pyFRET: An Open-Source Python Library for Analysis of Single Molecule Fluorescence Data

Author1[1], Author2[2], Author3[3,*]

1 Rebecca R. Murphy Department of Chemistry, University of Cambridge, Cambridge, UK

2 Magnus Kjaegaard Interdisciplinary Nanoscience Center, Aarhus, Denmark

3 Sophie E. Jackson Department of Chemistry, University of Cambridge, Cambridge, UK

4 David Klenerman Department of Chemistry, University of Cambridge, Cambridge, UK

∗ E-mail: dk10012@cam.ac.uk

Abstract

Single molecule Förster resonance energy transfer (smFRET) is a powerful experimental technique for studying the properties of individual biological molecules in solution. In recent years, significant progress has been made in developing new data collection methods, which significantly enrich enhance the information that can be gained from a smFRET experiment. However, adoption of smFRET techniques has so far been limited by a reliance on custom built software for data processing, which is independently developed and maintained by each research group in the field.

Here, we present pyFRET, an open source python package for the analysis of data from single-molecule fluorescence experiments from freely diffusing biomolecules. The package provides methods for the complete analysis of a smFRET dataset, from burst selection and denoising, through data visualisation and model fitting. We provide support for both continuous excitation and alternating laser excitation (ALEX) data analysis, including burst search algorithms and stochastic denoising. We also implement the recently developed Recurrence Analysis of Single Particles (RASP) algorithm for kinetic analysis of fluorescent subpopulations.

In this article, we demonstrate use of this package through analysis of a series of dual-labelled DNA duplexes, using both ALEX and continuous excitation data, to reproduce the characteristic sigmoidal FRET-eficiency curve. We also compare photon bursts from freely diffusing molecules with bursts from molecules flowed through the confocal volume, illustrating the effect of confocal dwell time on fluorescent emission.

pyFRET is available as a package downloadable from the Python Package Index (pyPI) under a three-clause BSD licence at `https://pypi.python.org/pypi/pyFRET/0.1.7.1`, together with links to extensive documentation and tutorials, including example usage and test data. Additional documentation including tutorials is hosted independently on ReadTheDocs, a dedicated documentation service (`https://readthedocs.org/projects/pyfret/`).

Author Summary

Introduction

Förster resonance energy transfer (FRET) [1] is a non-radiative energy transfer process that can occur between chromophoric molecules. The degree of energy transfer is dependent on the inter-fluorophore distance, allowing FRET to be used as a "molecular ruler" [1], to determine intramolecular distances. Since FRET was first used to measure the distance between two fluorescent dyes on individual molecules bound to a surface [2], single-molecule FRET (smFRET) has become a popular tool to investigate the structure and dynamics of biomolecules, both on a surface and diffusing freely in solution [3–5].

In a confocal smFRET experiment, biological molecules are labelled with two fluorescent dyes. The emission spectrum of the donor dye (D) is chosen to overlap with the excitation spectrum of the acceptor

(A). When the donor and acceptor are sufficiently close in space, exciting the donor dye results in FRET and fluorescent emission from the acceptor dye. The FRET efficeicny, E, which descrbes the proportion of excitation energy transferred from the donor to the acceptor, depends on the distance, r between the two dyes (Eq. 1) and R_0, the Förster distance, a dye dependent constant that describes the dye separation at which 50% energy transfer is achieved (Fig. **??** C)

$$E = \frac{1}{1 + (\frac{r}{R_0})^6} \tag{1}$$

Consequently, the distance between the two fluorophores can be determined from the ratio of donor and acceptor photons emitted during an excitation event (Eq. 3).

Experimentally, a collimated laser beam, is used to illuminate an extremely dilute solution of labelled molecules. When a labelled molecule diffuses through the laser beam, the donor dye is excited and photons are emitted from both donor and acceptor dyes. Emitted photons are collected through the objective and separated into donor and acceptor streams for collection and analysis (Fig. **??** A, B).

For a single fluorescent burst, the FRET efficiency, E, can be calculated as (Eq **??**):

$$E = \frac{n_A}{n_A + \gamma \cdot n_D} \tag{2}$$

for n_A and n_D detected acceptor and donor photons respectively and γ an experimentally determined instrument-dependent correction factor. During the course of a smFRET experiment, several thousand fluorescent bursts are collected and used to construct FRet Efficiency histograms. These histograms can be used to identify populations of fluorescent species [2], typically by fitting with multiple gaussian distributions.

Analysis of smFRET data involves several computational challenges. Firstly, photons emitted by a fluorescent molecule diffusing through the excitation volume must be identified against a noisy background. Secondly, identified bursts must be denoised, including removal of background auto-fluorescence and donor-acceptor crosstalk. Fluorescent bursts that are distorted by photobleaching or other photophysical artifacts should be identified and excluded. Multiple methods of burst selection and analysis have been developed and applied to the analysis of smFRET data [5–13]. However, software for analysis of smFRET data has thus far been developed on an ad hoc basis, with individual groups preparing and maintaining their own analysis scripts, leading to problems typical of research programming projects [14, 15].

Here we present pyFRET, an open-source python library, for the analysis of smFRET data. To our knowledge, this is the first open source software ever released by the smFRET research community. pyFRET is a small library that provides a toolkit facilitating all key steps in analysis of smFRET data: burst selection; cross-talk subtraction and burst denoising; data visualisation; and construction and simple fitting of FRET efficiency histograms. In providing this toolkit to the smFRET research community, we hope to facilitate the wider adoption of smFRET techniques in biological research as well as providing a framework for open communication about and sharing of data analysis tools.

Design and Implementation

Code Layout and Design

pyFRET provides four key data structures (classes) for manipulation of smFRET data. The FRET data object describes two fluorescence channels, corresponding to time-bins containing photons collected from donor (the donor channel, D) and acceptor (the acceptor channel, A) fluorophores. The ALEX data object describes four fluorescence channels, corresponding to the four temporal states in a smFRET experiment using Alternating Laser Excitation (ALEX), namely the donor channel when the donor laser is switched on (D_D); the donor channel when the acceptor laser is switched on (D_A); the acceptor channel

when the donor laser is on (A_D); and the acceptor channel when the acceptor laser is on (A_A). These data channels are implemented as numpy arrays, allowing efficient computations and selection operations. The data structure can readily be expanded to include data from more detectors, which is needed in e.g. three-colour or anisotropy measurements.

Two similar classes are used for fluorescence bursts identified using the burst search algorithms. In addition to the donor and acceptor channels, the FRET bursts class type holds three additional arrays, giving the first and last bin of each burst, and the duration of each burst. These three new attributes are similarly present in the ALEX bursts object, in addition to the four fluorescent channels in the ALEX data object. In addition to the methods present for the simple FRET and ALEX objects, the burst data objects also implement methods to plot burst duration and to analyse recurrent bursts (RASP).

The data analysis workflow is illustrated in Fig. **??** Following initialization of data objects, background subtraction, event selection, cross-talk correction and calculation of the FRET efficiency can each be performed with a single line of code. Simple but high-quality figures can be generated in a single step.

Simple Event Selection and Denoising

FRET Data

In the most simple smFRET experiment, fluorescently labelled molecules are excited by a laser that will excite the donor dye; all photons reaching the detectors during data acquisition binned as they are received into time-bins of length similar to the expected dwell-time of a molecule in the confocal volume (for freely diffusing molecules, a bin-time of 1 ms is typically used). Event selection then simply involves identifying time-bins that contain sufficient photons to meet a specified criterion. Two thresholding criterion are in common use. AND thresholding selects time bins for which $n_D > T_D$ AND $n_A > T_A$ for n_D and n_A photons in the donor and acceptor channels respectively, and T_D and T_A the donor and acceptor thresholds. In a similar manner, SUM thresholding considers the sum of photons observed in the donor and acceptor channels, selecting time bins for which $n_D + n_A > T$. These thresholding techniques can be implemented using a single call to a pyFRET function:

```
# Simple thresholding

# define thresholds
Td = 20 # donor threshold
Ta = 20 # acceptor threshold
T = 50  # combined threshold

# AND thresholding
data.threshold_AND(Td, Ta)

# SUM thresholding
data.threshold_SUM(T)
```

Following event selection, a simple method of denoising is to subtract from each selected event the average background autofluorescence observed in each channel and the average cross-talk between the two channels:

```
# Simple denoising

# removing autofluorescence
auto_donor = 0.5   # donor autofluorescence
auto_acceptor = 0.3 # acceptor autofluorescence
my_data.subtract_bckd(auto_donor, auto_acceptor)
```

```
# removing cross-talk
cross_DtoA = 0.05 # fractional cross-talk from donor to acceptor
cross_AtoD = 0.01 # fractional cross-talk from acceptor to donor
my_data.subtract_crosstalk(cross_DtoA, cross_AtoD)
```

This is the simplest method for event selection and denoising. However, it has several limitations. In particular, simple subtraction of constant values can lead to unphysical artifacts such as negative and fractional photon counts. Furthermore, the simple thresholding criteria for event selection are known to be biased [8], so can distort downstream data analysis.

ALEX Data

A more sophisticated smFRET experiment uses Alternating Laser Excitation (ALEX) during data acquisition. In this method, the diffuisng fluorescent molecules are subjected to excitation from both two lasers in rapid alternation [9]. One laser excites the donor fluorophore and the other can directly excite the acceptor fluorophore. The alternation of the laser excitation is fast on the timescale of molecular dwell-time in the confocal volume, allowing a single fluorescent molecule to receive multiple cycles of donor-acceptor direct excitation.

Instead of the two photon streams – donor and acceptor photons – observed in a simple smFRET experiment, in a confocal ALEX experiment, there are four streams: $F_{D_{ex}}^{D_{em}}$, $F_{D_{ex}}^{A_{em}}$, $F_{A_{ex}}^{D_{em}}$, $F_{A_{ex}}^{A_{em}}$. $F_{D_{ex}}^{D_{em}}$ and $F_{D_{ex}}^{A_{em}}$, respectively donor and acceptor emission during donor excitation, are analogous to the two original donor and acceptor photon streams in a smFRET experiment. $F_{A_{ex}}^{A_{em}}$ records acceptor emission during direct acceptor excitation, whilst $F_{A_{ex}}^{D_{em}}$ records donor emission during acceptor excitation. The presence of these extra channels provides additional information about the labelling state of molecules giving rise to fluorescent bursts, allowing exclusion of bursts from molecules lacking one of the FRET labels and bursts where one of the labels was bleached. Selection based on direct excitation of both fluorophores also removes the biases caused by simple AND or SUM thresholding.

pyFRET implements ALEX event selection as described in the original publication [16]. In brief, bursts are initially selected using a selection criterion based on the total number of photons emitted during donor and acceptor excitation: $F_{D_{ex}}^{D_{em}} + F_{D_{ex}}^{A_{em}} > T_D$ AND $F_{A_{ex}}^{A_{em}} > T_A$.

Following this initial event selection, a second selection step is performed, based on the ratio of photons emitted during donor and acceptor excitation periods. The photon stoichiometry, S is calculated for each burst as:

$$S = \frac{F_{D_{ex}}^{D_{em}} + F_{D_{ex}}^{A_{em}}}{F_{D_{ex}}^{D_{em}} + F_{D_{ex}}^{A_{em}} + F_{A_{ex}}^{A_{em}}} \tag{3}$$

Events for which the stoichiometry is either very close to one or very close to zero, indicating presence of only the donor or acceptor fluorophore respectively can be excluded using a second event selection criterion: $S_{min} < S < S_{max}$

Following event selection, remaining bursts can be corrected for photon leakage and direct excitation contributions. A two-dimensional scatter plot of FRET efficiency E and stoichiometery S is then produced, including one-dimensional histograms of both E and S.

pyFRET allows each of these steps to be performed separately, however they can also be combined into a single step combining event selection, denoising, FRET efficiency calculation and plotting:

```
# Simple ALEX analysis

g_factor = 0.95 # instrumental gamma factor
S_min = 0.2   # min accepted value of S
```

```
S_max = 0.8    # max accepted value of S
filepath = "path\to\my\file"
filename = "scatter_plot"
ALEX_data.scatter_hist(S_min, S_max, gamma=g_factor, save=True, filepath=filepath,
    imgname=filename, imgtype="png")
```

Example ALEX analysis scripts can be found in the supplementary information.

Burst Search Algorithms

Although using time-bins that are matched to the dwell-time of molecules in the confocal volume is simple, it is not ideal, as some bursts will be split over several bins, so may be counted as separate events, or not considered for analysis. More sophisticated event selection algorithms, typically called burst search algorithms [8], bin photons on a time-scale much shorter than the typical dwell time in the confocal volume and then scan the resultant photon stream for bursts of a specified duration and brightness. pyFRET implememts both All Photons Burst Search (APBS) and a Dual Channel Burst Search (DCBS) algorithms for both ALEX data and for simple FRET data, as originally described [8].

In APBS burst search for FRET data, photons from both donor and acceptor channels are considered together. A burst is defined according to three constants: T, the averaging window; M, the minimum number of photons within window T; and L, the minimum total number of photons required for an identified burst to be retained. These three values are used in a two-step process for burst identification.

Firstly, "the start (respectively, the end) of a potential burst is detected when the number of photons in the averaging window of duration T is larger (respectively, smaller) than the minimum number of photons M." [8]. In pyFRET, this initial search is performed using the convolve method from numpy [17] to provide a running sum across windows of T time-bins. Following initial burst identification, a burst is retained if it contains more than L photons [8].

The DCBS burst search is similar, but considers the donor and acceptor channels separately. For a burst to be accepted in DCBS, both channels must simultaneoulsy meet the running sum criterion, allowing exclusion of single colour bursts and bursts where one fluorophore bleaches.

The burst search algorithms implemented for ALEX data work in a similar manner. In the ALEX APBS method, bursts are identified by considering the total number of fluorescent photons $F_{total} = F_{D_{ex}}^{D_{em}} + F_{D_{ex}}^{A_{em}} + F_{A_{ex}}^{A_{em}}$ in each time bin. Bursts are identified where the running sum (calculated using the F_{total} photon stream) in the averaging window T exceeds M. The DCBS method considers donor excitation photons $F_{D_{ex}}^{D_{em}} + F_{D_{ex}}^{A_{em}}$ separately from photons emitted during direct acceptor excitation $F_{donor} = F_{A_{ex}}^{A_{em}}$, requiring that the running sum exceeds M for both F_{donor} and $F_{A_{ex}}^{A_{em}}$.

Example code for running a burst search algorithm is shown below:

```
# Burst search using FRET data

# required parameters
T = 50          # time window (bins)
M = 50          # first threshold
L = 60          # second threshold

# calling APBS algorithm
bursts_APBS = FRET_data.APBS(T, M, L)
```

RASP: Recurrence Analysis of Single Particles

A recent innovation in confocal smFRET is Reccurrence Analysis of Single Particles (RASP) [18]. RASP determines subpopulation interconversion kinetics by using the fact that fluorescent bursts ocurring within a small time window have a higher probability of being generated by the same molecule diffusing back through the confocal volume than from two independent fluorescent events.

RASP is a two-step process. First, initial bursts (b_1) with a FRET efficiency E_{b1} within some defined range $\Delta(E_{b1})$ are identified. Secondly, bursts (b_2) occurring within a time interval (called the recurrence interval) $T = (t_1, t_2)$ of b_1 are identified. Analysis of the distribution of FRET efficiencies in b_2, the population of recurrent bursts, provides information about the interconversion rate between subpopulations. The rates constants of interconversion can be extracted by fitting the relative subpopulations as a function of interval time.

pyFRET implements RASP using array masking, to allow efficient selection of relevant bursts. RASP can be called in a single step from a FRET bursts or ALEX bursts object, and a loop can readily be made to repeat the process at different time intervals:

```
# RASP

# initial E range: 0.4 < E < 0.6
Emin = 0.4
Emax = 0.6

# Time interval for re-occurrence
# given in number of bins
Tmin = 1000
Tmax = 10000

# selecting re-occurring bursts
recurrent_bursts = bursts_APBS.RASP(Emin, Emax, Tmin, Tmax)

# histogram of re-occurring bursts
recurrent_bursts.build_histogram(filepath, csvname, gamma=g_factor)
```

Compatibilities

pyFRET is written in Python. Both python 2 (v2.7) and python 3 (v3.3) are supported. pyFRET requires three further python libraries, namely numpy and scipy for data manipulation, and matplotlib for data visualisation. We recommend using the free Anaconda package bundle (`https://store.continuum.io/cshop/anaconda/`) to install these dependencies. Detailed installation instructions can be found in the pyFRET documentation (`http://pyfret.readthedocs.org/en/latest/tutorial.html#installing-pyfret`). pyFRET was written and tested in a Linux environment. However, it was written to be platform independent and has also been used successfully on both Apple and Windows machines.

The lack of Open Source software in the smFRET community has led to a proliferation of esoteric file-types used for data collection and storage. To make pyFRET as usable as possible for a wide range of smFRET researchers, we provide file parsers for simple .csv and .txt file formats, as well as our custom binary format. The pyFRET data structures can be initialised using simple python arrays of time-binned photons, for users whose file format is not currently supported. The tutorial and supplementary information provide example scripts for parsing common filetypes into pyFRET objects.

Experimental Methods

Benchmarking Using Simulated Datasets

To test the ability of the fitting algorithms to distinguish fluorescent populations with similar FRET efficiencies, we generated simulated datasets consisting of mixtures of FRET bursts with a known FRET efficiency and population size. These bursts were then fitted using pyFRET's two component mixture model and the results compared to the known input FRET efficiencies and population sizes. The code used to generate and fit these datasets can be found XXXX.

Showcasing the simple FRET Algorithms

We tested the pyFRET library using DNA duplexes dual-labelled with Alexa Fluor 488 and Alexa Fluor 647. The duplex sequences and labelling sites are shown in Table **??**. Labelled duplexes were diluted to a concentration of 50 pM in TEN buffer (10 mM Tris, 1mM EDTA, 100 mM Nacl), pH 8.0, containing 0.0001 % Tween-20. FRET data were collected for 15 minutes using continuous excitation at 488 nm at a power of 80 mW. Collected photons were binned online in intervals of 1 ms and stored in files of 10000 bins. ALEX data were collected for 15 minutes using alternating excitation at 488 and 640 nm, with respective laser powers of 80 and 70 mW, and a modulation rate of 0.1 ms, a dead-time of 0.1 μs and a delay compensation of 3 μs. ALEX data were then binned in intervals of 1 ms. The scripts and configuration files used to analyse these data using pyFRET can be found in the supplementary material.

Showcasing the Burst Search Algorithms

To test the burst search algorithms, we collected data under both ALEX and FRET conditions. Data were collected for 10 minutes, using laser powers of 130 μW in both donor and acceptor excitation. For FRET data collection, laser illumination by the donor laser was continuous and the data were binned online into time bins of 50 μs, roughly 5 % of the duration of the average burst from a freely diffusing molecule. The acceptor laser was not used. For ALEX data collection, photons were similarly binned online into time bins of 50 μs length, but the laser modulation rate was increased to 50 μs, with a dead-time of 0.1 μs and a delay compensation of 3 μs, corresponding to 2 modulations per short time bin.

To evaluate the effect of bin-time on performance of the burst search algorithms, a further dataset was collected using FRET excitation on the 6 bp duplex. For this dataset, data were collected for 10 minutes using the 488 nm laser at 130 μW. Data were binned into time bins of 10 μs, allowing for re-binning into longer time-bins as required.

Performance Analysis Using Mixtures of DNA Duplexes

To test the accuracy of the burst search algorithms, we collected ALEX data from mixtures of two DNA duplexes at known concentration ratios. The 4, 8 and 12 bp duplexes were used. Samples were prepared containing a mixture of two duplexes in TEN buffer with a total DNA concentration of 80 pM, divided in either a $3:1$ ratio (high concentration 60 pM, low concentration 20 pM) or a $1:1$ ratio (both components 40 pM). Data were collected under ALEX excitation for 10 minutes, using the conditions described above.

Testing the RASP Algorithm

To test the RASP Algorithm, a 1:1 mixture of 6 bp and 12 bp duplex was prepared to a total DNA concentration of 50 pM. A 400 μL aliquot of the dilute solution was placed in one chamber of a lidded, chambered coverslide (LabTex) to reduce evaporation during the measurement. FRET data were collected

for 600 minutes using continuous excitation at 488 nm at a power of 140 μW. Collected photons were binned online in intervals of 50 μs and stored in files of 1000000 bins.

Results

Evaluating Performance with DNA Duplexes

Simple Algorithms We tested the pyFRET library using DNA duplexes dual-labelled with Alexa Fluor 488 and Alexa Fluor 647. The duplex sequences, dye attachment sites and dye-dye separations are shown in Tables 1 and **??**. Event selection and denoising, calculation of FRET efficiency and the plotting and fitting of FRET efficiency histograms were performed using pyFRET. The results, shown in Fig. **??** for FRET data and Fig. **??**, demonstrate that even using the simplest event selection and denoising techniques, pyFRET is able to effectively fit histograms from single FRET populations (Fig. **??** and Fig. **??** A - E), to reproduce the characteristic sigmoidal FRET efficiency curve (Fig. **??** and Fig. **??** F).

Burst Search Algorithms We tested the pyFRET burst search algorithms using data collected from the same DNA duplexes but using a shorter bin-time. Sample results, from both APBS and DCBS analysis of FRET and ALEX data from the 6 bp duplex are shown in Fig. **??**. The characteristic sigmoidal FRET efficiency curves generated from DCBS analysis of all five duplexes are shown in Fig. 7 A (FRET data) and B (ALEX data). SAY SOMETHING ABOUT THE ZERO PEAK IN FRET APBS.

Evaluating the Burst Search Algorithms

In addition to demonstrating the functionality of the burst search algorithms, we have performed a comprehensive analysis of the impact of bin-time, threshold and detection window on the performance of the burst search algorithms. To our knowledge, this is the first time that such an analysis has been performed, and hence provides useful insight to other researchers depending on the performance of these algorithms.

Burst search algorithms were developed as an improvement to the simple thresholding technique, designed to reduce inaccuracies caused by photobleaching and by long bursts being split over multiple time-bins. To assess the improvement in data quality as a result of using the burst search algorithm, we collected data from the 6 bp duplex using time-bins of 10 μs, which could then be re-binned into longer time-bins for comparative analysis. Analysis was performed using both APBS and DCBS burst search algorithms on FRET data. For APBS, thresholds of $M = 20$ and $L = 20$ were used; the thresholds used for DCBS were $M = 10$ and $L = 10$. As the bin-time was varied, the minimum burst duration T, given in number of bins, was also varied, to keep the minimum burst duration to a constant time of 1 ms. The bin lengths and their corresponding value of T are shown in Table **??**. The results, shown in Fig. 8 A (DCBS) and B (APBS) are surprising. When the minimum time-interval for burst detection is not varied, the performance of the algorithm is essentially unaffected by the bin-times used: for both APBS and DCBS algorithms, the resultant FRET efficiency histograms are extremely similar, whether many short bins or a few long bins are searched.

Secondly, we evaluated the effect of the search window T on burst search performance. For a fixed bin-time of 50 μs, we varied the required burst duration T between 100 μs (2 bins) and 1000 μs (10 bins). The results, shown in Fig. 8 C for the DCBS algorithm are again surprising. Across all values tested, the shape of the FRET efficiency histogram is unaffectd by the size of the burst search window. The peak areas are also relatively unaffected by the search window size: the very shortest window of 100 μs retains only the brightest bursts, resulting in a slightly reduced peak area; other than this the number of detected events is not significantly altered.

Finally, we evaluated the effect of the thresholds M and L on the performance of the DCBS algorithm for both FRET and ALEX data using the high-FRET 6 bp duplex. We systematically varied the thresholds used in burst search analysis, then evaluated their effect on the fitted peak area and FRET efficiency. The results, shown in Fig. 9, display several interesting features. Firstly, the decline in peak area with increased threshold is striking for both FRET (Fig. 9 C) and ALEX (Fig. 9 D) data. Secondly, increasing the thresholds systematically reducuces the calculated FRET efficiency for the FRET dataset (Fig. 9 A). This is caused by the DCBS algorithm selecting against bursts that have a low donor count, as they do not meet the initial threshold M. This effect is not seen in DCBS on ALEX data (Fig. 9 B), as events are selected based on photons emitted during direct donor and acceptor excitation, so there is no bias towards intermediate FRET efficiencies.

We conclude this section with a number of recommendations. Firstly, we note that, when used on FRET data, the APBS and DCBS burst search algorithms perform in an analogous manner to simple AND and SUM thresholding. Consequently, they retain the well-known disadvantages of these thresholding methods: specifically, APBS retains a zero-peak caused by donor-only labelled molecules, whereas DCBS is biased against extreme FRET efficiencies, distorting the FRET efficiency histogram. ALEX data, does not display these biases, as the direct excitation of both fluorophores allows events to be selected independently of their FRET efficiencies. Consequently, ALEX is a superior technique and should be used when available.

Secondly, we note the importance of thorough evaluation of analysis tools. The burst search algorithms were developed as an improvement on simple thresholding analysis. However, our evaluation of burst search performance fails to provide evidence of significant performance enhancement. For both FRET and ALEX (XXX NEED TO INCLUDE THIS), performance of the burst search algorithm is not reduced when the search window is reduced to a single bin (functionally equivalent to the simple threshold), indicating equivalent performance of burst search and simple thresholding. Furthermore, as well as being computationally more expensive than simple thresholding, burst search has the significant disadvantage of generating much larger datasets, as an experiment of the same duration is split into smaller time intervals. This has the concommitant disadvantages of requiring more storage space and more processing time during analysis. We note that our implementation of DCBS and APBS was taken from the description in Nir et al. [8]. It is possible that this description was incomplete, and that an unreported addition to the algorithm as described does indeed enhance performance. However, in the absence of these details, for the reasons described above, we cannot recommend the adoption of burst search over simple thresholding.

Benchmarking the RASP Algorithm

We benchmarked RASP using data collected from a 1:1 mixture of 6 bp (high FRET) and 12 bp (low FRET) duplexes over a period of 10 hours (600 minutes). The parameters used in RASP analysis are shown in Table 4. RASP allows selection and analysis of bursts that occur within a time interval ΔT after bursts with FRET efficiency in the range $E_{min} - E_{max}$. We demonstrate the correct performance of the RASP algorithm by analysing bursts that occur in 1 ms intervals, centred at the stated times, following a high FRET burst. The resultant FRET histograms are shown in Fig. 10. At short recurrence intervals ($T_{max} < 8$ ms), not only do the great majority of events show a high FRET efficiency, but the number of events is greatly increased compared with longer recurrence intervals, demonstrating the enrichment of these events by molecules that have diffused back into the confocal detection volume.

Availability and Future Directions

pyFRET is available to download from PyPI under an open source BSD licence from the Python Package Index (`https://pypi.python.org/pypi/pyfret0.1.0`). Documentation can also be found here, whilst

a more extensive tutorial, including example scripts, can be found on our website (`http://rrm33.user.srcf.net/`) or in the Supplementary Information.

pyFRET currently provides basic tools for burst selection and denoising, based on simple thresholding and noise subtraction techniques. We are aware that more sophisticated methodologies exist and are currently working to produce stochastic denoising algorithms [12]. We have also developed a novel analysis method based on Bayesian statistics [19], for which source code is available (`https://bitbucket.org/rebecca_roisin/fret-inference`) and which we intend to fold into the pyFRET library. We are also working to increase support for the wide variety of file formats that result from custom-built data collection hardware.

smFRET is a fast-developing and active research field and we are keen to support scientific progress through development of high-quality usable software. We are keen to work with others to enable their use of and contribution to the pyFRET library. We welcome requests for custom analysis requirements and are happy to support others who wish to contribute additional code to the pyFRET infrastucture.

The data from DNA duplexes used in evaluating pyFRET can be found in the DataDryad online repository: XXX DOI XXX. iPython Notebooks that reproduce the analysis performed here are also included.

Acknowledgments

RRM would like to thank BBSRC for PhD funding.

References

1. Forster T (1948) Zweischenmolekulare energiewanderung undfluoreszenz. Annalen der Physik 2: 55–75.

2. Ha T, Enderle T, Ogletree DF, Chemla DS, Selvin PR, et al. (1996) Probing the interaction between two single molecules: Fluorescence resonance energy transfer between a single donor and a single acceptor. Proc Natl Acad Sci USA 93: 6264-6268.

3. Haran G (2003) Single-molecule fluorescence spectroscopy of biomolecular folding. J Phys: Condens Matter 15: R1291-R1317.

4. Schuler B, Lipman EA, Eaton WA (2002) Probing the free-energy surface for protein folding with single-molecule fluorescence spectroscopy. Nature 419: 743-747.

5. Weiss S (2000) Measuring conformational dynamics of biomolecules by single molecule fluorescence spectroscopy. Nat Struct Mol Biol 7: 724-729.

6. Deniz AA, Lawrence TA, Dahan M, Chemla DS, Schultz PS, et al. (2001) Ratiometric single-molecule studies of freely diffusing molecules. Annu Rev Phys Chem 52: 233–253.

7. Gell C, Brockwell D, Smith A (2006) Handbook of single molecule fluorescence. Oxford: Oxford University Press.

8. Nir E, Michalet X, Hamadani KM, Laurence TA, Neuhauser D, et al. (2006) Shot-noise limited single-molecule FRET histograms: Comparison between theory and experiments. J Phys Chem B 110: 22103-22124.

9. Kapanidis A, Laurence T, Lee N, Margeat E, Kong X, et al. (2005) Alternating-laser excitation of single molecules. Acc Chem Res 38: 532–533.

10. Muller BK, Zaychikov E, Brauchle C, Lamb DC (2005) Pulsed interleaved excitation. Biophys J 89: 3508-3522.

11. Doose S, Heilemann M, Michalet X, Weiss S, Kapanidis AN (2006) Periodic acceptor excitation spectroscopy of single molecules. Eur Biophys J 36: 669–674.

12. Kudryavtsev V, Sikor M, Kalinin S, Mokranjac D, Seidel CAM, et al. (2012) Combining mfd and pie for accurate single-pair frster resonance energy transfer measurements. ChemPhysChem 13: 1060–1078.

13. Eggeling C, Berger S, Brand L, Fries J, Schaffer J, et al. (2001) Data registration and selective single-molecule analysis using multi-parameter fluorescence detection. J Biotechnol 86: 163–180.

14. Wilson G (2006) Where's the real bottleneck in scientific computing? American Scientist 94: 5–6.

15. Merali Z (2010) Computational science: Error, why scientific programming does not compute. Nature 467: 775–777.

16. Lee NK, Kapanidis AN, Wang Y, Michalet X, Mukhopadhyay J, et al. (2006) Accurate fret measurements within single diffusing biomolecules using alternating-laser excitation. Biophys J 88: 2939–2953.

17. van der Walt S, Colbert SC, G V (2011) The numpy array: A structure for efficient numerical computation. Comput Sci Eng 13: 9–12.

18. Hoffmann A, Nettels D, Clark J, Borgia A, Radford SE, et al. (2011) Quantifying heterogeneity and conformational dynamics from single molecule fret of diffusing molecules: recurrence analysis of single particles (rasp). Phys Chem Chem Phys 13: 1857–1871.

19. Murphy RR, Danezis G, Horrocks MH, Jackson SE, Klenerman D (2014) Bayesian inference of accurate population sizes and fret efficiencies from single diffusing biomolecules. Anal Chem (submitted April 2014) : ?–?

Figure Legends

Tables

Table 1. Donor DNA Sequence

Construct	Sequence
Donor	TACTGCCTTTCTGTATCGC**5**TATCGCGTAGTTACCTGCCTTGCATAGCCACTCATAGCCT

DNA sequence of the donor-labelled strand, where **5** is a deoxy-T nucleotide, labelled with Alexa Fluor 488 at the C6 amino position.

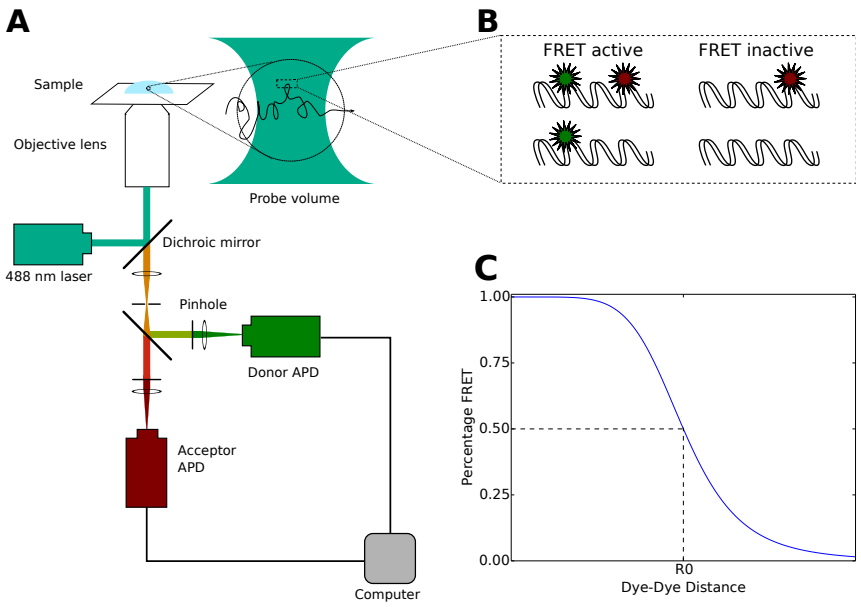

Figure 1. Instrumentation for a smFRET experiment. A) The confocal microscope, excitation and detection apparatus. B) Labelled molecules diffuse through the excitation volume. C) The characteristic sigmoidal dependence of FRET efficiency on dye-dye distance.

Table 2. Acceptor DNA Sequences

Separation / bp	Acceptor Sequence
4	AGGCTATGAGTGGCTATGCAAGGCAGGTAACTACGCGATAAGCGA**6**
6	AGGCTATGAGTGGCTATGCAAGGCAGGTAACTACGCGATAAGCGATA**6**
8	AGGCTATGAGTGGCTATGCAAGGCAGGTAACTACGCGATAAGCGATACA**6**
10	AGGCTATGAGTGGCTATGCAAGGCAGGTAACTACGCGATAAGCGATACAGA**6**
12	AGGCTATGAGTGGCTATGCAAGGCAGGTAACTACGCGATAAGCGATACAGAAA**6**

Preparing the dual-labelled dsDNA. An acceptor-labelled ssDNA, with the sequence shown was annealed to the indicated donor construct, to yield a dual-labelled construct with the labels separated by the given number of base pairs. In the displayed acceptor-strand sequences, **6** is a deoxy-T nucleotide, labelled with Alexa Fluor 647 at the C6 amino position.

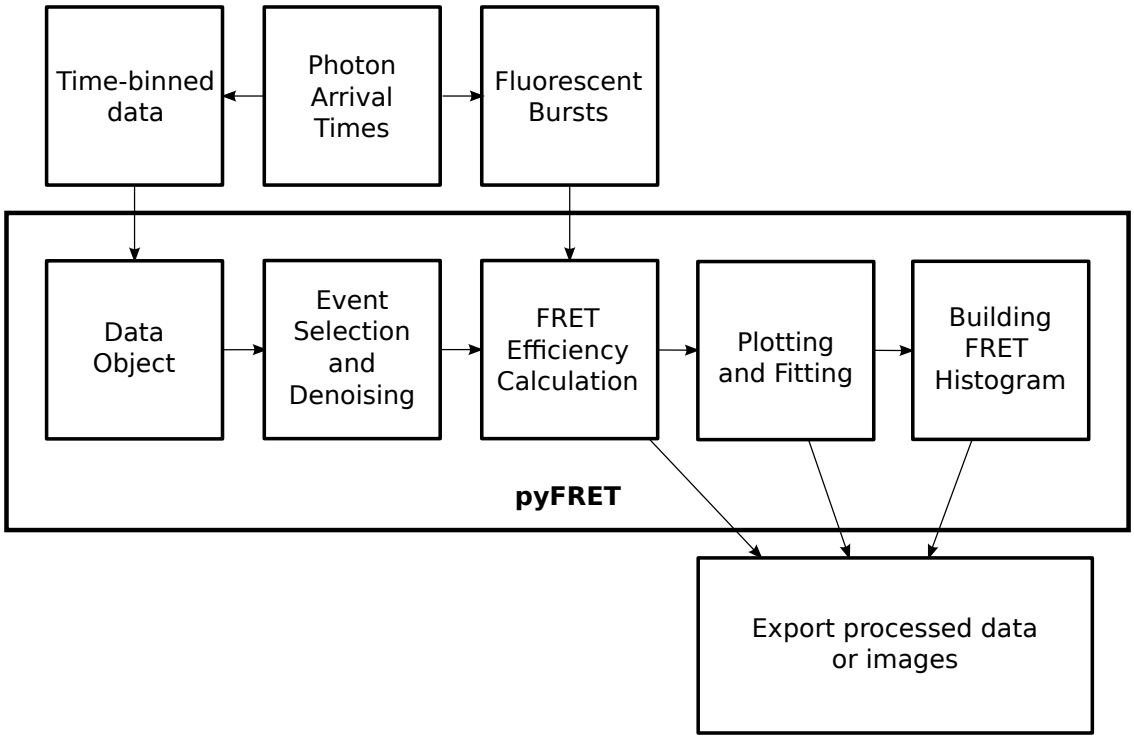

Figure 2. Typical workflow for data analysis using pyFRET.

Table 3. Bin-times

Bin-time / μs	T / bins
10	100
20	50
40	25
50	20
100	10
500	2
1000	1

Bin lengths and the corresponding minimum burst duration used to evaluate the effect of bin-time on burst search performance.

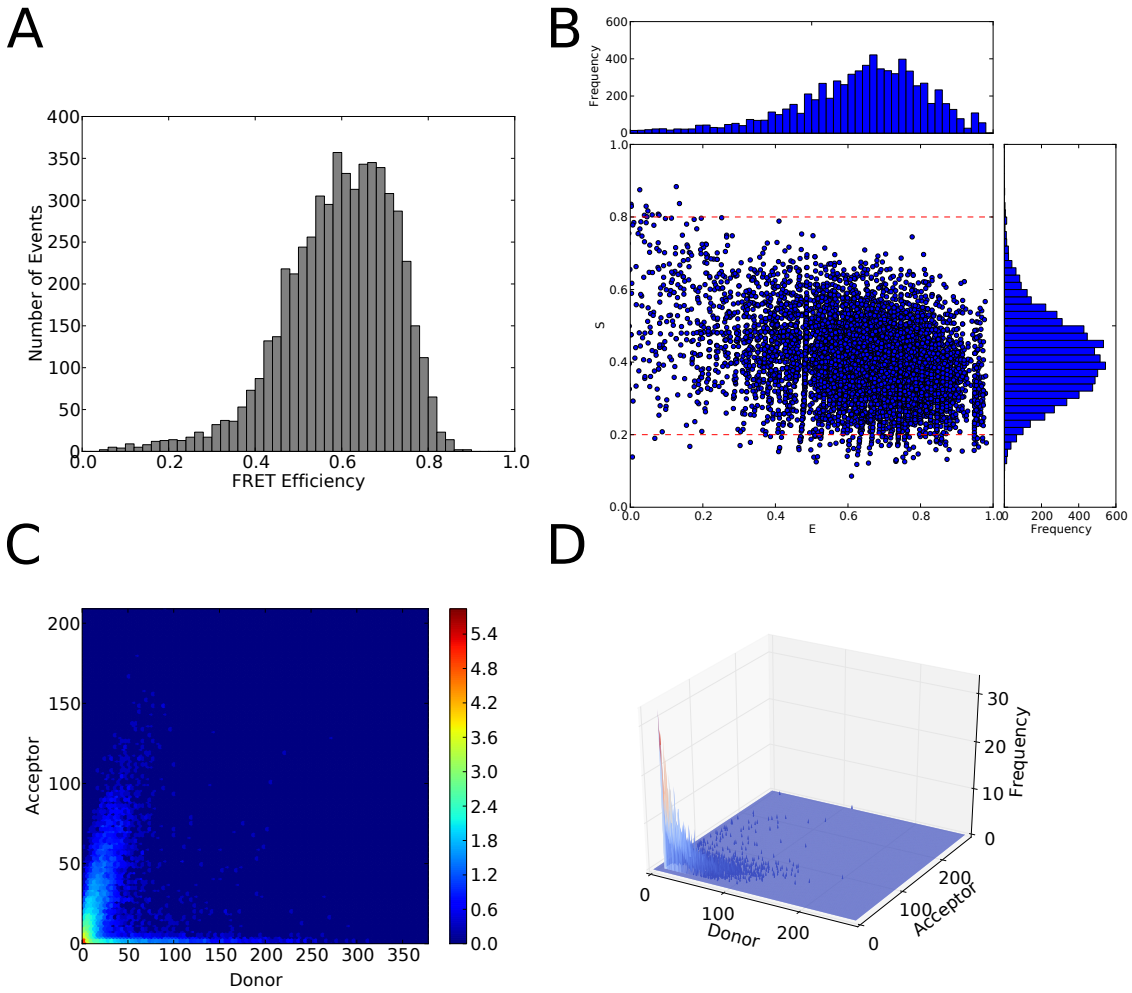

Figure 3. Figures made using pyFRET. A) A Proximity Ratio histogram. B) A scatter-plot of FRET efficiency and fluorophore stoichiometry from ALEX data. C) A heatmap of event frequencies. D) A 3D plot of event frequencies.

Table 4. RASP Parameters

Parameter	Value
T	20
M	10
L	10
E_{min}	0.65
E_{max}	0.85
ΔT	1 ms

Paramters used in RASP analysis of the 6 bp duplex - 12 bp duplex mixture.

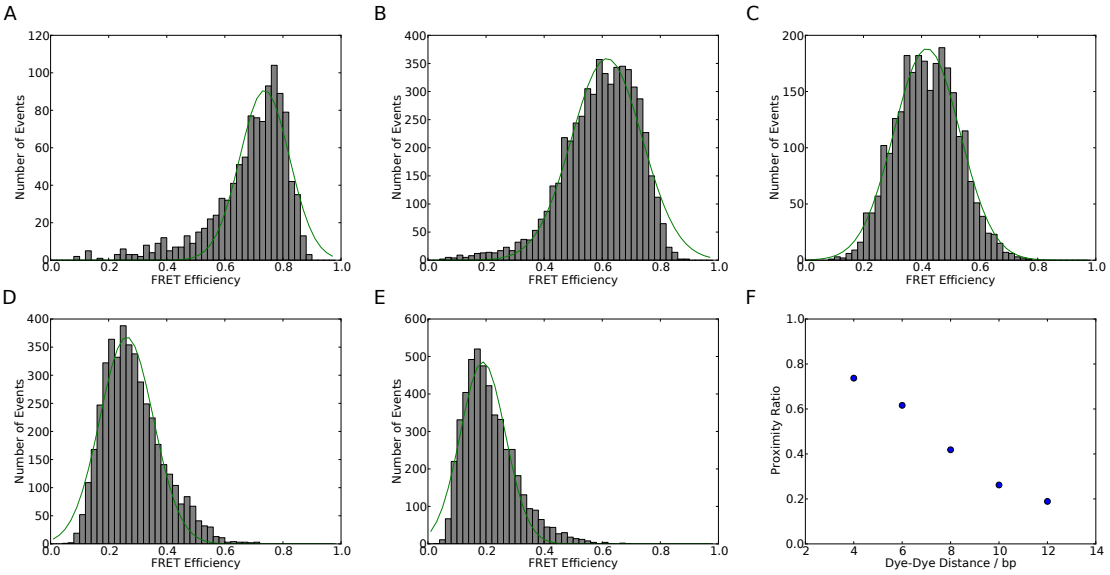

Figure 4. Analysis of FRET data from DNA duplexes using pyFRET. A - E: Fitted FRET histograms from DNA duplexes labelled with a dye-dye separation of 4, 6, 8, 10 and 12 base pairs respectively. F) Characteristic sigmoidal curve of FRET efficiency against dye-dye distance.

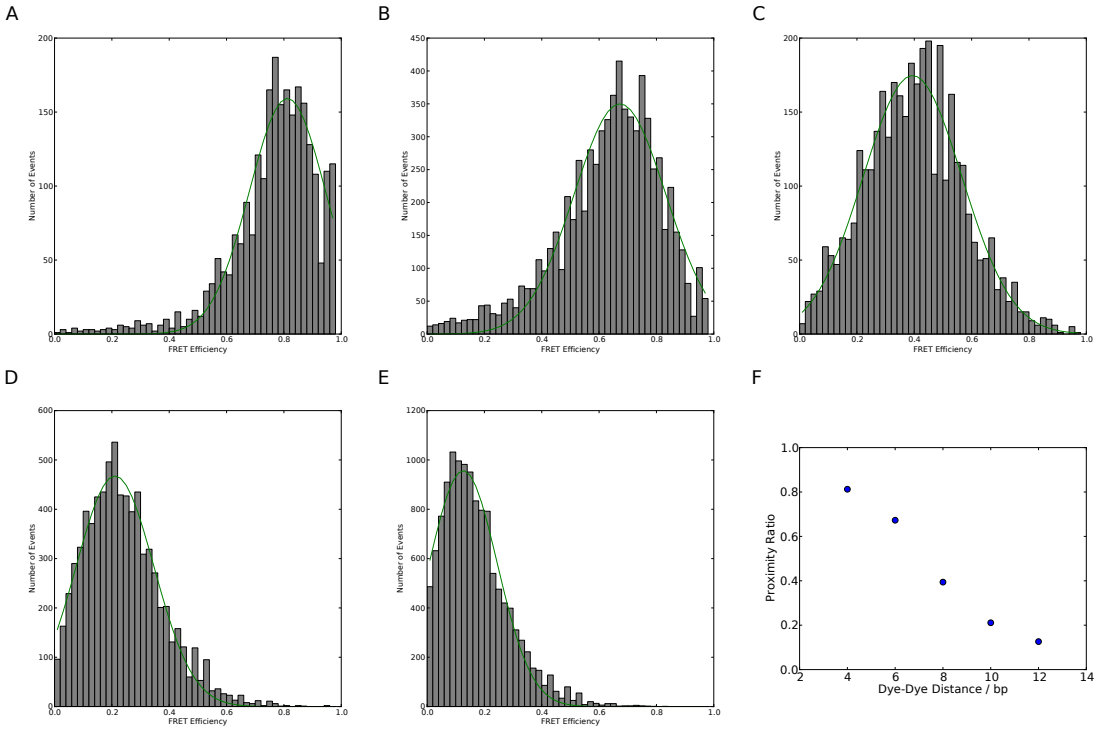

Figure 5. Analysis of ALEX data from DNA duplexes using pyALEX. A - E: Fitted FRET histograms from DNA duplexes labelled with a dye-dye separation of 4, 6, 8, 10 and 12 base pairs respectively. F) Characteristic sigmoidal curve of FRET efficiency against dye-dye distance.

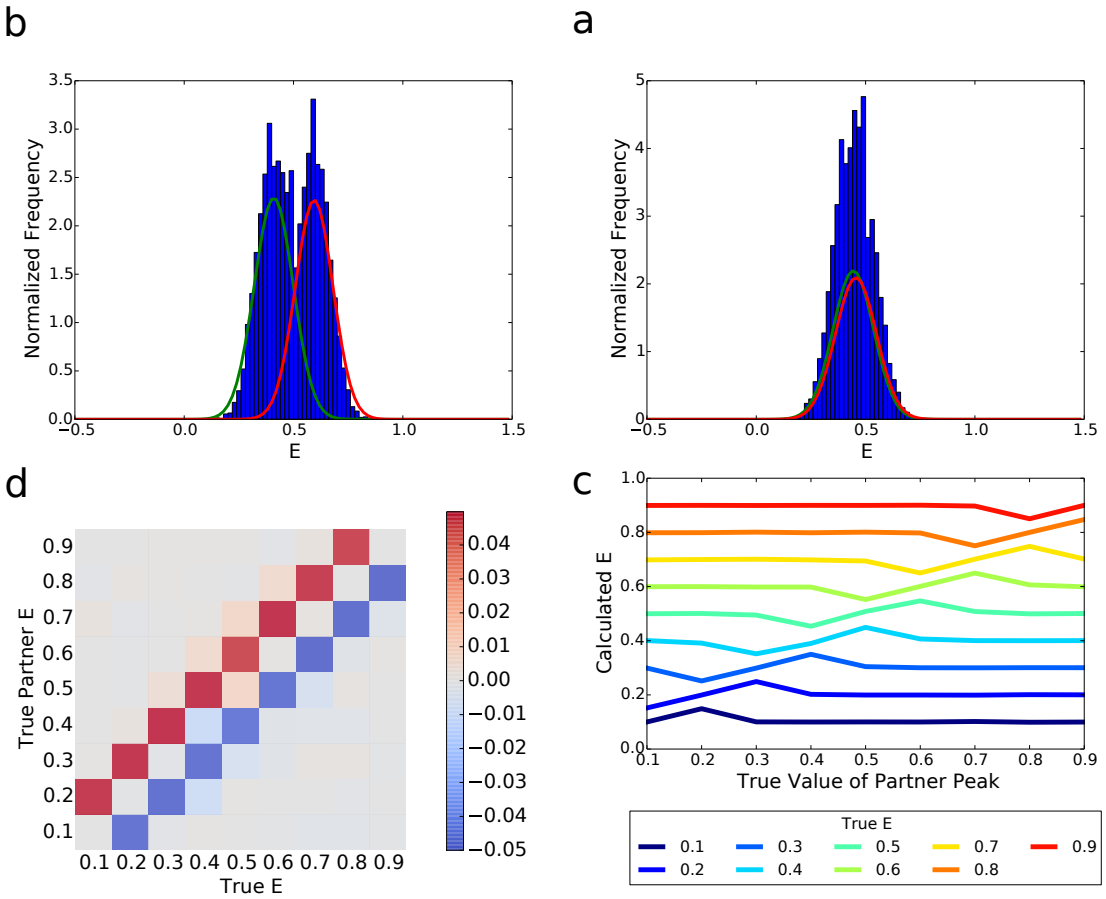

Figure 6. Benchmarking the gaussian fitting process using simulated data. A) Fitting a two-component Gaussian mixture to a simulated dataset simulating a 1:1 ratio of FRET populations, with FRET efficiencies of 0.4 and 0.6. The two peaks are easily distinguished and fitted correctly. B) Fitting a two-component Gaussian mixture to a simulated dataset simulating a 1:1 ratio of FRET populations, with FRET efficiencies of 0.4 and 0.5. There is significant overlap between the two peaks, so they cannot be distinguished by the fitting algorithm. C) Heatmap showing the relationship between partner FRET efficiency and the error in calculated FRET efficiency. Gaussian fits of datasets where the FRET efficiencies of the two peaks are extremely close to each other are distorted towards an intermediate value. D) One-dimensional representation of the heatmap shown in C), showing the distortion in calculated FRET efficiency when the two components of the mixture have very similar FRET efficiencies.

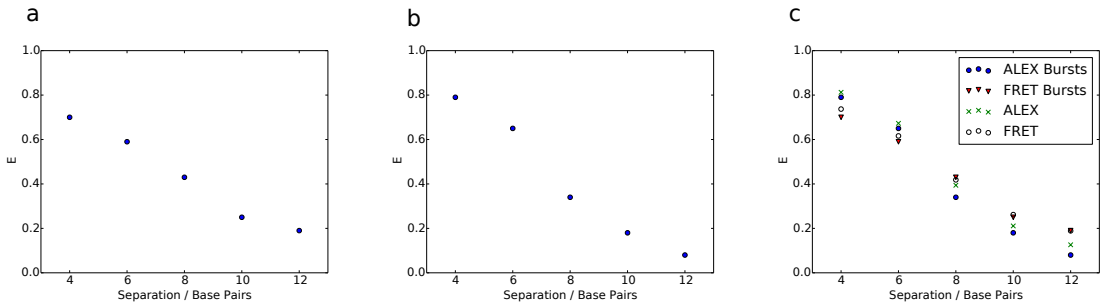

Figure 7. Plot of FRET Efficiency vs dye-dye separation. A) FRET data, analysed using the DCBS burst search algorithm. B) ALEX data, analysed using the DCBS burst search algorithm. C) Comparison of different methods. Blue circles and red triangles show ALEX and FRET burst data respectively; green crosses and open circles show simple thresholded data from ALEX and FRET experiments respectively.

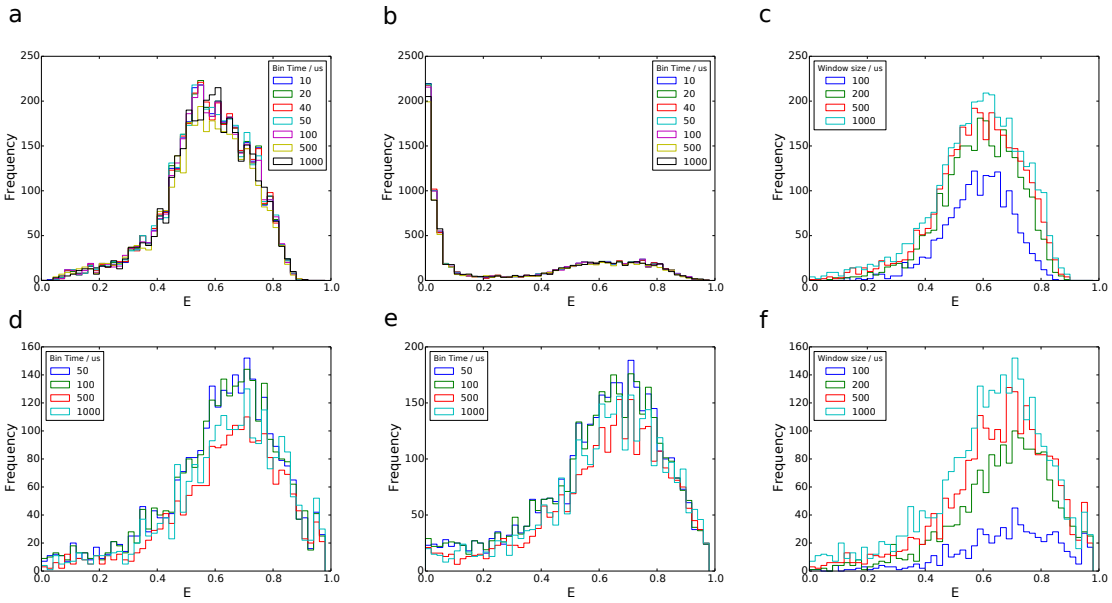

Figure 8. Evaluating the effect of detection window and bin time on the DCBS burst search algorithm for FRET data. A) Varying the length of the time-bin has little effect on the performance of the DCBS algorithm. B) Varying the length of the time-bin has little effect on the performance of the APBS algorithm. C) Reducing the length of the minimum detection window used in DCBS selects for very bright bursts. For very short windows, this reduces the number of detected bursts but does not affect the calculated FRET efficiency.

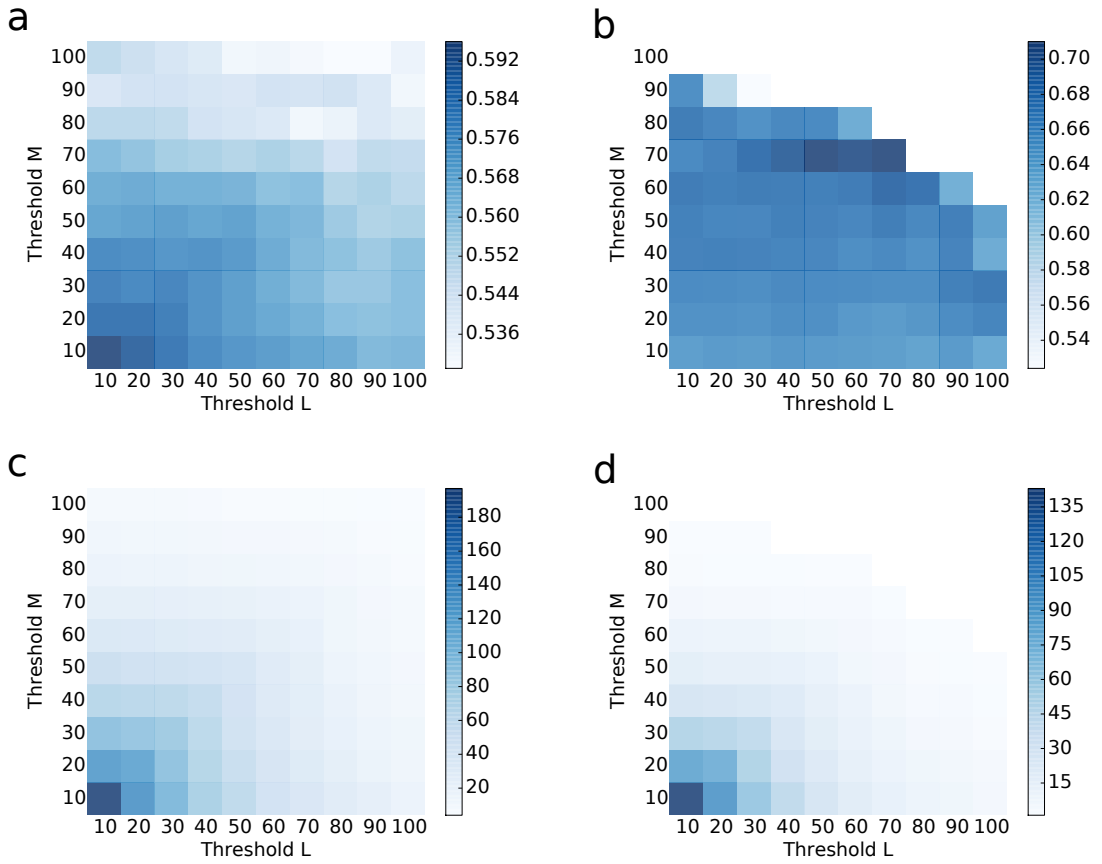

Figure 9. Heatmaps showing the effect on calculated FRET efficiency and peak area of varying the burst search thresholds L and M. A) Calculated FRET efficiency from DCBS analysis of FRET data. B) Calculated FRET efficiency from DCBS analysis of ALEX data. c) Calculated peak area from DCBS analysis of FRET data. D) Calculated FRET efficiency from DCBS analysis of ALEX data.

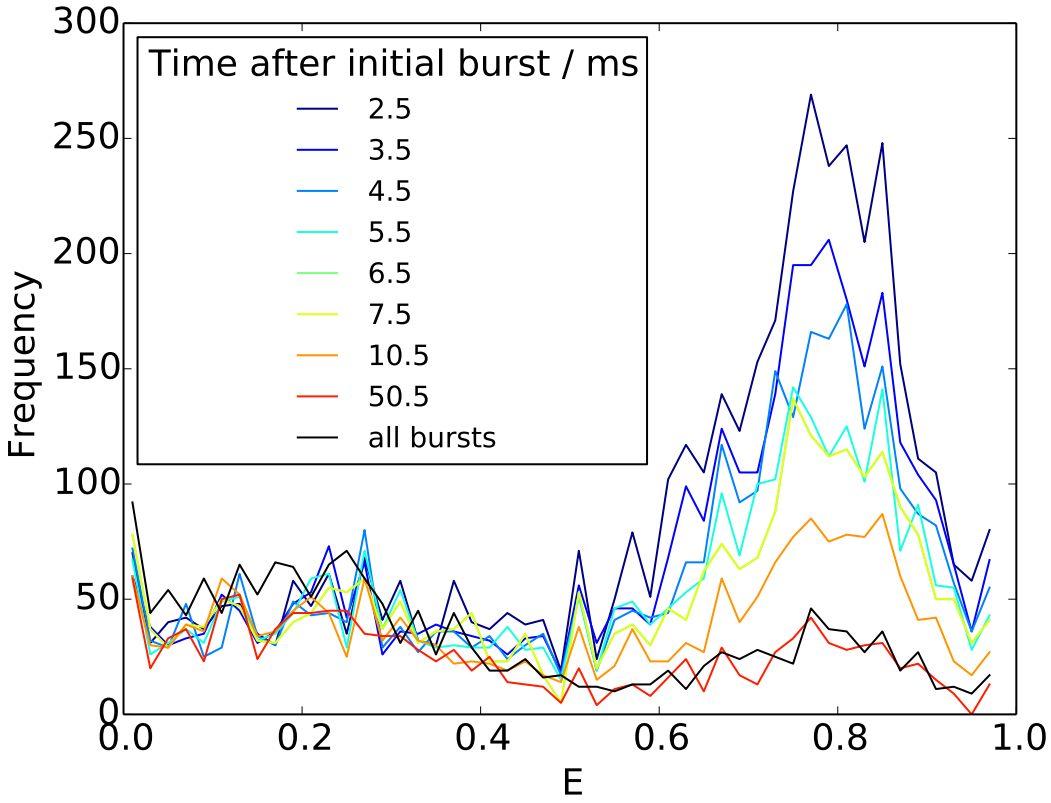

Figure 10. RASP analysis of FRET data from a mixture of a high-FRET and low-FRET duplex. The black line shows the baseline peak areas for the two duplexes. Other lines show the FRET histograms generated from bursts that occurred at the indicated time following a high-FRET burst. The Recurrence Interval ΔT was 1 ms, centred at the times shown in the legend. Note the greatly increased peak area for high-FRET bursts, demonstrating the recurrence of high-FRET events as molecules diffuse back into the confocal volume.