TwoTone 3.1 user's manual

Seamus J. Holden[1]

[1]Clarendon Laboratory, University of Oxford, Parks Road, Oxford, OX1 3PU, UK. Email:
s.holden1@physics.ox.ac.uk

Contents

1 Introduction

TwoTone is a program for automatic extraction and analysis of surface based single molecule FRET data. The principles and methods behind the software are described in detail in Holden et al. (1), however we provide a very brief description here

Our source data consists of synchronized movies of fluorescence emission in the donor and acceptor emission channels from multiple randomly distributed surface-immobilized single molecules. Our goal is to obtain a set of fluorescence and FRET trajectories for the immobilized molecules in the image. Image analysis for tFRET consists of two steps: *image registration* and *data extraction*. Image registration consists of the generation of a mapping (transform matrix) between the donor and acceptor emission channels using a calibration image of fluorescent beads. Data extraction, that is the extraction of fluorescence information from the movies consists of three sub-steps: *detection & localization*, *association*, *photometry*. Molecules in the image are identified and localized to high precision (detection & localization), and then linked between each detection channel (association), using the mapping generated in the image registration step. Photon counts and FRET information for each frame of the movie then extracted from each molecule.

After data has been extracted from the movies, we generally want to analyze it further, by visualization and timetrace analysis. Firstly, we filter out molecules and data points which do not meet criteria for further analysis, for example because they likely arise from multiple closely space molecules, or molecules labeled with only a donor rather than both a donor and acceptor.

TwoTone can analyze smFRET data generated using continuous wave excitation (CW) or the alternating laser excitation scheme (ALEX). For the purposes of this manual I will focus on analysis of ALEX data, mentioning any crucial differences for CW at each stage.

If you use this software in work leading to a scientific publication, please cite Holden et al. (1). Please check our website, `http://www.physics.ox.ac.uk/users/kapanidis/Group/Main.Software.html`, for updates to the software.

The copyright of the TwoTone software is held by the Isis Innovation Ltd. TwoTone is released under an "academic use only" license; for details please see the accompanying "TWOTONE_LICENSE.doc". Usage of the software requires acceptance of this license. Note that the files 'bpass.m' and 'HIST2.m', distributed with TwoTone, remain the copyright of their respective owners (see those m-files for details).

2 Installation

TwoTone consists of the main software package, written in MATLAB, together with the MFITSIO2 library for analysis of FITS images. Additionally, the use of the GaussFitTools library, for fast least squares PSF fitting, is strongly recommend to maximize analysis speed. However, due to licensing requirements, GaussFitTools is distributed separately, and should be downloaded and installed separately if required (see Section 2.1). TwoTone has been compiled and tested using MATLAB version 7.10, under 32-bit Windows environments, and 32 or 64-bit Linux (Ubuntu 10.04). Apple systems are not supported at this time.

TwoTone requires the following MATLAB toolboxes: Optimization toolbox, Image Processing toolbox, Statistics Toolbox. Users wishing to reduce toolbox usage should consider compiling TwoTone as a standalone executable using `mcc`. For Ubuntu 32 bit install, the following packages are required from the repositories: liblapack-dev libblas-dev libcfitsio3 libcfitsio3-dev.For Ubuntu 64 bit install, the following are required: libcfitsio3 libcfitsio3-dev. Users of other distributions will need to find equivalent packages.

To install TwoTone and MFITSIO2, first ensure that any previous version of the software are uninstalled and removed from the MATLAB search path (use 'File' >> 'Set Path' in MATLAB to check and modify the MATLAB search path). Then, unzip 'twotone3.1ReleaseBundle.zip', load MATLAB, and navigate to the 'twotone3.1ReleaseBundle' directory within the MATLAB environment. At the MATLAB command prompt, type

```
>> installtwotone
```

Note: For installs under Linux, if MATLAB is a system-wide install (ie. installed to a location outside of the user's home directory using root privileges), the above instructions should be carried out running MATLAB with root privileges.

2.1 GaussFitTools installation

To install GaussFitTools, first ensure that any previous version of the software are uninstalled and removed from the MATLAB search path (use 'File' >> 'Set Path' in MATLAB to check and modify the MATLAB search path). Then, unzip 'GaussFitTools.zip', load MATLAB, and navigate to the 'GaussFitTools' directory within the MATLAB environment. At the MATLAB command prompt, type

```
>> installGaussFitTools
```

As before, Linux users may need to install GaussFitTools with root privileges.

2.2 Compiling TwoTone as a standalone executable

3 Basic usage instructions

TwoTone can analyze files in either the FITS (REF) or TIF format. The portions of the image corresponding to the donor and acceptor emission areas on the camera are manually specified at the image registration and data extraction stages.

3.1 Image registration

The first step is to generate a mapping between the donor and acceptor emission channels, using the function `transformMovies`. After being asked to select a calibration image, you will be presented with the following (text) menu:

```
----TirfImage calibration and transformation----
Press return to load calibration image file
```

```
Current image limits are:
Donor:          1 256 1 256
Acceptor:       257 512 1 256
Would you like to change the image limits (y/n) :
```

The limits are listed in order "MinX, MaxX, MinY, MaxY". Ideally we would use a pretty GUI interface to select these limits, but for now, identify them using software such as *ImageJ*, and then select "y" to the above question, and input the data as follows:

```
Please input pixel limits for each emmision channel
Donor X start: 1
Donor X end: 256
Donor Y start: 1
Donor Y end: 256
Acceptor X start: 257
Acceptor X end: 512
Acceptor Y start: 1
Acceptor Y end: 256
Pixel limits inputs were:
Donor:          1 256 1 256
Acceptor:       257 512 1 256
Is this correct (y/n) :
```

TwoTone will save these coordinates for future usage, so you should only need to enter this once if you usually use the same microscope.

MATLAB's `cpselect` GUI (Figure 1) is then launched, to allow selection of matching beads in each image channel. After you have selected a reasonable number of points ($>$20), close the `cpselect` tool by selecting 'File'→'Close Control Point Selection Tool', or using the shortcut 'Ctrl-W'. You will be presented with an overlay image of the donor (coloured green) and acceptor (coloured red) channels (Figure 2). Well overlaid yellow molecules (yellow is just the color sum of green and red) indicate a "good" transform. If your transform is acceptable, save it as prompted.

3.2 Data extraction

The next step is to extract your data. All of the settings for data extraction are controlled using a GUI called `twotoneALEX` (Figure 3, `twotoneCW` for CW analysis). After setting your parameters, data extraction may be run for multiple files. Most of your parameters may be estimated automatically based on the size of molecular PSF (see below).

On loading your data ('File'→'Load Movie'), you will see an averaged image of the data for each detection channel (Figure 3.3-5). Below each image is a 'Threshold' box (Figure 3.6), together with a button to run auto-detection (Figure 3.7). Auto-detection identifies molecules (shown as red circles) as local maxima in a filtered image (viewable by clicking the 'Show filtered images' checkbox, Figure 3.16), and localizes them to high precision using a least squares fit to a Gaussian

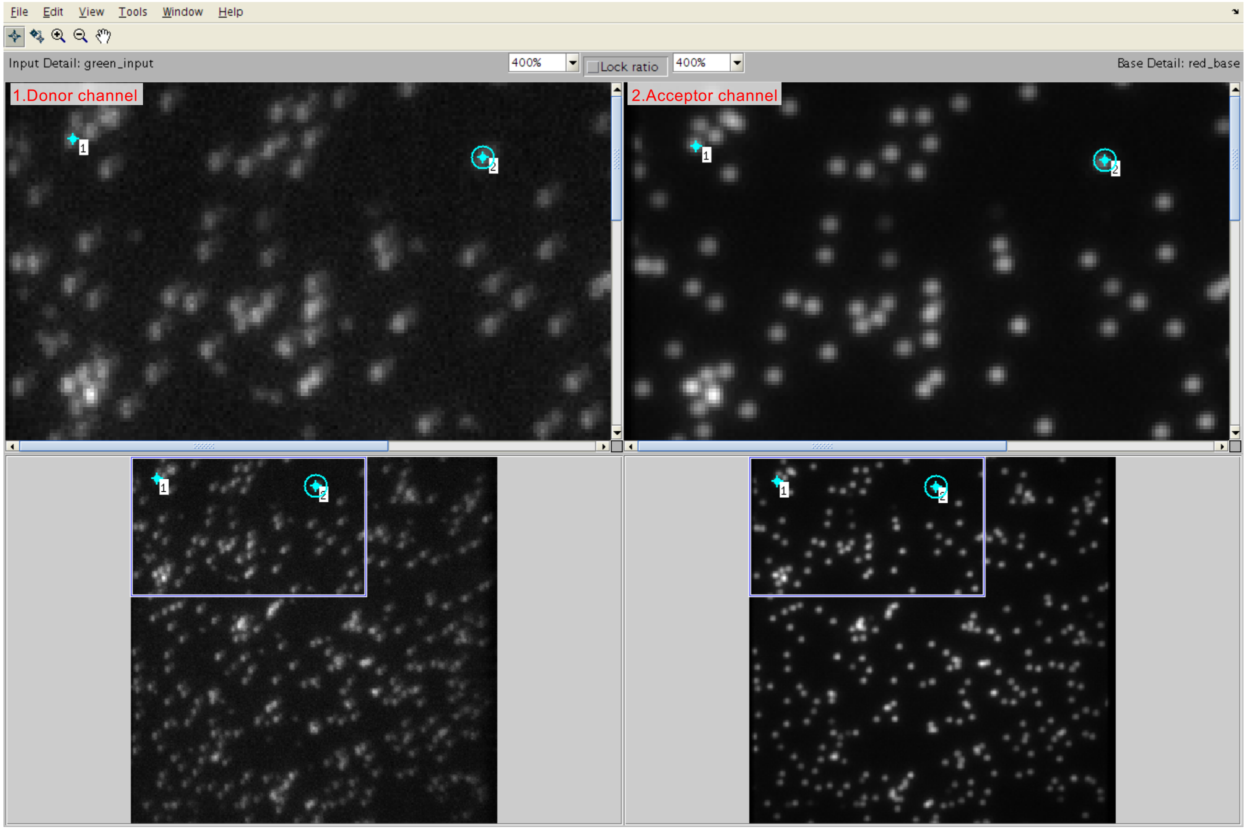

Figure 1:

PSF. You should carry out auto-detection for each channel, ensuring that the threshold is sufficient to pick up your molecules without including significant numbers of spurious, 'false', localizations.

Load the mapping for your data ('File'→'Load TFORM'), choose the filtering algorithm settings (Figure 3.8, see Section 3.2.3)and then click 'Link channels'.

Linked molecules are plotted as red crosses in each image, and the 'Linking algorithm results' window will appear (Figure 4), which shows various pieces of useful information about your data. The 'Cluster Distance Distribution' (Figure 4.A.1) is particularly useful since it a) shows you the quality of your mapping and b) allows you to set the 'Cluster distance threshold'. The linking algorithm maps all of the identified molecule positions into the donor emission channel coordinate system, and carries out distance-based hierarchical clustering using the cluster distance threshold. Molecules with a separation closer than the threshold are grouped together, and groups containing more than one molecule in a single channel are exclude from fitting. The distance distribution between all the molecules shows a zero-peak for the random distribution of molecules within the same channel, together with a secondary peak displaced from zero which indicates the mean distance shift between channels after mapping. With a poor mapping, the secondary peak will be significant (Figure 4.B); for an accurate mapping, the secondary peak should be nearly in-

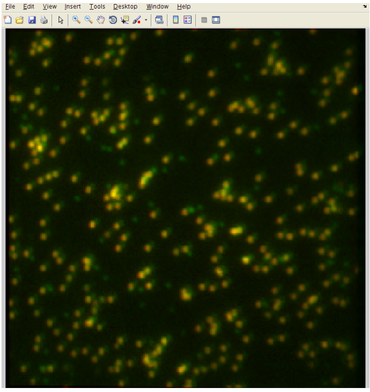

Figure 2:

distinguishable from the zero-peak (Figure 4.C). *The cluster distance threshold should be set just large enough to contain both the zero-peak and the secondary peak; for an accurate mapping, set the distance threshold just larger than the PSF width (see below), to allow for some uncertainty in the localization, eg due to overlapping molecules.*.

The eccentricity distribution (Figure 4.A.2) is a measure of the asymmetry of your molecules. An in-focus, isolated molecule will be reasonably symmetric, therefore very large asymmetry is indicative of overlapping unresolved molecules, which may be excluded by eccentricity-based filtering.

The width distribution (Figure 4.A.3) shows a plot of the 1-sigma PSF widths for the auto-detected molecules along the major and minor axes of the elliptical Gaussian model. Again, molecules with very large width likely arise from multiple overlapping molecules. Additionally, this plot gives us a good estimate of the average PSF width, which may be used to automatically set most of the fitting parameters (see below).

3.2.1 Automatic parameter estimation

You may use the information from the 'Linking algorithm results' window to adjust TwoTone default settings, located in 'Edit'→'Settings' (Figure 5). The best way to do this is to input the estimate of the mean PSF width (1.5 pixel for the data presented in Figure 4.A) into the 'Auto-calculate all parameters' box (Figure 5.8) and click 'Update Settings'. This updates all parameters except for the image limits of the donor and acceptor channels (Figure 5.1), and the frame averaging range (Figure 5.2). Hereafter I refer to the 1-sigma PSF width as '*width*'.

3.2.2 TwoTone parameters

The TwoTone parameters are:

1. *Image properties: Image limits.* This sets the image limits for each emission channel. Note that these must be the same as those used for TFORM generation, otherwise the mapping

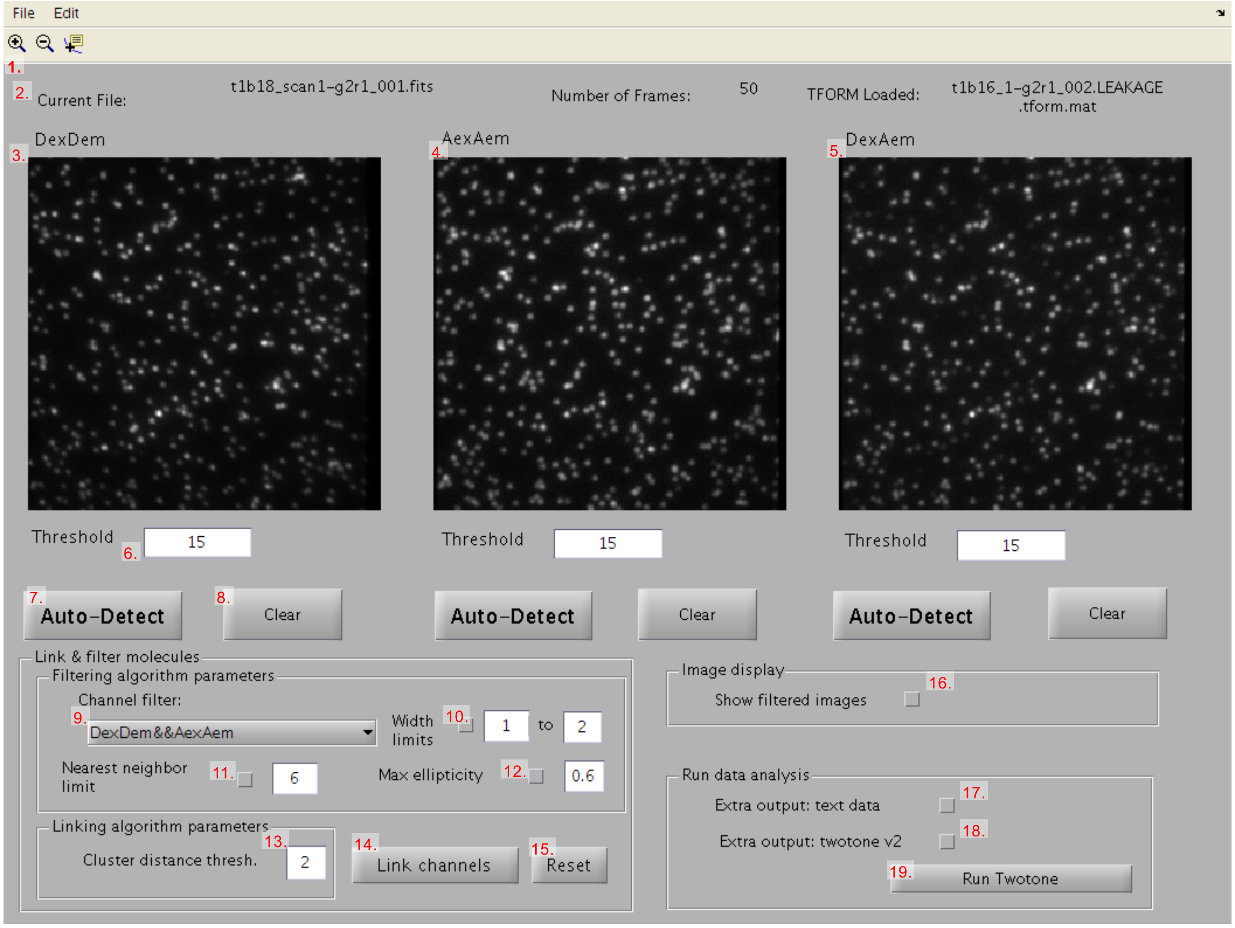
Current File: t1b18_scan1-g2r1_001.fits
Number of Frames: 50
TFORM Loaded: t1b16_1-g2r1_002.LEAKAGE .tform.mat
DexDem Threshold 15
AexAem Threshold 15
DexAem Threshold 15
Channel filter: DexDem&&AexAem
Width limits ☐ 1 to 2
Nearest neighbor limit ☐ 6
Max ellipticity ☐ 0.6
Cluster distance thresh. 2
Show filtered images ☐
Extra output: text data ☐
Extra output: twotone v2 ☐

Figure 3:

will not work properly.

2. *Image properties: Average frames.* This sets the frame range over which to calculate the averaged image for autodetection. I usually choose to skip the first frame since a quirk of our hardware means this is often recorded as a blank image.

3. *Autodetection parameters: Band pass kernel diameter.* Before autodetection, the image is convolved with a Gaussian kernel of this diameter. This increases the relative amplitude of Gaussian PSFs of this size, and reduce background and large scale features in the image. *Set this to* `max(round(2*width),1)`.

4. *Autodetection parameters: Fitting subimage radius.* Molecules are first localized to pixel-level accuracy by identifying local maxima in the image. Molecules are then fitted with an elliptical Gaussian PSF by selecting a small square subimage around the identified molecule to localize the PSF with high precision. The most important information for localization of

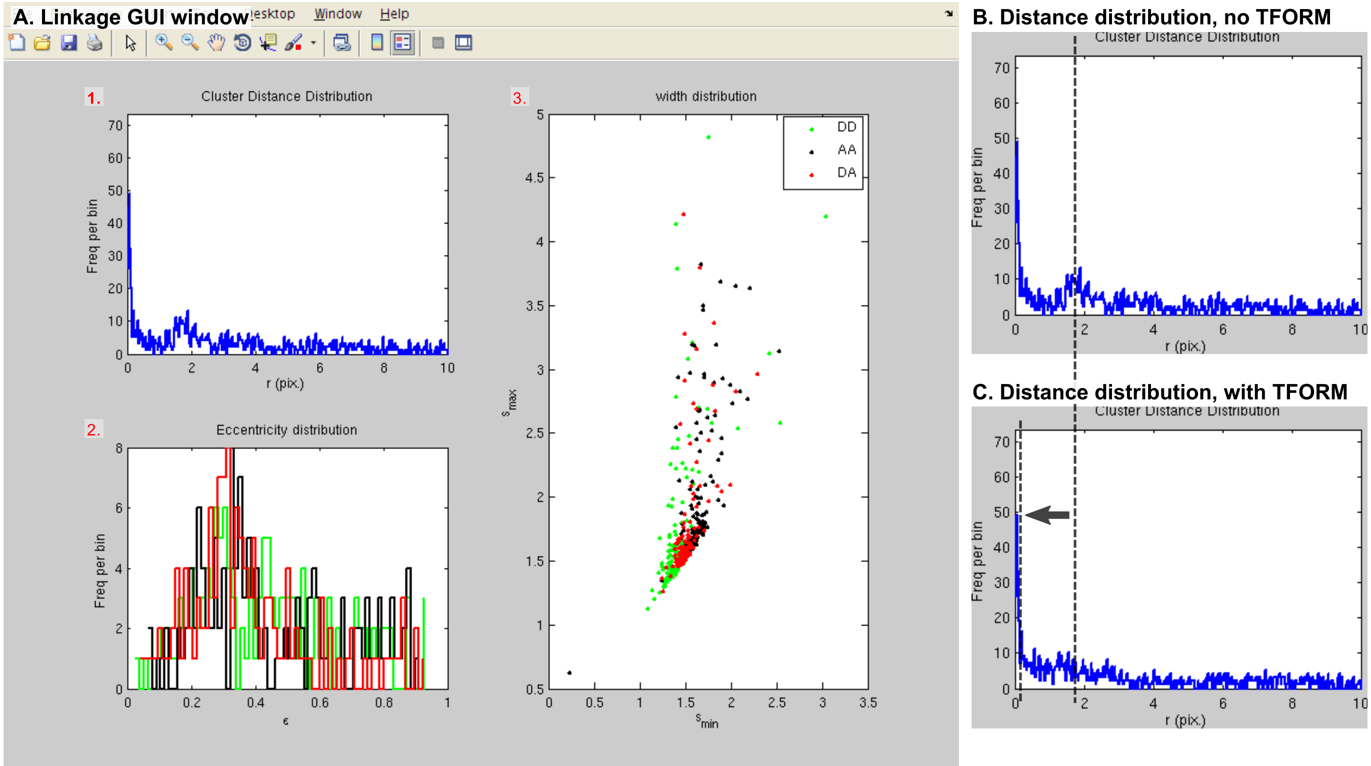

Figure 4:

PSF position is contained within the slope surrounding the maxima of the PSF (2), in contrast to the case of photon counting, where it is the levels of maxima and minima of the PSF which are important. This means that a quite small fitting subimage radius may be used for position localization, which has the advantage of improved performance in crowded images, and reduced fitting time. *Set this to* `max(2, round(2*width))`.

5. *Data analysis parameters: Analysis algorithm.* Currently, there are four algorithms available for photon counting in TwoTone. These are *aperture photometry* ('ring'), LSQ fitting of a fixed position circular Gaussian PSF ('fixedGauss'), LSQ fitting of a fixed position elliptical Gaussian PSF ('fixedGaussEllipse'), and LSQ fitting of a free position elliptical Gaussian PSF ('freeGaussEllipse'). The default setting is fixedGauss, which provides a good balance between final signal to noise, performance in crowded fields and fitting speed. If there is significant focal drift within a movie, PSFs are otherwise very elliptical, or simply if maximum signal to noise is required, fixedGaussEllipse should be used. If there is significant (x, y) drift in the movie, ('freeGaussEllipse') should be used. For applications at high signal to noise and low image density, fitting speed may be maximized by using the ring fit. Algorithm parameters and settings are listed below

   - Fit algorithm: *fixedGauss (default)*, fixedGaussEllipse, freeGaussEllipse.

  - *windowRadius*: Size of (square) subimage. The subimage must contain both the local image background and the maximum intensity, therefore a relatively large radius is required. *Set this to* `max(2, round(4*width))`.
  - *minWidth*: Minimum fitted PSF width. This is to avoid fitting noisy background with a 'spike' PSF. *Set this to* `max(0,width - 0.5)`.
  - *maxWidth*: Maximum fitted PSF width. This is to avoid fitting noisy background with a low amplitude, very large width PSF, which causes transient large photon count spikes in the absence of signal. *Set this to* `width + 0.5`.
- Fit algorithm: freeGaussEllipse. In addition to the above parameters, freeGaussEllipse has one additional parameter
  - *maxPositionChange*: Maximum x or y position change from initial localization. This should be half of the subimage radius. *Set this to* `windowRadius/2`.
- Fit algorithm: ring.
  - *innerCircleRadius*: Radius of circle centered on molecule position used to measure photon count.
  - *outerCircleRadius*: Outer radius of aperture centered on molecule position, used to estimate local background for molecule.
  - *ringTime*: In order to increase signal to noise, the background measurement (assumed to vary only slowly with time) is averaged over *ringTime* frames before and after the current frame (ie total $2ringTime + 1$, alternation is accounted for by skipping frames). *By default, this is set to 3.*

6. *Data analysis parameters: Fit parameters.* Each fit parameter mentioned above may be changed using this drop down menu box.

7. *Data analysis parameters: File append name.* The TwoTone output file is saved with this filename. Eg., for input file 'foo.fits', and extension, '.dataout.mat', the output data is saved as 'foo.fits.dataout.mat'.

8. *Auto-calculate all parameters.* Auto-set parameters based on PSF width. Use the PSF width as estimated from the linking window.

3.2.3 TwoTone filtering algorithm

The filtering algorithm (Figure 3.9-12) allows a subset of molecules to be selected for analysis. The most important option here is the 'Channel filter' (Figure 3.9), which filters molecules based on stoichiometry. Default is to select molecules emitting in DD and AA ('DD&&AA', note && is shorthand for logical AND, || is shorthand for logical OR), there are lots of other options. Next you can set 'Width limits' (Figure 3.10), to exclude molecules that are very broad or very narrow. You can also exclude molecules with very close nearest neighbors (Figure 3.11) or very eccentric molecules (Figure 3.12). Generally, at this stage I prefer to filter molecules based on stoichiometry only, and leave shape/ size/ nearest-neighbor filtering for the data analysis stage.

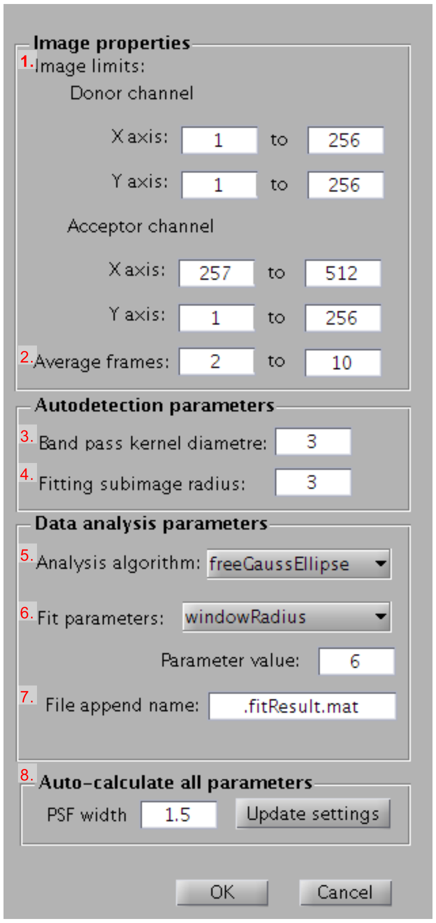

Image properties
Image limits:
Donor channel
X axis: 1 to 256
Y axis: 1 to 256
Acceptor channel
X axis: 257 to 512
Y axis: 1 to 256
Average frames: 2 to 10

Autodetection parameters
Band pass kernel diametre: 3
Fitting subimage radius: 3

Data analysis parameters
Analysis algorithm: freeGaussEllipse
Fit parameters: windowRadius
Parameter value: 6
File append name: .fitResult.mat

Auto-calculate all parameters
PSF width 1.5

Figure 5:

3.2.4 Running TwoTone

After you have set all your parameters, you are ready to run TwoTone. TwoTone can be run in batch mode, so for movies performed under similar conditions, you can use a single movie to set the correct parameters (perhaps checking the parameters for a second movie) and then analyze all the movies together. Run TwoTone by clicking the push-button (Figure 3.19). In addition to the standard TwoTone output format, specified in Section 5, you can output a simple text-file version of the results (Figure 3.17). For users of previous TwoTone versions (esp. Seneca users), the old TwoTone data format may also be output (Figure 3.18).

3.3 Data analysis

TwoTone provides some basic data analysis capabilities: data filtering, timetrace plotting, and histogramming of data. For ALEX data, the two relevant functions are `plotTimetraceALEX` and

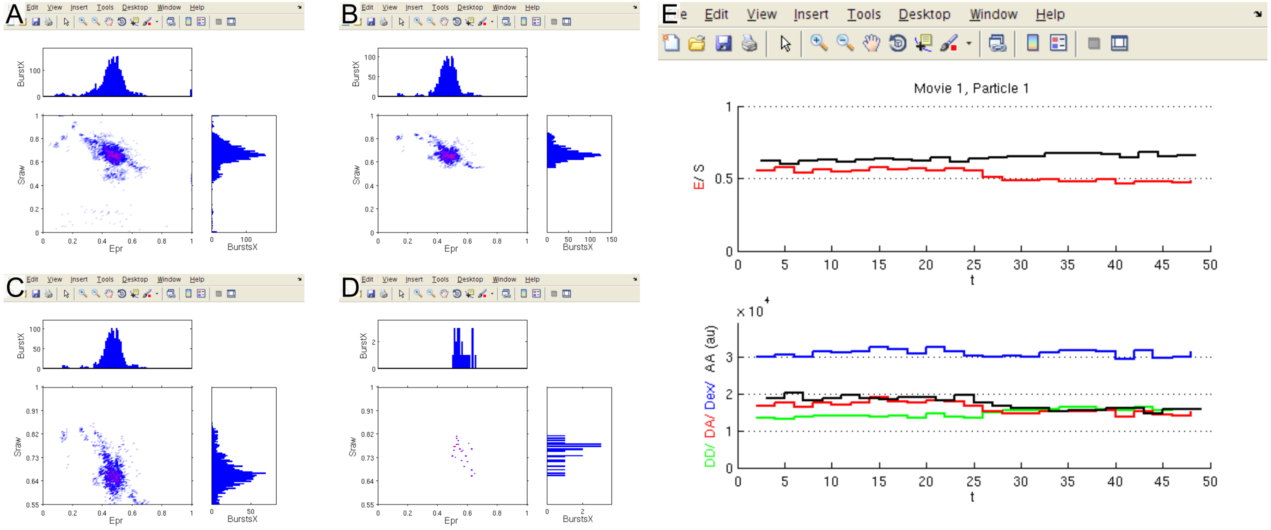

Figure 6:

`plotHistALEX` (for CW, the relevant functions are `plotTimetraceCW` and `plotHistCW`

First, lets consider histogramming some data for dsDNA at 50% FRET, analyzed using elliptical Gaussian fitting, in the file 'mydata.mat'. I enter the command:

```
>> plotHistALEX('myData.mat')
```

and obtain the output in Figure 6.A. However I would like to filter my data a little, so I input the following:

```
>> plotHistALEX('myData.mat',{'all_aDetParam','nearestNeighbor','>'
    ,6},{'fretData','AA','>',500},{'fretData','S','>',0.55})
```

which produces Figure 6.B. The filtering format consists of inputing a cell (the curly braces '{}') containing the following information: {*FilterType, FilterParam, Range, Value*}. The options for each of these parameters are as follows:

- FilterType: 'fretData'. All *frames* failing this threshold type are excluded.
    - FilterParam: 't_Dex', 't_Aex', 'DD', 'DA','AA', 'AA', 'E', 'S'
- FilterType: 'all_aDetParam'. All *molecules* failing this threshold type are excluded.
    - FilterParam: 'nearestNeighbor'
- FilterType: 'DD_aDetParam'. All *molecules* failing this threshold type are excluded.
- FilterType: 'DA_aDetParam'. All *molecules* failing this threshold type are excluded.
- FilterType: 'AA_aDetParam'. All *molecules* failing this threshold type are excluded.

  - FilterParam: ‘amplitude’, ‘sx’, ‘sy’, ‘eccentricity’,
- FilterType: ‘DD_aDetPos’. All *molecules* failing this threshold type are excluded.
- FilterType: ‘DA_aDetPos’. All *molecules* failing this threshold type are excluded.
- FilterType: ‘AA_aDetPos’. All *molecules* failing this threshold type are excluded.
  - FilterParam: ’X’ ’Y’
- Range: ‘$<$’ or ‘$>$’. Molecules/frames *outside* of this threshold are excluded.
- Value: Number

For the above example, I excluded out molecules with very close nearest-neighbors, *frames* with very dim acceptor emission, and *frames* with stoichiometry $S < 0.55$. As many filters as desired may be passed to the plot functions.

Since I have excluded frames with low stoichiometry, it makes sense to also adjust the plot range. I do this by:

```
>> plotHistALEX(’myData.mat’,’SLim’, [.55 1],{’all_aDetParam’,’
   nearestNeighbor’,’>’,6},{’fretData’,’AA’,’>’,500},{’fretData’,’S’
   ,’>’,0.55})
```

obtaining the output in Figure 6.C.

To plot only a selection of molecules, I can use either the ‘RelativeIndex’, referring to the n-th molecule after any filtering or on combination of multiple movies, or I can use the ‘AbsoluteIndex’, n, m, where n is the movie number, and m is the m-th molecule analyzed in that movie. I this case, I plot the 10th molecule in the movie:

```
>> plotHistALEX(’myData.mat’,’SLim’, [.55 1],{’all_aDetParam’,’
   nearestNeighbor’,’>’,6},{’fretData’,’AA’,’>’,500},{’fretData’,’S’
   ,’>’,0.55},’AbsoluteIndex’,[1,10])
```

which gives me:

```
??? Error using ==> selectParticle at 59
No data meets these criteria
```

This means that there molecule number 10 did not meet the filtering criteria. If I instead used relative indexing,

```
>> absoluteIndex = plotHistALEX(’myData.mat’,’SLim’, [.55 1],{’
   all_aDetParam’,’nearestNeighbor’,’>’,6},{’fretData’,’AA’,’>’
   ,500},{’fretData’,’S’,’>’,0.55},’RelativeIndex’,1)
```

I obtain Figure 6.D. The ‘absoluteIndex’ of the first molecule to meet the criteria is molecule [1 19], which is the 19th molecule in the 1st movie (in this case the only movie).

It is also possible to analyze multiple movies, by running `plotHistALEX({’myData1.mat’,’myData12.mat’},’`

Additionally, the filtered data can be output for further analysis by running:

```
>> [absoluteIndex plotData]= plotHistALEX('myData.mat',...)
```

'plotData' is a structure containing a matrix of data, 'plotData.data' (each row is one frame), and a cell 'plotData.dataName', listing the data contained in each column.

The interface for plotting timetraces,using `plotTimetraceALEX` is identical, except that if you do not specify a particle, the first (relative indexed) particle is plotted. Ie,

```
plotTimetraceALEX('myData.mat')
```

is identical to

```
plotTimetraceALEX('myData.mat','RelativeIndex',1)
```

which produces the output in Figure 6.E.

Plotting of CW data is very similar, for details see the function reference.

4 Function reference

The folder 'infoForProgrammers' contains useful information if you need to modify or interface with the software.

4.1 convertTwotoneToOldFormat

```
convertTwotoneToOldFormat(infilename,outfilename)

DESCRIPTION:
  Convert data from TwoTone versions >=3.1 to old format.
INPUT:
  infilename  - Name of input data file
  outfilename - Name to save output data file.
OUTPUT: None
```

4.2 convertTwotoneToText

```
convertTwotoneToText(infilename,outfilename)

DESCRIPTION:
  Convert data to simple text file output.

INPUT:
  infilename  - Name of input data file
  outfilename - Name to save output text file.
OUTPUT: None
```

4.3 plotHist_ALEX

```
[absoluteIndex plotDataOut Ex En Sx Sn]= plotHistALEX(plotData,
   varargin)

DESCRIPTION:
  Plot ES histogram from TwoTone output data. Perform filtering on
     the data.

INPUTS:
  plotData - 3 options:
      'file1.mat' - name of single file for analysis
      {'file1.mat', 'file2.mat',...,'filen.mat'} - cell containing
         the names of mulitple files for analysis
      plotDataMatrix - the matrix output from plotHistALEX can be re
         -input (eg to select a different particle)
  filter (optional): (any number of these may be applied): {'
     filterType','filterParam','>' OR '<', Value}
      FilterType:  'fretData',
        FilterParam:   't_Dex', 't_Aex', 'DD', 'DA','AA', 'AA', 'E',
            'S'
      FilterType: 'all_aDetParam'
        FilterParam:  'nearestNeighbor'
      FilterType:    'DD_aDetParam'
      FilterType:    'DA_aDetParam'
      FilterType:    'AA_aDetParam'
        FilterParam:     'amplitude', 'sx', 'sy', 'eccentricity',

      FilterType:    'DD_aDetPos'
      FilterType:    'DA_aDetPos'
      FilterType:    'AA_aDetPos'
        FilterParam:     'X' 'Y'
      Range: '<' or '>'
      Value: Scalar number
  'NBin', nBin (optional) - number of bins in histogram
  ELim, [eMin eMax] (optional) - maximum and minimum fret values
  SLim, [sMin sMax] (optional) - max and min stoichiometry values
  'RelativeIndex', relativeIndex (optional) - n-th particle which
     meets thresholds in dataset. Multiple particles may be
     specified
  'AbsoluteIndex', [movieNo, ParticleNo] (optional) - n-th particle
     in m-th movie - multiple particles may be specified on separate
      rows.

OUTPUTS:
  absoluteIndex: [movieNo, ParticleNo] - vector of all plotted
     particles
  plotDataOut: structure containing plotted data.
    Contains:
      plotOutput.dataName ={'movieNo' 'particleNo' 'DDX' 'DDY' '
         t_Dex' 't_Aex' 'DD' 'DA' 'AD' 'AA' 'E' 'S'}
      plotOutput.data - matrix containing columns specified in
         dataName. Each row is one frame
  Ex, En - fret histogram data
  Sx, Sn - stoiciometry histogram data
```

4.4 plotHist_CW

```
[absoluteIndex plotDataOut Ex En]= plotHistCW(plotData, varargin)

DESCRIPTION:
  Plot E histogram from TwoTone output data. Perform filtering on
     the data.

INPUTS:
  plotData - 3 options:
      'file1.mat' - name of single file for analysis
      {'file1.mat', 'file2.mat',...,'filen.mat'} - cell containing
         the names of mulitple files for analysis
      plotDataMatrix - the matrix output from plotHistALEX can be re
         -input (eg to select a different particle) - in this case
         filtering arguments will be IGNORED!
  filter (optional): (any number of these may be applied): {'
     filterType','filterParam','>' OR '<', Value}
      FilterType:  'fretData',
        FilterParam:   't', 'D','A','E'
      FilterType: 'all_aDetParam'
        FilterParam:  'nearestNeighbor'
      FilterType:    'D_aDetParam'
      FilterType:    'A_aDetParam'
        FilterParam:     'amplitude', 'sx', 'sy', 'eccentricity',
      FilterType:    'D_aDetPos'
      FilterType:    'A_aDetPos'
        FilterParam:     'X' 'Y'
      Range: '<' or '>'
      Value: Scalar number
  'NBin', nBin (optional) - number of bins in histogram
  ELim, [eMin eMax] (optional) - maximum and minimum fret values
  'RelativeIndex', relativeIndex (optional) - n-th particle which
     meets thresholds in dataset. Multiple particles may be
     specified
  'AbsoluteIndex', [movieNo, ParticleNo] (optional) - n-th particle
     in m-th movie - multiple particles may be specified on separate
      rows.

OUTPUTS:
  absoluteIndex: [movieNo, ParticleNo] - vector of all plotted
     particles
  plotDataOut: structure containing plotted data.
    Contains:
      plotOutput.dataName ={'movieNo' 'particleNo' 'DX' 'DY' 't' 'D'
          'A' 'E'}
      plotOutput.data - matrix containing columns specified in
         dataName. Each row is one frame
  Ex, En - fret histogram data
```

4.5 plotTimetrace_ALEX

```
[absoluteIndex plotDataOut]= plotTimetraceALEX(plotData, varargin)

DESCRIPTION:
  Plot fret, stoichiometry and intensity timetraces from TwoTone
     output data. Perform filtering on the data.

INPUTS:
  plotData - 3 options:
      'file1.mat' - name of single file for analysis
      {'file1.mat', 'file2.mat',...,'filen.mat'} - cell containing
         the names of mulitple files for analysis
      plotDataMatrix - the matrix output from plotHistALEX can be re
         -input (eg to select a different particle) - in this case
         filtering arguments will be IGNORED!
  filter (optional): (any number of these may be applied): {'
     filterType','filterParam','>' OR '<', Value}
      FilterType:  'fretData',
        FilterParam:   't_Dex', 't_Aex', 'DD', 'DA','AA', 'AA', 'E',
            'S'
      FilterType: 'all_aDetParam'
        FilterParam:  'nearestNeighbor'
      FilterType:    'DD_aDetParam'
      FilterType:    'DA_aDetParam'
      FilterType:    'AA_aDetParam'
        FilterParam:     'amplitude', 'sx', 'sy', 'eccentricity',

      FilterType:    'DD_aDetPos'
      FilterType:    'DA_aDetPos'
      FilterType:    'AA_aDetPos'
        FilterParam:     'X' 'Y'
      Range: '<' or '>'
      Value: Scalar number
  'RelativeIndex', relativeIndex (optional) - n-th particle which
     meets thresholds in dataset. Multiple particles may be
     specified
  'AbsoluteIndex', [movieNo, ParticleNo] (optional) - n-th particle
     in m-th movie - multiple particles may be specified on separate
      rows.
'IntegrationTime', integrationTime (optional) - Specify the duration
    of a single frame of the raw data - used to plot the time axis.
   If not specified an integration time of 1 is used.

OUTPUTS:
  absoluteIndex: [movieNo, ParticleNo] - vector of all plotted
     particles
  plotDataOut: structure containing plotted data.
    Contains:
      plotOutput.dataName ={'movieNo' 'particleNo' 'DDX' 'DDY' '
         t_Dex' 't_Aex' 'DD' 'DA' 'AD' 'AA' 'E' 'S'}
      plotOutput.data - matrix containing columns specified in
         dataName. Each row is one frame
```

4.6 plotTimetrace_CW

```
[absoluteIndex plotDataOut]= plotTimetraceCW(plotData, varargin)

DESCRIPTION:
  Plot fret, and intensity timetraces from TwoTone output data.
     Perform filtering on the data.

INPUTS:
  plotData - 3 options:
      'file1.mat' - name of single file for analysis
      {'file1.mat', 'file2.mat',...,'filen.mat'} - cell containing
         the names of mulitple files for analysis
      plotDataMatrix - the matrix output from plotHistALEX can be re
         -input (eg to select a different particle) - in this case
         filtering arguments will be IGNORED!
  filter (optional): (any number of these may be applied): {’
     filterType’,’filterParam’,’>’ OR ’<’, Value}
    FilterType:  ’fretData’,
      FilterParam:   ’t’, ’D’,’A’,’E’
    FilterType: ’all_aDetParam’
      FilterParam:  ’nearestNeighbor’
    FilterType:    ’D_aDetParam’
    FilterType:    ’A_aDetParam’
      FilterParam:     ’amplitude’, ’sx’, ’sy’, ’eccentricity’,
    FilterType:    ’D_aDetPos’
    FilterType:    ’A_aDetPos’
      FilterParam:     ’X’ ’Y’
    Range: ’<’ or ’>’
    Value: Scalar number
    ’NBin’, nBin (optional) - number of bins in histogram
    ’RelativeIndex’, relativeIndex (optional) - n-th particle which
       meets thresholds in dataset. Multiple particles may be
       specified
    ’AbsoluteIndex’, [movieNo, ParticleNo] (optional) - n-th
       particle in m-th movie - multiple particles may be specified
       on separate rows.
    ’IntegrationTime’, integrationTime (optional) - Specify the
       duration of a single frame of the raw data - used to plot the
        time axis. If not specified an integration time of 1 is used
       .

OUTPUTS:
  absoluteIndex: [movieNo, ParticleNo] - vector of all plotted
     particles
  plotDataOut: structure containing plotted data.
    Contains:
plotOutput.dataName ={’movieNo’ ’particleNo’ ’DX’ ’DY’ ’t’ ’D’ ’A’ ’
   E’}
plotOutput.data - matrix containing columns specified in dataName.
   Each row is one frame
```

4.7 transformMovies

```
transformMovies()
```

```
DESCRIPTION:
  carry out image transforms and beads calibration

  You manually pick the control points using matlabs cpselect
     function
  The donor channel is presented on the left, acceptor channel on
     the right.
  Once you are finished, click "File">"Close␣Control␣Point␣Selection
     ␣Tool"
  and you will be presented with a transformed overlay of the two
     images
  If they look correct, accept the transform and it will be saved.

  We use a projective transform (read cp2tform documentation)
  If you want to change the transform type, you can modify the "
     align" function
  starting at line ~246.
  Returned channel is the transform which maps donor molecules into
     the acceptor channel
  ie
    x_A  = T x_D (in matrix notation
  You apply this transform by running
    x_A = tformfwd(TFORM,x_D);
  in matlab
  to go the other way, just run
    x_D = tformfwd(TFORM,x_A);
```

4.8 twotoneALEX

```
twotoneALEX()

DESCRIPTION: GUI-based analysis of ALEX TIRF-FRET data. See main
   text for details.
```

4.9 twotoneCMD

```
logFilePath = twotoneCMD(fileFilter, configPath)

DESCRIPTION: Analyse TIRF-FRET data in batch mode.%
INPUTS:
  - fileFilter:  string containg an expression for which all
     matching files in
    local directory will be analysed
```

```
 - configPath: path to the config file used to carry out the
    analysis.
Optional arguments:
 'OutputTextData', true/false
 'OutputTwotoneOldFormat', true/false
OUTPUTS:
 - logFilePath: path of the log file summarising results of analysis
```

4.10 twotoneCW

```
twotoneCW()

DESCRIPTION: GUI-based analysis of CW TIRF-FRET data. Very similar
   to twotoneALEX, discussed in main text.
```

5 Twotone output file specification

```
% analysis settings
twotoneData.settings.fileAppendString = '.fitResult.mat';
twotoneData.settings.imageSettings.twotoneVersion = '3.1.0alpha';
twotoneData.settings.imageSettings.alternationPeriod = 2;
twotoneData.settings.imageSettings.nChannel = 2;
twotoneData.settings.imageSettings.channelName = {'Dem','Aem'};
twotoneData.settings.imageSettings.channelImageLim=
   [[1,256,1,256];[257,512,1,256]];
twotoneData.settings.imageSettings.nADetChannel= 3;
twotoneData.settings.imageSettings.aDetChannelName={'DexDem', '
   AexAem', 'DexAem' };
twotoneData.settings.transformMatrixSettings.applyTform = 0;
twotoneData.settings.transformMatrixSettings.channelLinkage=
   {[1,2]}; %specifies which channel the transform is from and to (
   in this case 1->2) ;
twotoneData.settings.transformMatrixSettings.fileName= {''};
twotoneData.settings.fitSettings.algorithm= 'fixedGauss'; %OR: '
   fixedGaussEllipse','ring''freeGaussEllipse'
twotoneData.settings.fitSettings.fitParamName= {'windowRadius','
   minWidth','maxWidth'};
  %OR: 'innerCircleRadius','outerCircleRadius','ringTime'
  %OR: 'windowRadius','minWidth','maxWidth','maxPositionChange'
twotoneData.settings.fitSettings.param= [6, 1,2];
twotoneData.settings.fretDataName = {'t_Dex','t_Aex','DD','DA','AD',
   'AA','E','S'};
  % OR {'t','D','A','E'} for CW
```

```
twotoneData.settings.autoDetectSettings.averageFrameLim= [2, 10];
twotoneData.settings.autoDetectSettings.bandPassKernelDiameter= 3;
twotoneData.settings.autoDetectSettings.fitSubImageRadius= 3;
twotoneData.settings.autoDetectSettings.thresholds =  [15,15, 10];
  %OR  [15,15] for CW
twotoneData.settings.autoDetectSettings.clusterDistanceThresh  = 3;
twotoneData.settings.autoDetectSettings.nearestNeighbor.apply  = 0;
twotoneData.settings.autoDetectSettings.nearestNeighbor.thresh = 6;
twotoneData.settings.autoDetectSettings.ellipticity.apply  = 0;
twotoneData.settings.autoDetectSettings.ellipticity.thresh = 0.6;
twotoneData.settings.autoDetectSettings.PSFwidth.apply= 0;
twotoneData.settings.autoDetectSettings.PSFwidth.lim= [1, 2];
twotoneData.settings.autoDetectSettings.linkageFilter  = 'DexDem&&
   AexAem';
  % OR: 'DexDem', 'AexAem','DexAem','DexDem&&AexAem', 'DexDem&&
     DexAem', 'AexAem&&DexAem', 'DexDem&&AexAem&&DexAem','DexDem||
     AexAem||DexAem', 'DexDem||AexAem'

% analysis results
twotoneData.results.movieInfo.path: ''
twotoneData.results.movieInfo.name: 't1b18_scan1-g2r1_001.fits'
twotoneData.results.movieInfo.fitsHeader: [1x1 struct]
twotoneData.results.movieInfo.firstGreenFrame: 2
twotoneData.results.analysisDate: '101110'
twotoneData.results.aDetImages: {[256x256 int16]  [256x256 int16]
   [256x256 int16]}
twotoneData.results.aDetParamName: {'amplitude'  'sx'  'sy'  '
   eccentricity'}
twotoneData.results.transformMatrix.tform = {TFORM1} %allows storage
    of multiple tforms ie for 3color alex
twotoneData.results.aDetMolPositions.included =
    [Nx2 double]    [Nx2 double]    [Nx2 double]
twotoneData.results.aDetMolPositions.excluded =
    [Mx2 double]    [Mx2 double]    [Mx2 double]
twotoneData.results.fitParamName: {'A0'  'sx'  'sy'  'BG'  'X'  'Y'
    'theta'}

twotoneData.results.data: [NNx1 ] %data is an array containing all
   analysed molecules - one for each molecule

%autodetection results are output for each autodetection channel, (1
    per column). the order is defined by twotoneData.settings.
   imageSettings.aDetChannelName
```

```
twotoneData.results.data(1).aDetData.aDetPos: [3x2 double]
twotoneData.results.data(1).aDetData.aDetPos.aDetParam: [3x4 double]
    %param names are defined by twotoneData.results.aDetParamName
twotoneData.results.data(1).aDetData.nearestNeighborDist: 2.0024
twotoneData.results.data(1).aDetData.nParticle: [3x1 double]

twotoneData.results.data(1).intensity[nFRAME x nEmissionChannel] %
   photon counts in each emission channel. Order defined by
   twotoneData.settings.imageSettings.channelName
twotoneData.results.data(1).fitParam{nEmissionChannel}[nFrame x
   nFitParam] % cell containing fitting parameters for each emission
    channel. Order defined by
wotoneData.settings.imageSettings.channelName and twotoneData.
   results.fitParamName
twotoneData.results.data(1).fretData[nAlternationPeriod x nFretParam
   ] % fret data, order defined by   twotoneData.settings.
   fretDataName
```

References

1. Holden, S. J., S. Uphoff, J. Hohlbein, D. Yadin, L. Le Reste, O. J. Britton, and A. N. Kapanidis, 2010. Defining the limits of single-molecule FRET resolution in TIRF microscopy. *Biophys. J. (in press)* .

2. Bobroff, N., 1986. Position measurement with a resolution and noise-limited instrument. *Rev. Sci. Instrum.* 57:1152–1157.