280,1. To hold it constant in the fit, write:tauD_3=280,0. In the table below (and occasionally in source-code) the notation for detectors $\{\alpha, \beta, \gamma\}$ is switched to $\{0, 1, 2\}$ for convenience.

Table 1: F3CS_GlobalFit fitting parameter syntax for input files

Syntax	**Parameter**	**Eqn.**
Molecular & background factors		
N_t_j=N_j,$\mathbb{F}$	The number of species j per focal volume	$14-17$
tauD_j=τ_D,$\mathbb{F}$	Diffusion time of species j, $\tau_{D,j}$. Units: μs	$14-17$
tauf_j=$\tau_{f,j}$,$\mathbb{F}$	Triplet decay time for species j.	$14-17$
T0_j=0.0,$\mathbb{F}$	Triplet fraction for species j	$14-15$
g_inf0x1=$G_{\alpha\times\beta}(\infty)$,$\mathbb{F}$	Asymptotic value, typically 1 ± 10^{-5}	$14-15$
g_inf0x1x2=$G_{\alpha\times\beta\times\gamma}(\infty,\infty)$,$\mathbb{F}$	Asymptotic value, typically 0 ± 10^{-3}	17
q_dye_dye=$Q_{\mathrm{dye,dye}}$,$\mathbb{F}$	Diagonal element of brightness matrix Q. Units: kHz / molecule	18
occ_0=$\{1.000, 0.000, ..., 0.820\}$	Row of the occupancy matrix	18
bg_0=b_α,$\mathbb{F}$	Background per channel. Units: kHz	16
Instrument factors		
gamma_j=0.1861,$\mathbb{F}$	Ratio between τ_D in $M(\tau)$ and $M'(\tau)$	$14-15$
frac_gamma_j=0.33,$\mathbb{F}$	Fraction of the fit in the $M'(\tau)$ form	$14-15$
omega=ω,$\mathbb{F}$	Axial ratio, dimensionless	$14-17$
gamma_R=1.65,$\mathbb{F}$	Triple-correlation volume ratio $\gamma_R = \gamma_3/\gamma_2^2$	17
Channel_Correction_0=$C_{\alpha,\alpha}$,$\mathbb{F}$	Only fit N-1 such terms, N=Num channels	18
Xi= $\{\{\Xi_{\mathrm{ch0,dye0}}, \Xi_{\mathrm{ch0,dye1}}, \Xi_{\mathrm{ch0,dye2}}\}, ..., \{\Xi_{\mathrm{ch2,dye0}}, \Xi_{\mathrm{ch2,dye1}}, \Xi_{\mathrm{ch2,dye2}}\}\}$	Bleedthrough matrix Ξ values	18
k2=$k_2^\alpha, k_2^\beta, k_2^\gamma$	Afterpulsing corrections (also for k3, k4)	10

Intensity_Weight=pre-factor
When intensities are fit alongside correlation curves, the weight of the fit is determined by a

combination of experimental errors (which are much lower for $i(t)$ than for $G(\tau)$) and a user-specified pre-prefactor. Best fits occur when this factor is simply 1.0. To disregard intensity for fitting, set this factor to 0.

Plot_Z_max = 0.0004,𝕌
Plot_Z_min = -0.00001,𝕌
Manual axis scales for plotting triple correlations all on the same scale (requires plplot package) (𝕌=1⇒ use, 𝕌=0 ⇒ do not use).

num_entries=n Enter the number of total files. (To help parse the text file.) Enter each file following this example:
tag=_rxn18 timepoint = 0.0
Files to process are entered by specifying the filename tag_string entered at data acquisition time, and the curve number (timepoint) generated by the contaminant rejection program.

Fit Ranges: Data are fit between $\tau_{\min}$ and $\tau_{\max}$ for double-correlations, and $(\tau_{1,\min}, \tau_{2,\min})$ and $(\tau_{1,\max}, \tau_{2,\max})$ for triple-correlations.
fit_range_start = x ⇒ $\tau_{\min}$ is the xth smallest element in the τ domain
fit_range_stop = y ⇒ $\tau_{\max}$ is the $(y-1)$th-largest element in the τ domain (e.g. $y = 0$ is the largest).
fit_range_start_triplecorr = x' As above, but for both τ_1 and τ_2.
fit_range_stop_triplecorr = y'

Usage: F3CS_GlobalFit input_file tag_string verbose_flag {plotdata — plotguess — fitdata — filegen}
Example: F3CS_GlobalFit input.file.18.txt _rxn18 0 fitdata

Option	**Explanation**
input_file	Filename of the file containing fit parameters
tag_string	A string identifier that becomes part of the data file name.
verbose_flag	0 or 1. 1 outputs details of fitting calculations and scripts to plot data
plotdata	– Plots the data in each data set
plotguess	– Graphs the initial guess and data
fitdata	– Fits each data entry as described above
filegen	– Lists available timepoints for the 1st data set in the list

Output: Fit constants are output to the file gtaul.fit_params.dat. If the plplot library is installed (see installation instructions), presentation-quality figures are output in a Shaded_tag.ps file which can be converted into a pdf and used for vector graphics. If the verbose tag is 1, a variety of gnuplot scripts are output with which the primary double-correlation data can be analyzed.

2.12 F3CS_GlobalFit_3D_Plot

Usage is identical to F3CS_GlobalFit.

2.13 F3CS_LocalFit

Usage: F3CS_LocalFit intput_file tag_string
Example: F3CS_LocalFit input.30S.txt _rxn18

Option	Explanation
input_file	Path to a text file containing instructions for the fit.
tag_string	A string identifier that becomes part of the data headings.

Please find a prototype input file "input.local.txt" provided with the sourcecode. The input file is processed very similarly to the F3CS_GlobalFit input file in that both initial guess and the fit / don't fit options are specified as value,{0,1,2}, but as explained in the tutorial below there are additional options aside from fit locally, fit globally or don't fit at all.

3 Tutorial

This section assumes that you have already compiled all or some of the individual programs that make up Triple Correlation Toolbox. Please see Section 1 if you have not yet done this.

3.1 Generating simulated data for the tutorial

A simulated data acquisition program, F3CS_StochasticData, allows the user to test the basic functionality of the software package without installing a National Instruments data acquisition card that is required to run both data acquisition programs F3CS_DAQ and F3CS_AlignTool. The data simulate 10 immobilized molecules that each transition between three states (Fig. 1) in a futile cycle. Each of the states has a different "colour" in that the intensity of state one mostly is detected in the first detector, state two mostly in the second detector and state three mostly in the third detector. The simulation first calculates state trajectories then calculates detected photon counts as a random poisson process where the mean emission rate depends both on the state and the detector. To make it plain that these data are simulated, they look very different from diffusion data and fitting programs later in the tutorial will not fit these data sensibly.

Run the following command to generate > 70 s of simulated data (actually about 80 s in order to complete the last raw data file).

```
F3CS_StochasticData _test1 70
```

(Note: Actual data acquisition proceeds similarly, e.g. one types: "F3CS_DAQ _test1 70 500" where the extra term 500 specifies that the instrument will optionally shutter the laser to protect the detectors when counts exceed 500 kHz)

F3CS_StochasticData will take about 7 mins to run, and will produce 6 files:

rxn_1_test1.w3c
rxn_2_test1.w3c
...
rxn_6_test1.w3c

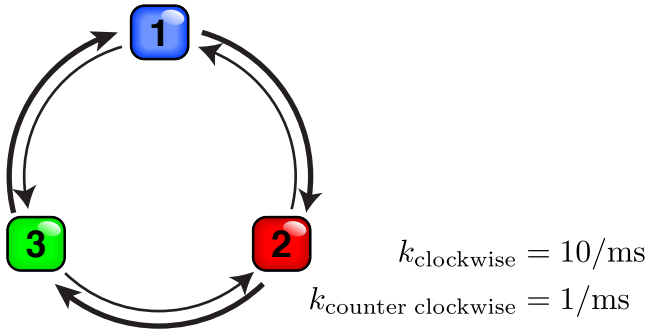

Figure 1: The three-state reaction scheme used to generate data in F3CS_StochasticData. Since the product of the clockwise rates differs from the product of the counterclockwise rates, the scheme corresponds to a futile cycle.

The files represent 13 s of data each, and filenames contain the string "_test1," which is an identifier that we will use throughout the analysis of this dataset.

There's a problem though – just like real microscope data, the data output by F3CS_StochasticData contain a simulated optical artifact. What's worse, I'm not going to tell you where it is, and it will ruin our data if we don't get rid of it.

3.2 Examining raw data

We can examine the "raw data" using F3CS_TimeTrace and the free plotting program gnuplot (http://www.gnuplot.info/):

```
F3CS_TimeTrace rxn_1_test1.w3c 800 2000 test1
```

The above command-line arguments specify that the data were collected with 800 ns time resolution and should be down-sampled to 2 kHz. Data are output to a .dat file (TimeTrace_test1.dat) that can be visualized in gnuplot; here the first 100 ms are plotted (Fig. 2)

```
gnuplot> plot [0:0.1][] "TimeTrace_test1.dat" u 1:2 w l,
"TimeTrace_test1.dat" u 1:3 w l, "TimeTrace_test1.dat" u 1:4 w l
```

The data are not uniform, but are rich with fluctuations – brilliant! There is still that issue of the lurking artifact though... extra credit goes to those who can find it with F3CS_TimeTrace.

3.3 Analyzing data with FCS and FCCS

We can now perform double-correlation analysis using F3CS_2FCS to calculate the FCS and FCCS integrals:

```
F3CS_2FCS _test1 1 6 8
```

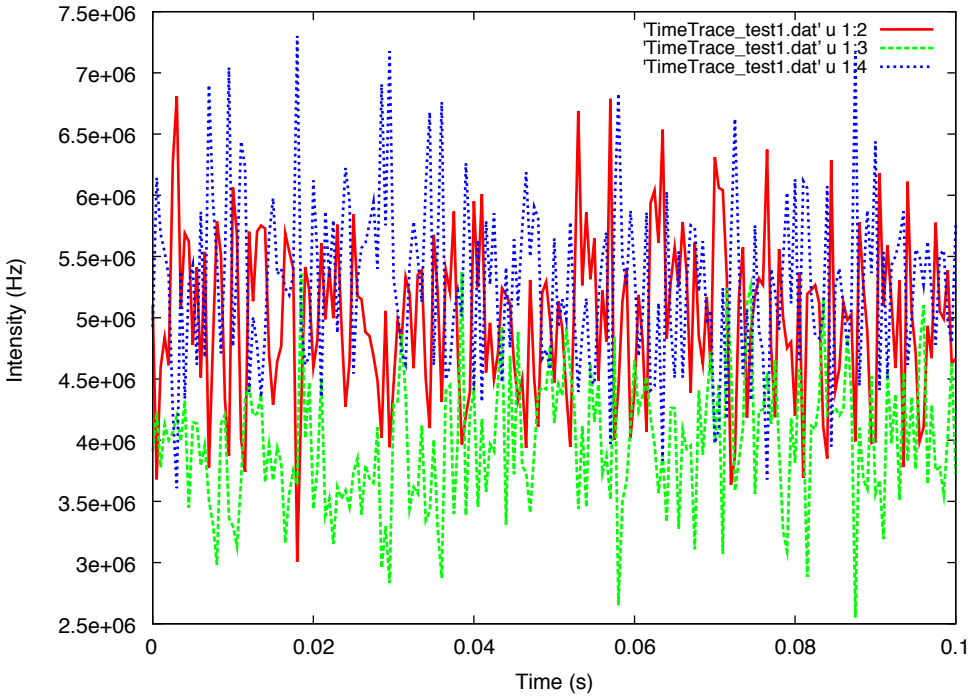

Figure 2: Time trace of 100ms of the simulated stochastic dataset.

This should take about a minute to run. The "_test1" string identifies the data again, the numbers 1 and 7 denote the numbers of the first and last datafile respectively, and 8 tells the program to use up to 8 cores of the CPU. Since this dataset is relatively small, only three cores were utilized, one per pair of datafiles. Three output .bin files are generated, one for every pair of .w3c files processed:

G_array_test1_1.bin
G_array_test1_3.bin
G_array_test1_5.bin

The binary .bin files contain correlation curves calculated from sequential fragments of data. Referring to the data fragments as timepoints, for each timepoint three average intensities and six correlation curves are calcualted: $G_{\alpha\times\alpha}(\tau), G_{\beta\times\beta}(\tau), G_{\gamma\times\gamma}(\tau), G_{\alpha\times\beta}(\tau), G_{\beta\times\gamma}(\tau), G_{\gamma\times\alpha}(\tau)$. Intensities and correlation curves from each time-points are stored separate after this stage, though most likely they will be combined later. This grants the user flexibility to later bin the data to arbitrary time resolutions, and also prevent occasional optical artifacts from contaminating the entire dataset.

At this point, we can't look at the raw correlation data because they are stored in binary format, and also because there are so many individual curves. Before we can look at the data, we have to bin and possibly remove artifacts.

3.4 Binning data & Removing optical artifacts

This next step eliminates outlier data generated by optical artifacts. In general, outlier rejection proceeds by looking for curves which differ relative to the others, and of course this could be a natural consequence of an evolving reaction. For now, we will assume that the system is stationary (which is correct for this simulation), and analyze all data as a single time-point. To do this, type:

```
F3CS_Outlier2 _test1 1 3.0 -1
```

The command-line argument 1 tells the program to start with the first G_array file, i.e. G_array_test1_1.bin. The statistical cutoff of $\chi = 3.0$ eliminates data that are different from the mean by more than 3.0 standard deviations and -1 tells the program to analyze all of the dataset as one measurement. In this case, screen output is as follows:

```
...
Total number of curves in the data = 767, from  80.53 s data
Binning matched to number of curves (767)
Outlier at time 11.64 s (curve 111/767),  Error = 26.834 x sigmas
Outlier at time  9.96 s (curve  95/767),  Error =  6.139 x sigmas
Outlier at time 10.38 s (curve  99/767),  Error =  6.298 x sigmas
Outlier at time 11.11 s (curve 106/767),  Error =  6.473 x sigmas
Outlier at time 11.53 s (curve 110/767),  Error =  6.663 x sigmas
Outlier at time 11.22 s (curve 107/767),  Error =  6.871 x sigmas
Outlier at time 10.80 s (curve 103/767),  Error =  7.099 x sigmas
Outlier at time 10.07 s (curve  96/767),  Error =  7.352 x sigmas
Outlier at time 10.49 s (curve 100/767),  Error =  7.634 x sigmas
Outlier at time 10.91 s (curve 104/767),  Error =  7.951 x sigmas
Outlier at time 10.59 s (curve 101/767),  Error =  8.312 x sigmas
Outlier at time 11.32 s (curve 108/767),  Error =  8.726 x sigmas
Outlier at time 10.17 s (curve  97/767),  Error =  9.209 x sigmas
Outlier at time 10.70 s (curve 102/767),  Error =  9.782 x sigmas
...
Outlier at time 51.38 s (curve 489/767),  Error =  3.014 x sigmas
Outlier at time  6.40 s (curve  61/767),  Error =  3.008 x sigmas
Outlier at time 16.88 s (curve 161/767),  Error =  3.013 x sigmas
33 curves were discarded from this bin
...
```

Here we can see that the dataset was analyzed in 767 individual pieces, and the most outlier-like dataset was 26.834 standard deviations different from the mean. After that curve was eliminated, the mean and standard deviation were recalculated and data reexamined to find another curve at 6.139 sigmas, et cetera. Note that as curves are removed the worst standard deviation eventually drops such that no outliers are greater than the 3.0 sigma criterion we specified on the command-line. However, since the standard deviation is reevaluated every time a curve is thrown out, this behaviour is not guaranteed and some datasets will be decimated before the outlier rejection program finishes. In such cases, the user should try a larger χ value, and evaluate the dataset critically for usability.

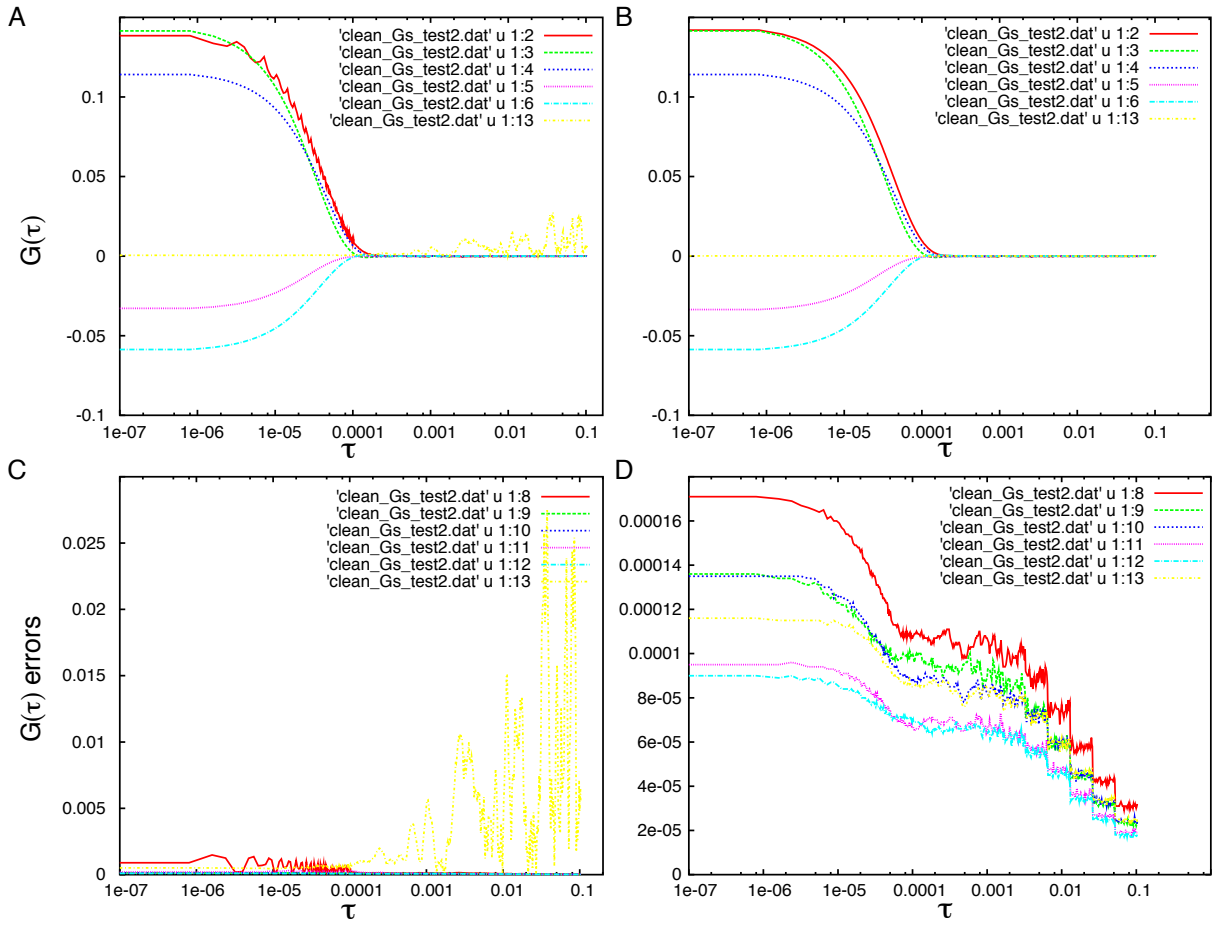

Figure 3: **A** Correlation data before outlier rejection show ripples in $G_{\alpha\times\alpha}(\tau)$ (red) and $G_{\gamma\times\alpha}(\tau)$ (yellow). **B** The same data after outlier rejection lack the ripples. **C** Experimental errors before rejection are quite large, while **D** errors after rejection are much smaller on the scale of point-to-point fluctuations in the correlation data.

Did the program remove the artifact we had introduced in F3CS_StochasticData? Yes, by and large. The artifact was introduced at time 10.0 s and persisted until time 11.67 s, and F3CS_Outlier2 finds the greatest outliers between 9.96 s and 11.64s. Note that data outside of that range were also eliminated at the end, e.g. at time = 16.88 s. This is because the random nature of the simulation will create a distribution of curves such that one curve will be least similar to the rest and will appear as an apparent outlier. More than a technicality, this point illustrates that any outlier rejection program will remove legitimate data to some degree and it is frequently difficult for the user to determine whether legitimate data are being discarded. The best way to do this is to look for elimination of patterned noise from the traces.

According, the process can be monitored by comparing data before and after the outlier rejection stage, as illustrated in Fig. 3. Data are output to two files:

bin_clean_test1.bin
clean_Gs_test1.dat

The first file is binary and is to be read by datafitting programs and the second contains double-correlation FCS curves that can be plotted using third-party programs such as gnuplot. A few gnuplot commands produce a plot of $G_{\alpha\times\alpha}(\tau)$, $G_{\beta\times\beta}(\tau)$ and $G_{\gamma\times\gamma}(\tau)$ FCS data (Fig. 4):

```
gnuplot> set logscale x
gnuplot> plot 'clean_Gs_test1.dat' u 1:2 w l,
         'clean_Gs_test1.dat' u 1:3 w l,
         'clean_Gs_test1.dat' u 1:4 w l
```

The numbers "x" following "u 1:x" correspond to data columns, where the second column of the .dat file contains $G_{\alpha\times\alpha}(\tau)$, the third $G_{\beta\times\beta}(\tau)$, the fourth $G_{\gamma\times\gamma}(\tau)$, the fifth $G_{\alpha\times\beta}(\tau)$, the sixth $G_{\beta\times\gamma}(\tau)$ and the seventh $G_{\gamma\times\alpha}(\tau)$. Curves $8-13$ contain the error estimates for the correlation curves, in the same order. Accordingly, $G_{\alpha\times\beta}(\tau)$, $G_{\beta\times\gamma}(\tau)$ and $G_{\gamma\times\alpha}(\tau)$ FCCS data (Fig 5) are plotted using:

```
gnuplot> plot 'clean_Gs_test1.dat' u 1:5 w l,
         'clean_Gs_test1.dat' u 1:6 w l,
         'clean_Gs_test1.dat' u 1:7 w l
```

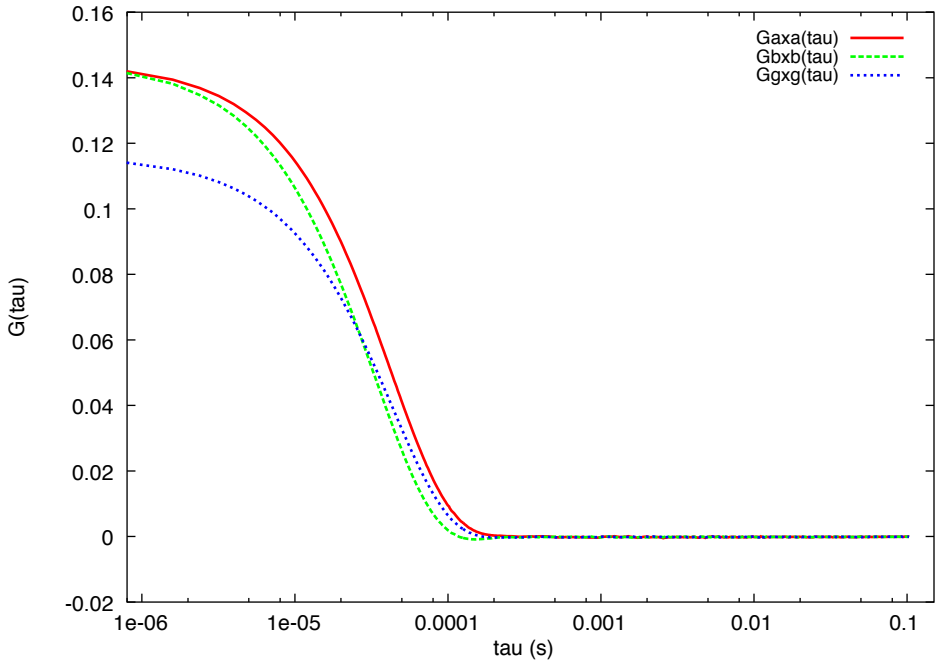

Figure 4: Sample FCS curves from the stochastic process

3.5 Treating datasets as evolving reactions

If the sample is reacting slowly, i.e. on a time-scale $100\times$ slower than the greatest value of τ, the dataset can be split into multiple parts for analysis. This is done in the outlier rejection stage by specifying how many curves to bin. To be precise, each FCS or FCCS curve is the result of

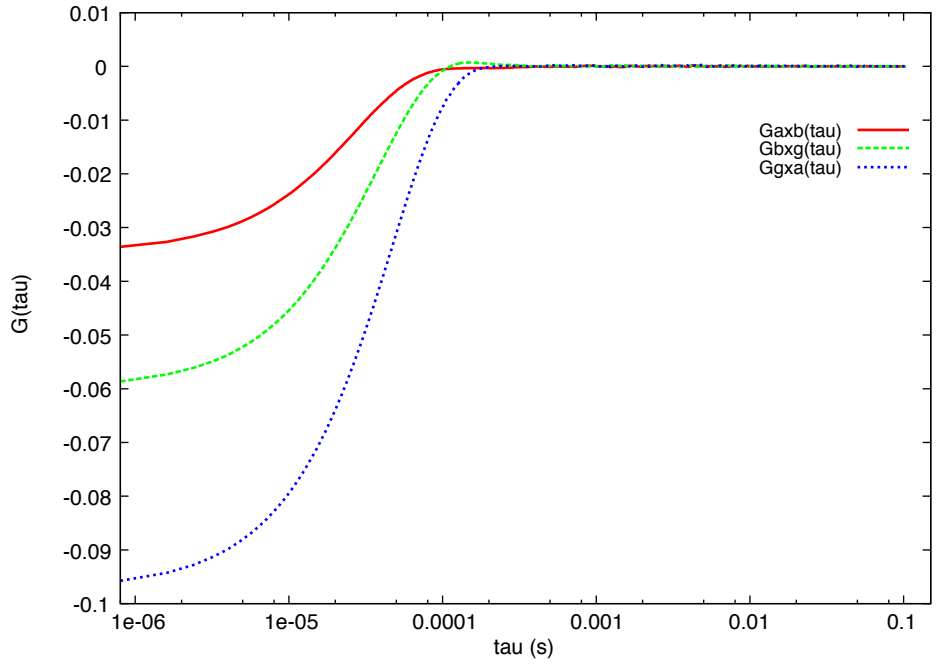

Figure 5: Sample FCCS curves from the stochastic process. Note: In contrast with the simulated data, both FCS and FCCS curves resulting from pure diffusion would be positive.

a single correlation integral calculation from a segment of the raw data that is the length of the maximum correlation time, about 100ms in this case.

Here, 16 curves are binned by changing the last argument fed to F3CS_Outlier2 from "-1" to "16,"

```
F3CS_Outlier2 _test1 1 10 16
```

Similarly, F3CS data are processed with F3CS_Outlier3 (this is just for reference, don't do this now because we haven't evaluated F3CS integrals yet):

```
F3CS_Outlier3 _test1 1 10 2
```

The binning numbers for F3CS_Outlier2 and F3CS_Outlier3 must be 8× different. This is because F3CS data are in general less precise than FCS data, and so each F3CS curve is calculated from 8× the amount of data. Thus the bin number for F3CS_Outlier3 must be $1/8^{\text{th}}$ of the bin number for F3CS_Outlier2 in order to obtain the same time-resolution for both FCS and F3CS data.

Curves can again be visualized in gnuplot. To plot the n^{th} datapoint, specify the $(12 \times n)^{th}$ curve, for example $G_{\alpha\times\alpha}(\tau)$ at times $0, 1, 5$ (Fig. 6):

```
gnuplot> plot 'clean_Gs_test1.dat' u 1:2 w l,
         'clean_Gs_test1.dat' u 1:14 w l,
         'clean_Gs_test1.dat' u 1:62 w l
```

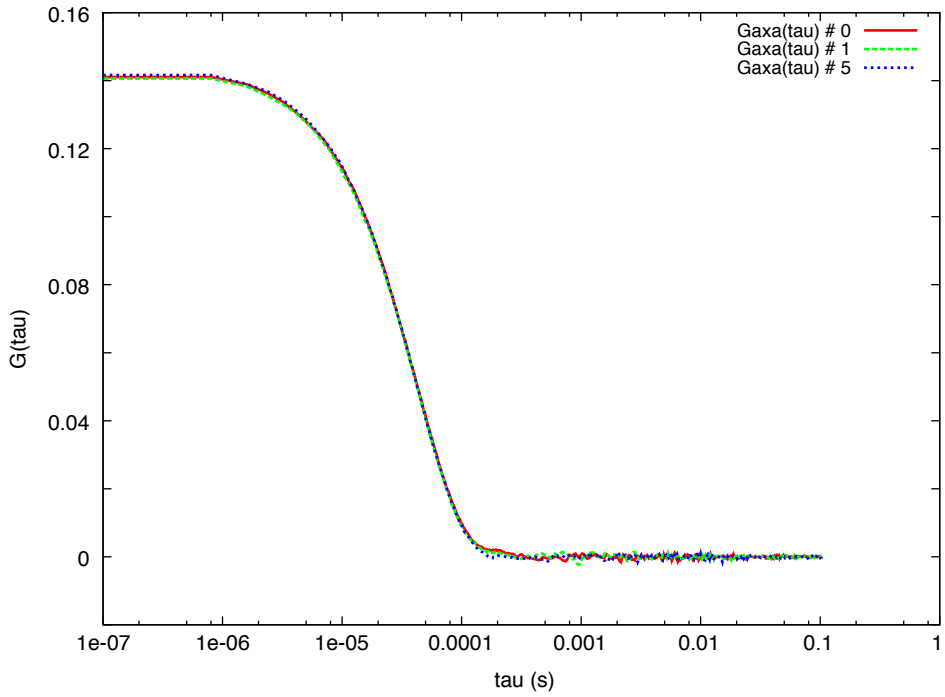

Figure 6: Sample FCS curves from sequential segments of the dataset. Well, this is a bit boring, they're all the same because the simulated data arise from a stationary process and the curves only differ as a result of statistical sampling. However, data from a reaction on a time-scale slower than τ could show a gradual evolution, e.g. binding of ribosomal proteins to the 16S rRNA would shift the apparent τ_{D} to the right as in Fig. 8B of the main manuscript.

3.6 Analyzing data with F3CS

Right, let's compute three correlations: $G_{\alpha\times\beta\times\gamma}(-\tau_1,-\tau_2), G_{\beta\times\gamma\times\alpha}(-\tau_1,-\tau_2)$ and $G_{\gamma\times\alpha\times\beta}(-\tau_1,-\tau_2)$ by running to following:

```
F3CS_AxBxG _test1 1 6 8
```

This takes about 3 mins depending computer speed. The correlation integrals are written to binary files:

G_array_test1_1.bin2
G_array_test1_3.bin2
G_array_test1_5.bin2

Let's not do this here, but in principle the other types of F3CS data could have been computed by running the following, for $G_{\alpha\times\alpha\times\beta}(-\tau_1,-\tau_2), G_{\beta\times\beta\times\gamma}(-\tau_1,-\tau_2)$ and $G_{\gamma\times\gamma\times\alpha}(-\tau_1,-\tau_2)$ you would run:

```
F3CS_AxAxB _test1 1 6 8
```

And for $G_{\alpha\times\alpha\times\alpha}(-\tau_1,-\tau_2), G_{\beta\times\beta\times\beta}(-\tau_1,-\tau_2)$ and $G_{\gamma\times\gamma\times\gamma}(-\tau_1,-\tau_2)$ you would run:

```
F3CS_AxAxA _test1 1 6 8
```

Returning to the output from F3CS_AxBxG, outliers can be eliminated using the F3CS-specific outlier rejection program, which requires both .bin and .bin2 files:

```
F3CS_Outlier3 _test1 1 2.8 -1
```

This should reject about four curves. Outlier rejection is accomplished by performing statistical tests solely on double-correlation data, for consistency when fitting double and triple correlation data. After outliers are eliminated, the reduced data are output in both binary format (bin_clean_test1.bin2) and text format (clean_GGGs_test1.dat). To see the correlated functions, for example $G_{\gamma\times\alpha\times\beta}(\tau_1,\tau_2)$, we can again use gnuplot (Fig 7):

```
gnuplot> set logscale x
gnuplot> set logscale y
gnuplot> set ticslevel 0
gnuplot> splot 'clean_GGGs_test1.dat' u 1:2:5 w pm3d
```

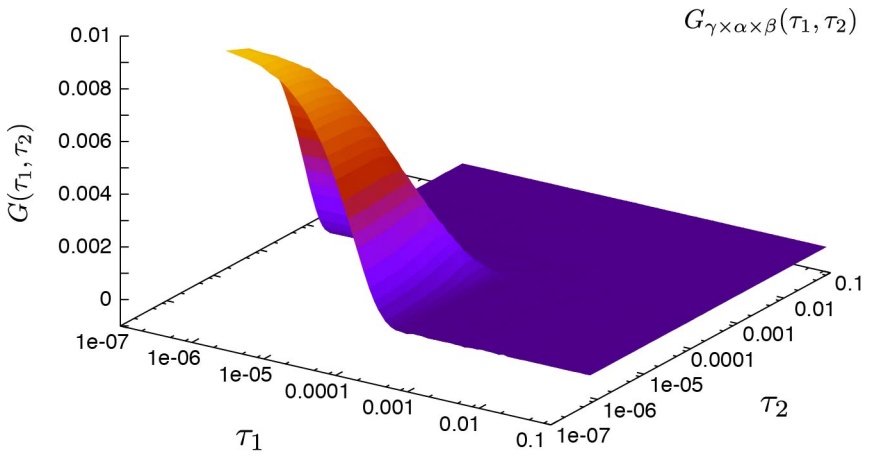

Figure 7: $G_{\gamma\times\alpha\times\beta}(\tau_1,\tau_2)$ analysis of the stochastic data

Several types of F3CS analysis from F3CS_AxBxG, F3CS_AxAxA and F3CS_AxAxB are further shown in Fig. 8, demonstrating the different information present in each type of triple correlation integral.

3.7 Fitting data using F3CS_LocalFit

This program is a quick-and-dirty way to fit FCS and FCCS data to the classical fit parameters that describe amplitude (N), infinite time values ($G(\infty)$), diffusion time (τ_D) and beam eccentricity (ω). Unlike F3CS_GlobalFit, each curve is fit without any notion of brightness, and F3CS data can not be fit. Rather, this program is most useful for analyzing titrations in a consistent way or obtaining fit parameters that form the basis of initial guesses for more complex fits in F3CS_GlobalFit. Real data, such as multi-component reactions, are sometimes more complex than the $N, G(\infty), \tau_D, \omega$ can describe, and so additional fit parameters allow the user to fit data

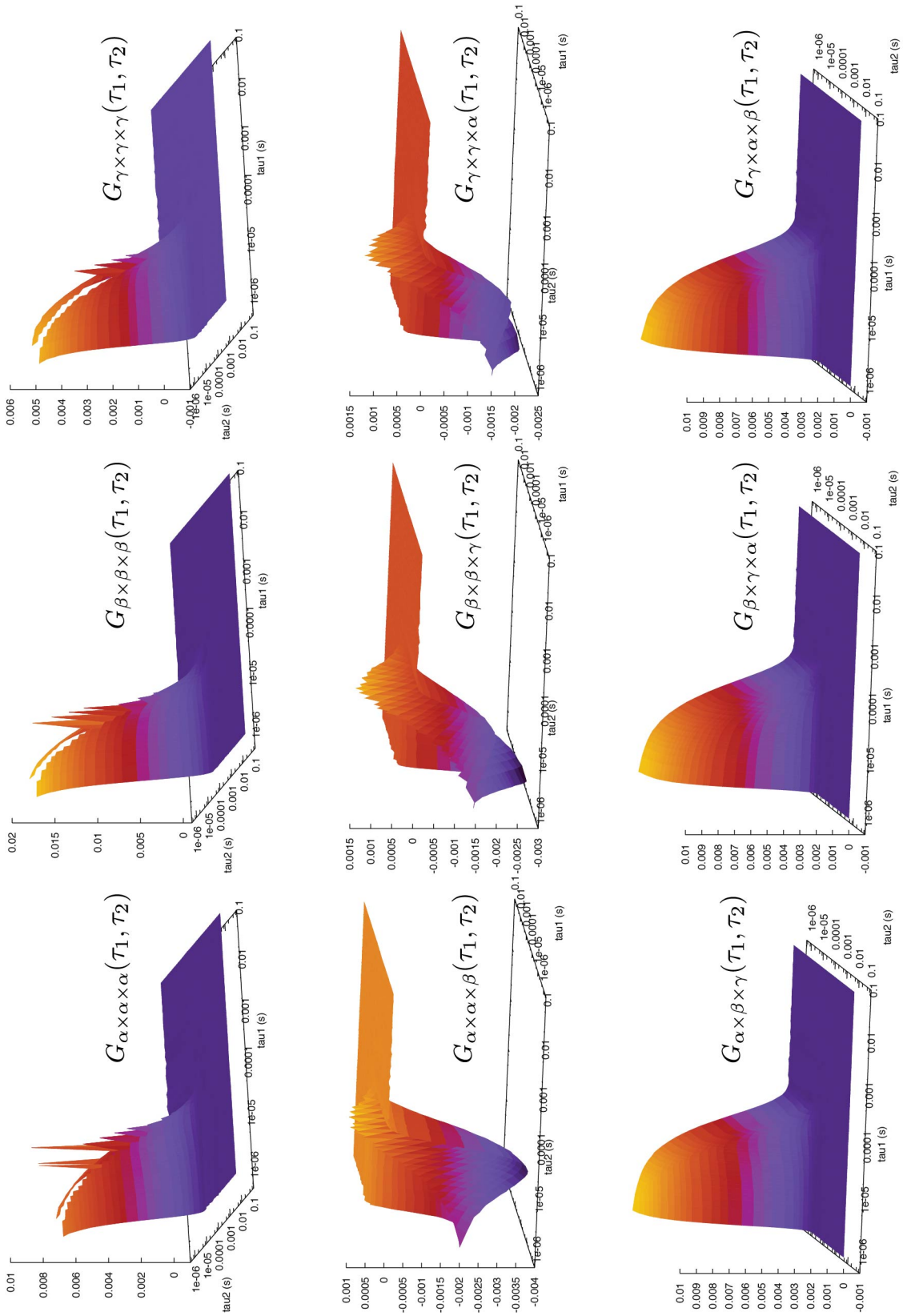

Figure 8: Nine different types of F3CS analysis. Each function analyzes the same data in mathematically unique ways, such that for non-zero delay times (not plotted on the log scales) there are no duplicate datapoints in this slide.

as a function of two species that have different τ_D values; θ is the fraction of the sample in one state. Even more fit parameters allow for the treatment of the optoelectronic artifact of afterpulsing and the dye photophysical phenomena of triplet states, and the reader is referred back to the main manuscript for exact fitting equations and fit coefficients.

Let's start by fitting the FCS curves generated in the previous section. To make sure we're on the same page, quickly bin all the data together using F3CS_Outlier2:

```
F3CS_Outlier2 _test1 1 3.0 -1
```

These data can be analyzed by running F3CS_LocalFit with the control script input.local.txt located in the F3CS_Tools directory. Just a note, this script is very finicky in that F3CS_LocalFit expects each input parameter to be presented in the same order, and errors will occur if the user does things such as swapping lines or deleting the double slashes that separate sections.

If you open the control file and scroll down to line 20, you should see the following lines that tell the F3CS_LocalFit which files to operate on, by simply specifying the number of files (num_entries) and then a tag for each file:

```
Data Location, Description (timepoints 0-indexed)
num_entries=3
tag=_test1 timepoint = 0.0 corr=0x0
tag=_test1 timepoint = 0.0 corr=1x1
tag=_test1 timepoint = 0.0 corr=2x2
```

Different tags will lead to different files being opened. Timepoint specifies the zero-indexed number of the reaction to process. If we had run F3CS_Outlier2 with a smaller bin size such that multiple timepoints were processed, we could fit these later timepoints by setting timepoint=1.0, timepoint=2.0, etc. "Corr" specifies the type of auto- or cross-correlation, with 0x0 implying $G_{\alpha\times\alpha}(\tau)$, 1x1 implying $G_{\beta\times\beta}(\tau)$, 2x0 implying $G_{\gamma\times\alpha}(\tau)$ etc. If we wanted to analyze all six FCS and FCCS curves, we could change this section to read:

```
Data Location, Description (timepoints 0-indexed)
num_entries=6
tag=_test1 timepoint = 0.0 corr=0x0
tag=_test1 timepoint = 0.0 corr=1x1
tag=_test1 timepoint = 0.0 corr=2x2
tag=_test1 timepoint = 0.0 corr=0x1
tag=_test1 timepoint = 0.0 corr=1x2
tag=_test1 timepoint = 0.0 corr=2x0
```

Before we go into detail about how to fit a set of curves, let's use the program to do an initial fit. As mentioned earlier, this fit will be a very bad fit because the simulated data are not diffusion data, but instead reaction data for immobilized molecules. But we can still illustrate some relevant aspects of the fitting program. Run the fit as follows:

```
F3CS_LocalFit input.local.txt badfit
```

A number of things should appear on the screen:

```
Fitting round 1: Number of Parameters: 1
Status = success, Red.Chi^2 = 4262.03880883
```

```
...
Status = success, Red.Chi^2 = 2765.32987616
Fitting Round 1 complete:  chisq/dof = 2765.33
Fit parameters:
N: 5.5527/0.0004 Th: 1.0000/0.0000 TD1: 30.0000/0.0000 TD2 :1567.0000/0.0000
w: 1.1000/0.0000 G(inf):-0.0000/0.0000
Tf1: 0.5000/0.0000 Tf2: 1.5000/0.0000 T0,1: 0.1000/0.0000 T0,2: 0.0000/0.0000
Gm1: 0.1861/0.0000 Gm2: 0.3500/0.0000 fGm1: 0.3300/0.0000 fGm2: 0.3300/0.0000

N: 5.5527/0.0004 Th: 1.0000/0.0000 TD1: 30.0000/0.0000 TD2 :1567.0000/0.0000
w: 1.1000/0.0000 G(inf):-0.0000/0.0000
Tf1: 0.5000/0.0000 Tf2: 1.5000/0.0000 T0,1: 0.1000/0.0000 T0,2: 0.0000/0.0000
Gm1: 0.1861/0.0000 Gm2: 0.3500/0.0000 fGm1: 0.3300/0.0000 fGm2: 0.3300/0.0000

N: 5.5527/0.0004 Th: 1.0000/0.0000 TD1: 30.0000/0.0000 TD2 :1567.0000/0.0000
w: 1.1000/0.0000 G(inf):-0.0000/0.0000
Tf1: 0.5000/0.0000 Tf2: 1.5000/0.0000 T0,1: 0.1000/0.0000 T0,2: 0.0000/0.0000
Gm1: 0.1861/0.0000 Gm2: 0.3500/0.0000 fGm1: 0.3300/0.0000 fGm2: 0.3300/0.0000


Fitting round 2: Number of Parameters: 3
Status = success, Red.Chi^2 = 2768.31879262
...
Status = success, Red.Chi^2 = 2750.38402443
Status = iteration is not making progress towards solution, Red.Chi^2 = 2750.38402443
Fitting Round 2 complete:  chisq/dof = 2750.38
Fit parameters:
0x0_test1_0.00 Red.Chi^2: 3244.6875 N: 5.5779/0.0010 Th: 1.0000/0.0000
TD1: 30.6758/0.0112 TD2 :1567.0000/0.0000 w: 1.1000/0.0000 G(inf):-0.0000/0.0000
Tf1: 0.1782/16.5736 Tf2: 1.5000/0.0000 T0,1: 0.1000/0.0000 T0,2: 0.0000/0.0000
Gm1: 0.1861/0.0000 Gm2: 0.3500/0.0000 fGm1: 0.3300/0.0000 fGm2: 0.3300/0.0000

1x1_test1_0.00 Red.Chi^2: 2915.9217 N: 5.5779/0.0010 Th: 1.0000/0.0000
TD1: 30.6758/0.0112 TD2 :1567.0000/0.0000 w: 1.1000/0.0000 G(inf):-0.0000/0.0000
Tf1: 0.1782/16.5736 Tf2: 1.5000/0.0000 T0,1: 0.1000/0.0000 T0,2: 0.0000/0.0000
Gm1: 0.1861/0.0000 Gm2: 0.3500/0.0000 fGm1: 0.3300/0.0000 fGm2: 0.3300/0.0000

2x2_test1_0.00 Red.Chi^2: 2090.5429 N: 5.5779/0.0010 Th: 1.0000/0.0000
TD1: 30.6758/0.0112 TD2 :1567.0000/0.0000 w: 1.1000/0.0000 G(inf):-0.0000/0.0000
Tf1: 0.1782/16.5736 Tf2: 1.5000/0.0000 T0,1: 0.1000/0.0000 T0,2: 0.0000/0.0000
Gm1: 0.1861/0.0000 Gm2: 0.3500/0.0000 fGm1: 0.3300/0.0000 fGm2: 0.3300/0.0000

Errors:  5.19e-02  5.85e-01  8.69e+02

Overall Red. Chi Sq: 2750.3840
For Plotting: gnuplot> load 'gnuplot_test1.txt'
```

```
For Plotting: gnuplot> load 'gnuplot_test1.fits.txt'
For Plotting: gnuplot> load 'gnuplot_test1.errors.txt'
For Plotting: gnuplot> load 'gnuplot_test1.resids.txt'
...
Program completed.
```

Fitting takes place in two rounds, the idea being that you fit the most crude values, such as N and $G(\infty)$ here first, then fit all the values together later, and this two-step process helps find global minima more efficiently. You tell the program which values to fit in each round by assigning each fit parameter an integer between 0 and 4. 0 means don't fit this parameter. 1 or 2 mean fit this parameter in the second fitting round only, 3 or 4 mean fit this parameter in the first and second rounds. The integers are specified to the right of the initial values, for example here N is a 3, τ_{D} is a 1 and all others are 0s.

```
N_t=8,3
theta=1.0,0
tauD_1=30,1
tauD_2=1567,0
omega=1.1,0
...
```

Looking at the above output again, the reduced χ^2 value decreases in every round. Ideally, this value should be between 0.7 and 2.0, and here the large magnitude shows that the fit is rubbish, as we'd expect. Let's take a look at the fits, by opening up gnuplot from the same directory and running the commands that the program suggests at the end of the screen output, i.e. for fits (Fig. 9):

```
load 'gnuplot_test1.fits.txt'
```

and for weighted residuals (Fig. 10):

```
load 'gnuplot_test1.resids.txt'
```

Obviously we could fit the data better if each curve was fit with a unique amplitudes N and $G(\infty)$ and delay times τ_{D}. To do this, modify the input.local.txt file as follows, and also include a triplet-state amplitude to help the fit at shorter times:

```
N_t=8,4
...
tauD_1=30,2
...
g_inf=-0.00001,4
...
T0_1=0.1,2
```

The new fits and residuals (Figs. 11, 12) are better because the reduced χ^2 value has decreased to from ~ 2700 to ~ 800 (again, such a large χ^2 means the model is wrong, which we knew.) At any rate, the key point here is that F3CS_LocalFit can have both individual and global fit parameters. This is particularly useful when you are trying to analyze a family of related curves such as a reaction, where it is beneficial to fit all data with the same τ_{D} and ω values for consistency, but it is necessary to give each datapoint a separate N or θ value that reports on the extent of the reaction.

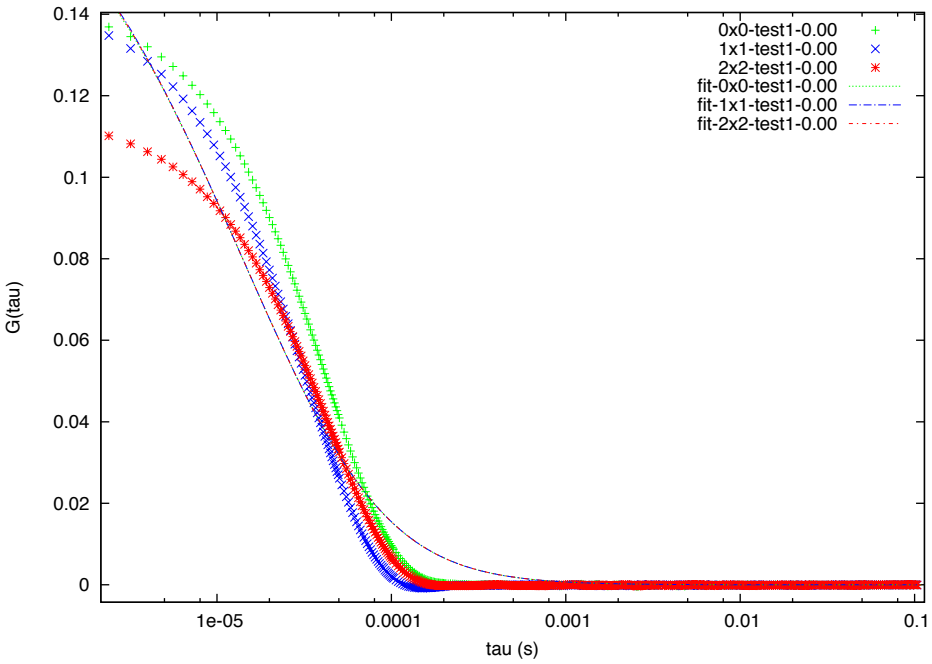

Figure 9: F3CS_LocalFit fits to FCS data (using an incorrect model) find an average fit to all three FCS curves when global fitting parameters are employed.

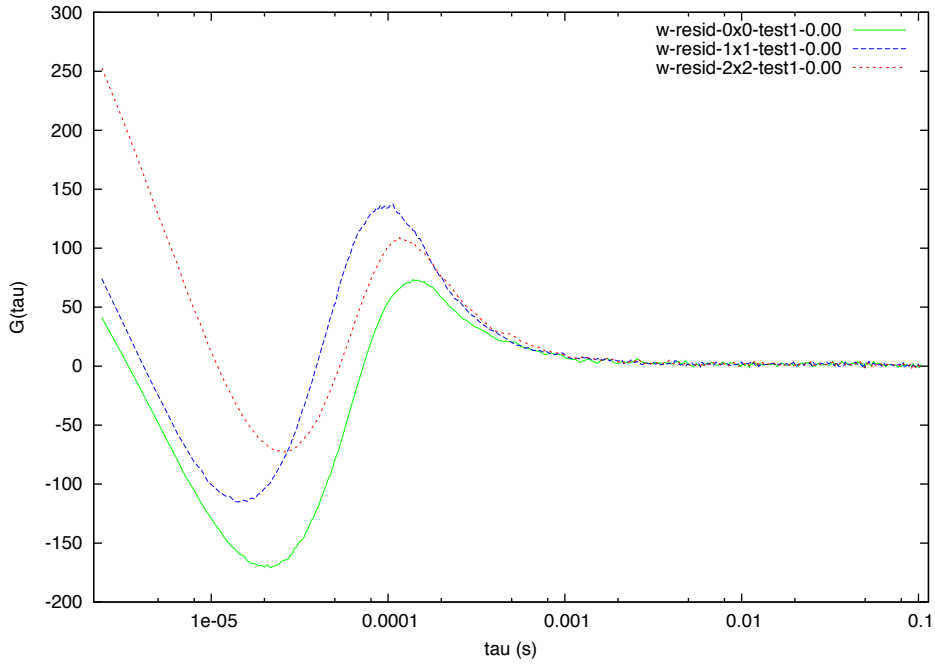

Figure 10: Weighted residuals from F3CS_LocalFit fits. The residuals are large but oscillate around zero, indicating that the fit program is doing the best it can with the wrong model.

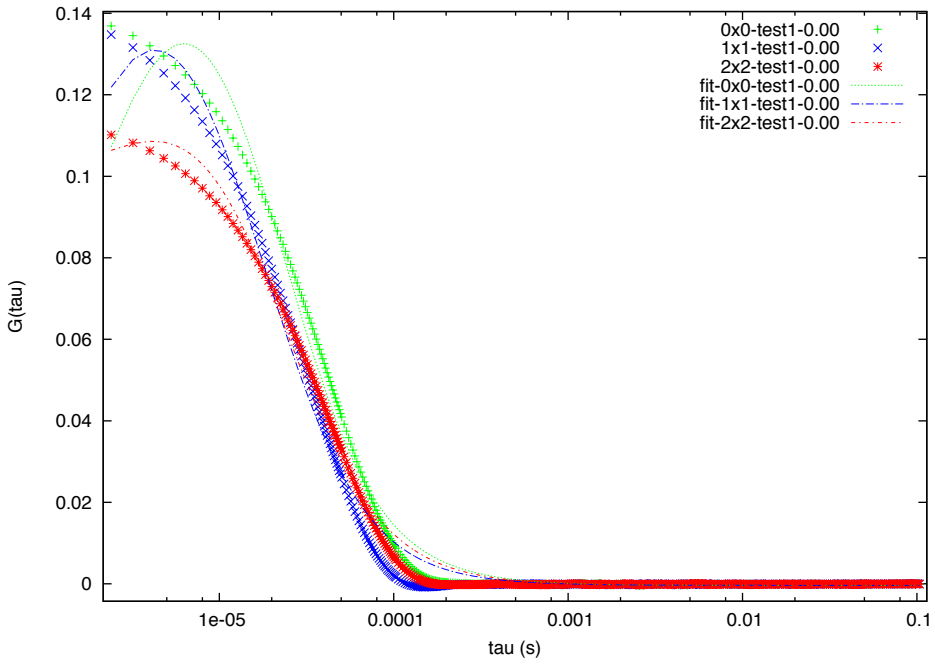

Figure 11: F3CS_LocalFit fits to FCS data obtained by using multiple fit parameters per curve. This fits data better than global fit parameters in Fig. 9.

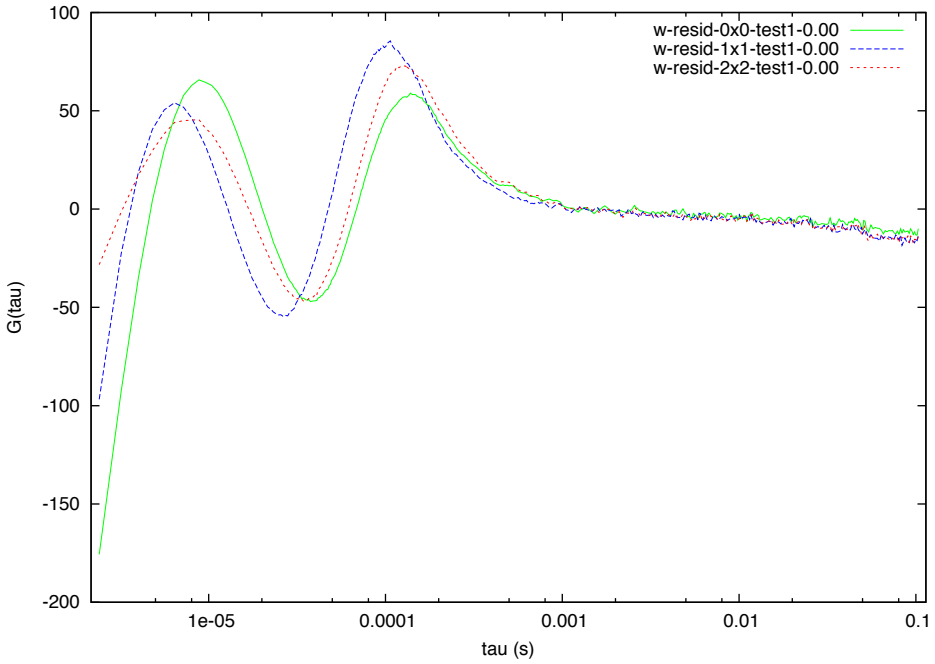

Figure 12: Weighted residuals from F3CS_LocalFit fits, corresponding to fits in Fig. 11.

3.8 Fitting data with F3CS_GlobalFit

F3CS data are complex and difficult to visualize, so the first function of F3CS_GlobalFit, "plotdata" is simply to plot primary data in a manageable way. Output from the commands below will give a blank gnuplot screen because the output is rather in the form of an EPS file (multiGGGt_test1.0.eps) that must be opened by an appropriate viewer / editor such as Illustrator (Fig. 13):

```
F3CS_GlobalFit input.global.txt tag 0 plotdata
gnuplot> load 'gnuplot_test1_GGGs.0.txt'
(then open multiGGGt_test1.0.eps using a program of your choice)
```

It is important to start the fit with an accurate guess, particularly because there are a large number of datapoints to fit and the fitting "landscape" is rugged enough that global minima are not always reached on the first attempt. Finding an accurate initial guess should be done by trial and error, with changes to the fitting parameters made to the input file that we used last time, "input.global.txt," and the results plotted using the second mode of F3CS_GlobalFit, the plotguess mode. Specifically, the guess file shows both the data and the model (Fig. 14) while the resids file shows the weighted residuals (Fig. 15).

```
F3CS_GlobalFit input.global.txt tag 0 plotguess
gnuplot> load 'gnuplot_test1_GGGs.0.0.guess.txt'
gnuplot> load 'gnuplot_test1_GGGs.0.0.resids.txt'
```

Once a satisfactory guess has been obtained, the data are fit using the "fitdata" mode of F3CS_GlobalFit. Again, the primary determinate of fit quality is the reduced χ^2 value, and in this case at least the fits are again going to be imperfect because the data do not match the model, and the reduced χ^2 value will end up around 400:

```
>F3CS_GlobalFit input.global.txt demo 0 fitdata
Fit Mode
 Status = success, Red.Chi^2 = 5128.80464240
 Status = success, Red.Chi^2 = 2399.88305332
...
 Status = success, Red.Chi^2 = 393.33456208
chisq/dof = 393.335
...
Overall Red. Chi Sq: 393.3346
Species 0
  N=     8.371/1.3e-01  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  30.00          T0=   -0.500          fracG=  0.330
Species 1
  N=     7.635/6.6e-02  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  30.00          T0=   -0.500          fracG=  0.330
Species 2
  N=    11.482/2.0e-01  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  30.00          T0=   -0.500          fracG=  0.330
Species 3
  N=     1.301/1.9e-01  tauf=  1.000          gamma=  0.350          w=   1.100
```

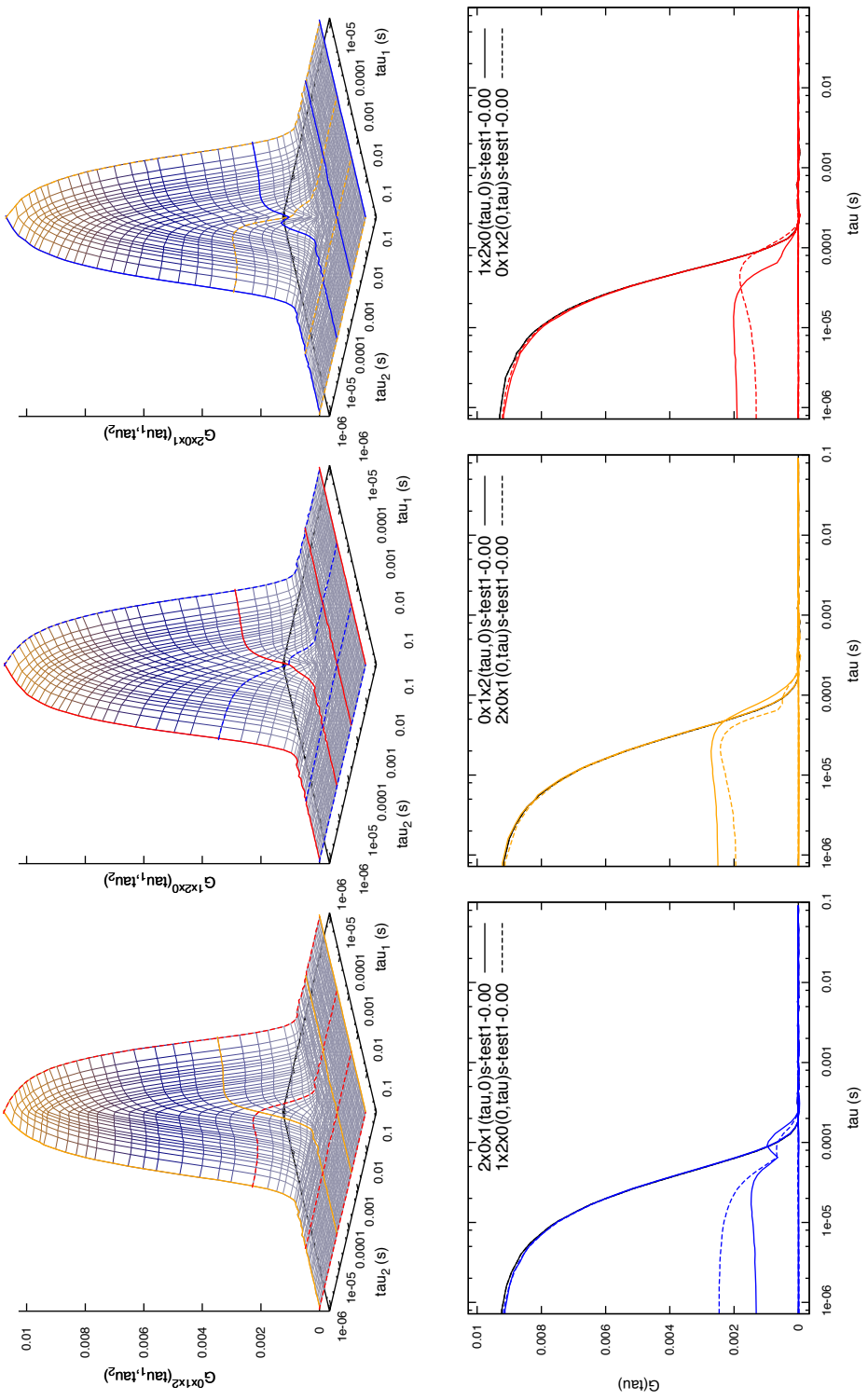

Figure 13: Displaying data with F3CS_GlobalFit.

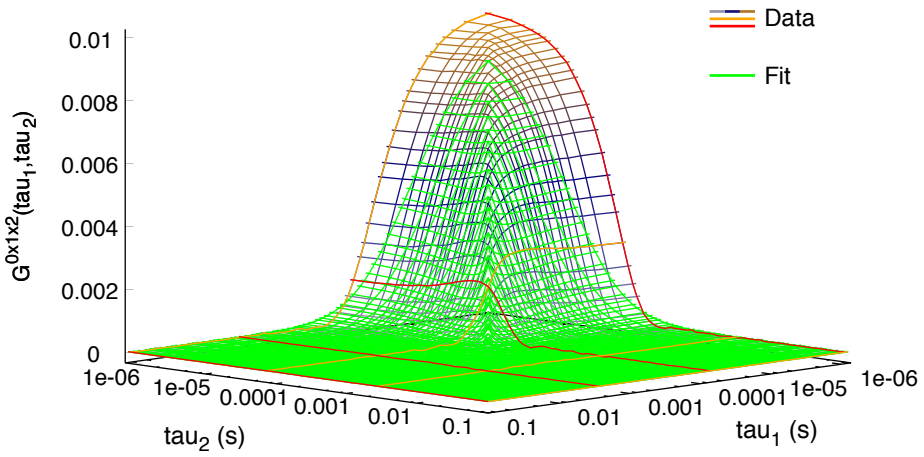

Figure 14: The initial fit function can be tuned using F3CS_GlobalFit in "plotguess" mode. (A) Data and fits for FCS and FCCS curves, along with their (B) weighted residuals.

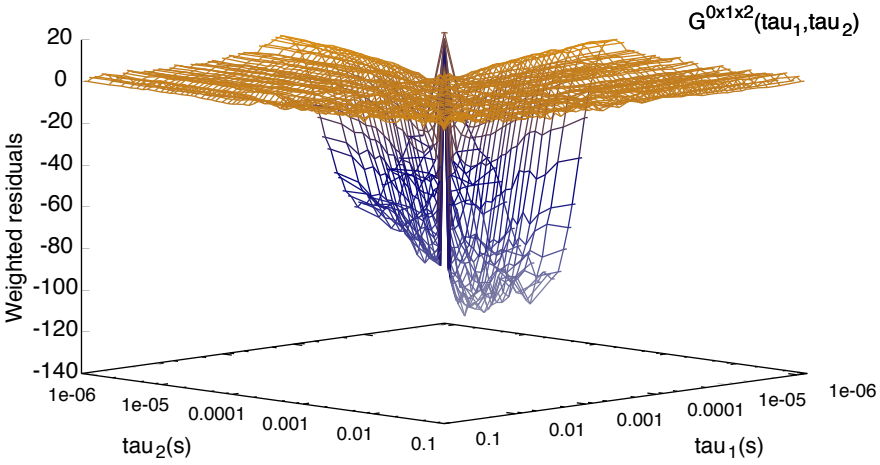

Figure 15: The initial fit function for F3CS can also be tuned using F3CS_GlobalFit in "plotguess" mode.

```
  tauD=  40.00          T0=   -0.500          fracG=  0.330
Species 4
  N=    -1.156/4.9e-02  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  40.00          T0=   -0.500          fracG=  0.330
Species 5
  N=     1.795/2.6e-01  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  40.00          T0=   -0.500          fracG=  0.330
Species 6
  N=    -2.337/2.5e-02  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  30.00          T0=   -0.500          fracG=  0.330
```

```
Species 7
  N=    -3.894/3.4e-02  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  30.00          T0=   -0.500          fracG=  0.330
Species 8
  N=    -5.168/6.0e-02  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  30.00          T0=   -0.500          fracG=  0.330
Species 9
  N=     1.816/1.9e-02  tauf=  1.000          gamma=  0.350          w=   1.100
  tauD=  30.00          T0=   -0.500          fracG=  0.330
q_0=1120.215/1.2e+01   bg_0=  1.000  ...
q_1=1373.215/9.7e+00   bg_1=  1.000  ...
q_2= 858.782/8.5e+00   bg_2=  1.000  ...
...
For Plotting: gnuplot> load 'gnuplot_test1_Gs.0.fits.txt'
For Plotting: gnuplot> load 'gnuplot_test1_GGGs.0.0.fits.txt'
...
For Plotting: gnuplot> load 'gnuplot_test1_GGGs.0.0.resids.txt'
...
For Plotting: gnuplot> load 'gnuplot_test1.0.allresids.txt'
...
Program completed.
```

The final fit parameters are displayed for all ten species analyzed here and like before there are several gnuplot scripts to facilitate data visualization, and their output can be seen in Figs. 16 - 18. The last gnuplot command displays the residuals for FCS, FCCS, F3CS and intensity data, flattened out into a straight line (Fig. 19). In addition to the gnuplot scripts shown above,

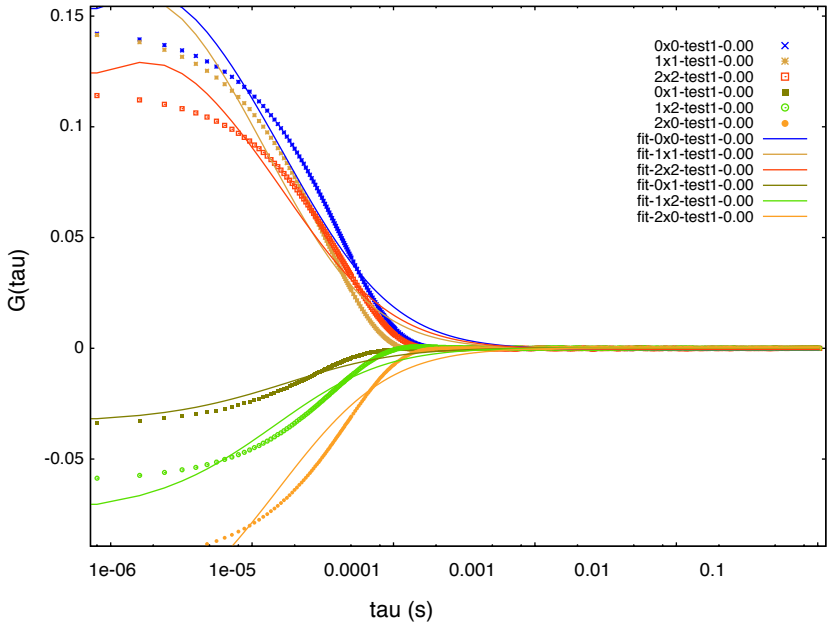

Figure 16: The FCS and FCCS portions of the global fits

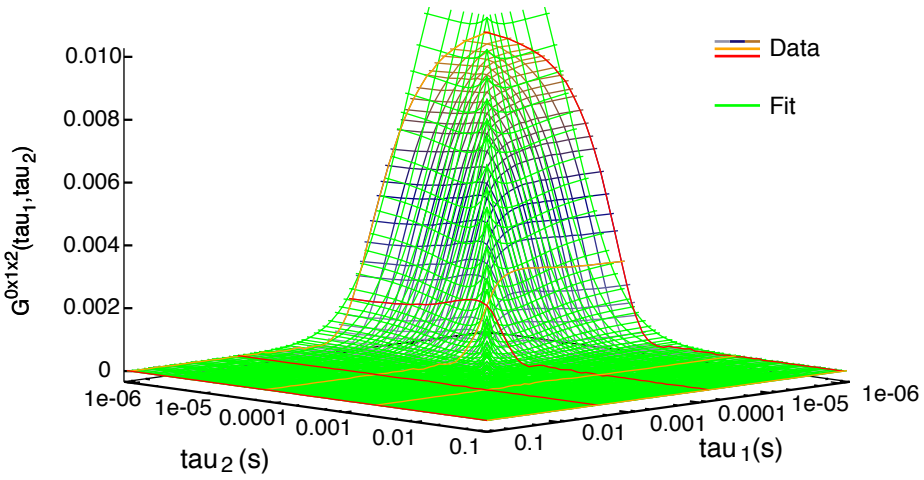

Figure 17: The F3CS portion of the global fits

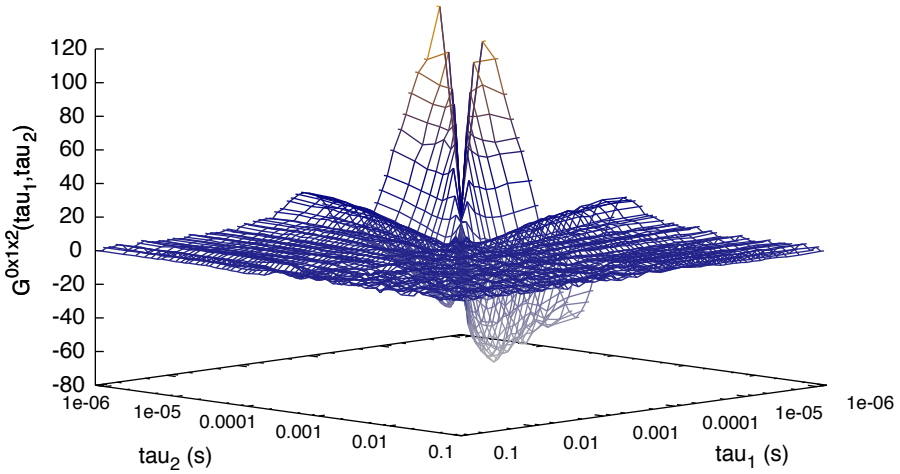

Figure 18: Weighted residuals for the F3CS portion of the global fits. Note that unlike Fig. 15, the fitted residuals oscillate around zero.

the fit parameters alone are output to a tab delineated file "gtaul.fit_params.dat," that can be read by common data analysis programs.

3.9 Fitting data with F3CS_GlobalFit_3D_Plot

F3CS data are difficult to display in general because they require three-dimensional plots and contain a large number of datapoints. The shaded plots afforded by the plplot library assist the viewer by giving a sense of tangibility to the 3D surfaces (e.g. Fig 6E in the main manuscript), and it is generally easier to get to interpret such plots instead of contour maps. F3CS_GlobalFit_3D_Plot generates such plots, using the same syntax and usage as F3CS_GlobalFit. These are output as a multi-page ps file, which can be converted into a

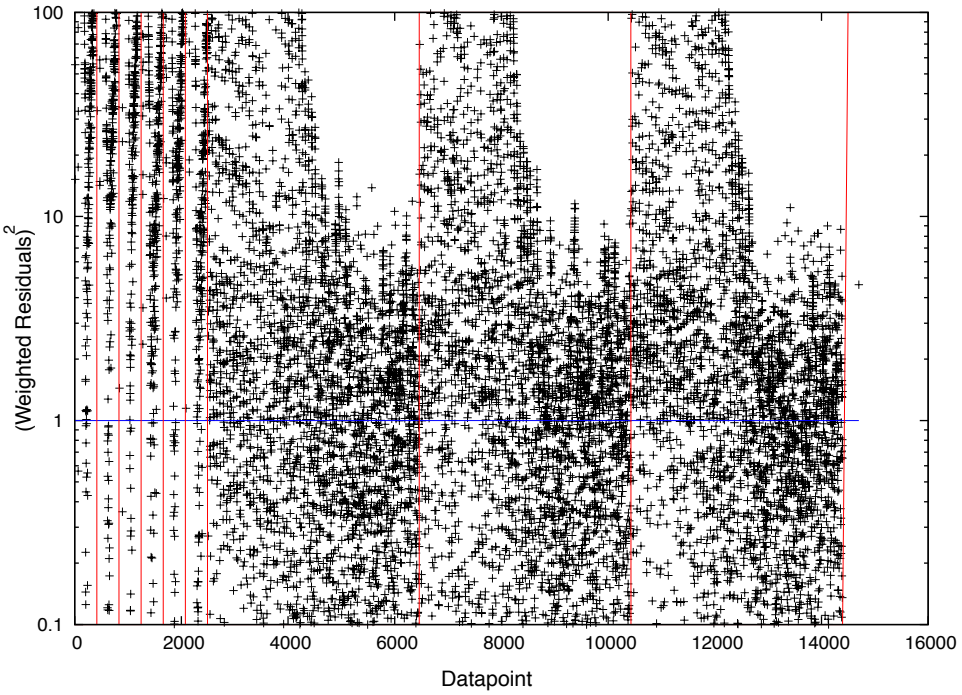

Figure 19: All squared weighted residuals from a F3CS_GlobalFit fit. From left to right: 3×FCS, 3×FCCS, 3×F3CS, 3× intensity (two intensity points are off the scale in this example). Note that 14,000+ datapoints were fit to a single model. These terms contribute equally to the overall reduced χ^2 statistic.

pdf by the utility ps2pdf, or by standard vector graphics programs. Note that a black layer occasionally is added to the graphics, this can simply by deleted to reveal the full plot beneath.

```
F3CS_GlobalFit_3D_Plot input.global.txt tag 0 fitdata
ps2pdf Shaded_test1.0.ps
```

3.10 Time reversal for detecting irreversible processes

The data generated for this tutorial arose from an irreversible process (Fig. 1), and as such the forward and reverse triple correlations are not necessarily identical. This can be examined by calculating the difference between forward and reverse correlations, $\Delta G(\tau_1, \tau_2)$. Reverse correlations, $G(-\tau_1, -\tau_2)$ have already been calculated in the tutorial, they are the default item to calculate using F3CS_AxBxG. To calculate forward correlations, $G(\tau_1, \tau_2)$, raw data in ".w3c" files are first copied to a set of ".r.w3c" files using F3CS_Reverser, which reverses the time-order of data as it copies them:

```
F3CS_Reverser _test1 1 6
```

Above, the 1 and 6 denote the numbers of the first and last data files to be reversed, and correspondingly six files are output. The reversed data are then run through the normal triple correlation routine by appending ".r" to the tag:

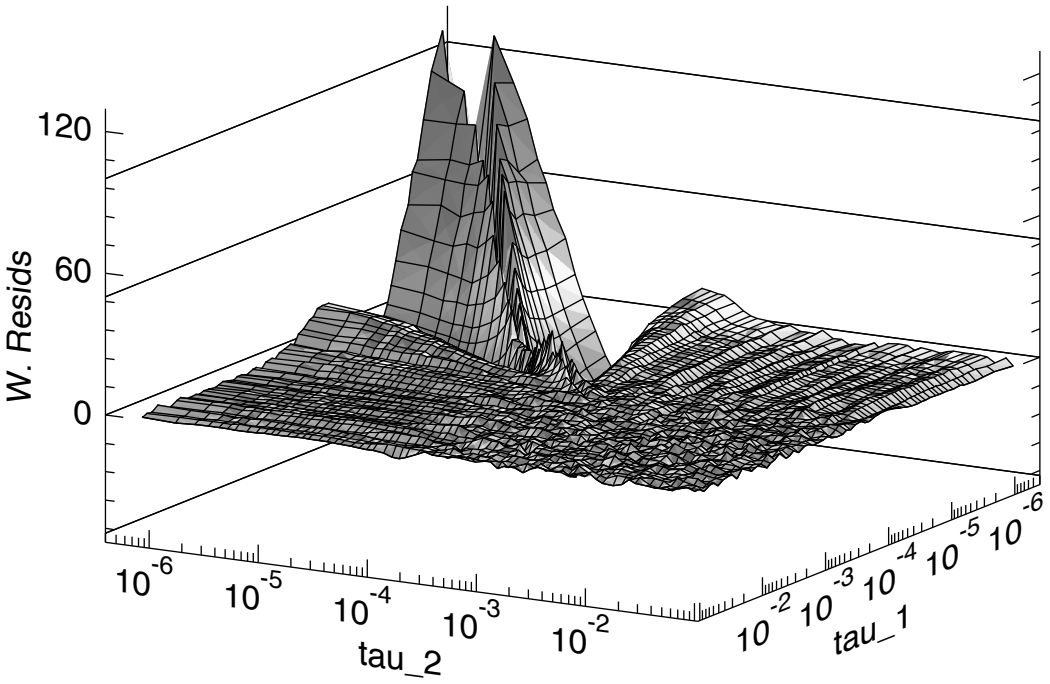

Figure 20: Shaded curves output by F3CS_GlobalFit_3D_Plot, replotting similar data as seen in Fig 18.

```
F3CS_AxBxG _test1.r 1 6 8
```

Like before, the correlation integrals are written to binary files, but this time with ".r" in the tag:

G_array_test1.r_1.bin2
G_array_test1.r_3.bin2
G_array_test1.r_5.bin2

using F3CS_Difference, which takes
and the output are additionally subjected to outlier rejection:

```
F3CS_Difference _test1 1 3.2 -1
```

As before, data are output in both binary format (bin_clean_test1.bin2) and text format (clean_GGGs_test1.dat), and a gnuplot script (samples_GGGs_test1.script) allows the user to quickly view the output (Fig. 21):

```
gnuplot> load 'samples_GGGs_test1.script'
```

No fit functions have yet been derived to fit $\Delta G(\tau_1, \tau_2)$ data, and thus there are no ways to (sensibly) fit these data using F3CS_GlobalFit or F3CS_LocalFit.

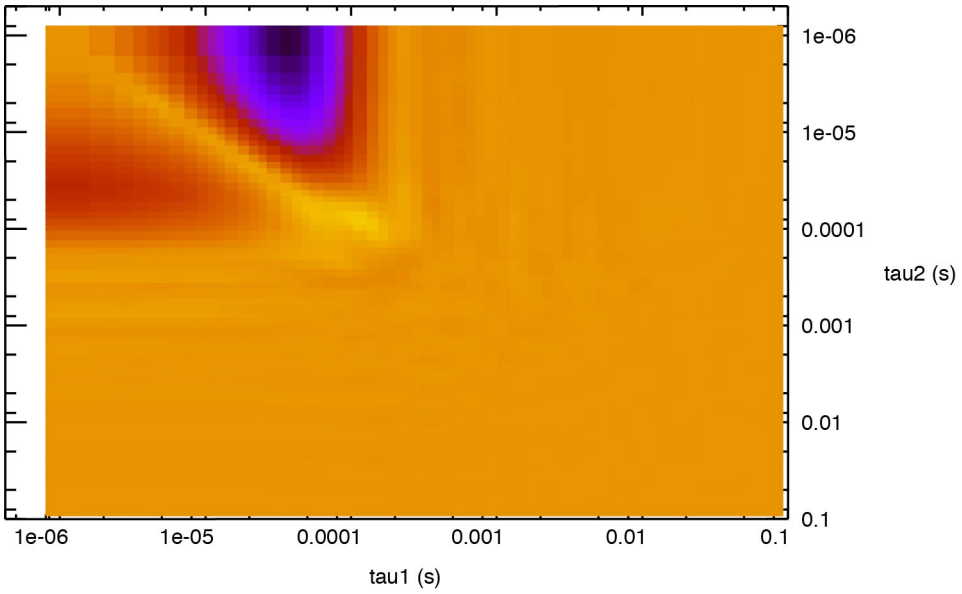

Figure 21: $\Delta G(\tau_1, \tau_2)$ analysis of the stochastic data. Non-zero values (non-gold) indicate areas where forward and backwards triple correlations are not identical, and the existence of non-zero values indicates that the process is irreversible on some level. With real samples, this type of analysis is useful to examine the degree of photobleaching.