The iMSD screen and description of the algorithms

This tutorial is an in-depth explanation of the algorithms used in the iMSD. Here we also show how to use the various options in the iMSD page.

The equations used in this tutorial can be found in Di Rienzo et al (PNAS 2013).

**Step 1.** Open the RICS page. Files from a camera based system (TIRF or SPIM) are read using readers in the RICS page or in the iMSD page. They access the same reader. The reader includes support for BIN, RAW, TIFF, SIF, SPE and other camera based formats. Read your data file. The data input for the iMSD analysis is in the form of a stack of time images (at regular times).

**Step 2.** In the tools menu of the RICS page click on iMSD. The iMSD page should show. First a 3D FFT algorithm is used to implement the calculation of the following correlation function.

\[ G(\xi, \chi, \tau) = \frac{\langle I(x, y, t) \cdot I(x + \xi, y + \chi, \tau + t) \rangle}{\langle I(x, y, t) \rangle^2} - 1 \]

This is the STICS (Spatio Temporal Image Correlation Spectroscopy) correlation function. Note that this function can also be called for two channel cross-correlation iMSD by selecting Ch1-Ch2 in the RICS page. Since the correlation function is computed using the 3D-FFT, the sizes in x, y and t must be a power of 2. The routine will check for the size and it will calculate only the largest 3D matrix of power of 2 size that fits into the input data matrix. For example, if you collect 500 frames in time, only 256 will be used, but if you collect 512 frames then all of them will be used. So during the data acquisition acquire either a power of 2 frames or slightly more. Click on “correlation function” to calculate the correlation function. In this tutorial the term ACF refers to the STICS formula above.

**Step 3.** Once the correlation function is calculated (it could take several seconds), proceed to the calculation of the iMSD by clicking on “Process iMSD”. You have the choice of two 2 algorithms, one (the default) is based on the fit using a Gaussian function of the vertical and horizontal lines of the ACF. The other is based on the calculation of the central moment of the ACF.

According to Di Rienzo et al, the ACF should have the form

\[ G(\xi, \chi, \tau) = g(\tau) \cdot \exp\left(-\frac{\xi^2+\chi^2}{\sigma^2(\tau)}\right) + g_\infty(\tau) \]

In the function, the shifts in the \(\xi\) and \(\chi\) directions are the parameters x[3] and x[4], which will provide the velocity in the x and y directions, respectively. So in the equation the \(\xi\) and \(\chi\) parameters are actually \(\xi + v_x\), \(\chi + v_y\) where \(v_x\) and \(v_y\) are the component of the velocity (in pixel units) in the x and y direction, respectively. The background level (the \(g_\infty\) term) is given by x[0] and the variance (the \(\sigma^2\) term) of the Gaussian is given by x[2] and it is equal for x and y. The amplitude of the Gaussian has no units (the \(g(\tau)\) term) is given by x[1]. X[1] and x[2] are forced to be positive. Note that the value of x[2]
(variance) has units of pixel squares. Each ACF is fitted to a Gaussian. Three algorithms are available for fitting the Gaussian ACF. These methods are the “moments” calculation, the fit of the vertical and horizontal lines of the ACF using a Gaussian function and the fit of the entire ACF also using a Gaussian. The code for the fit functions and for the moment analysis is given in the appendix.

The result of this fitting of the ACF is a series of values of the 5 parameters (2 velocities (x and y), background, amplitude and variance of the Gaussian) per each time delay (ACF). The variance as a function of the time delay will be fitted in the next operation to provide the “mode” of diffusion.

You can simulate an ACF sequence. For the simulation, you can change the parameters using the edit fields to generate different simulations. You can also add a velocity term.

The calculation of the Gaussian fit or the moments can act on the real data or on the simulated data. The simulation is activated by checking simulate Gaussian ACF and then clicking “Process iMSD”.

NOTE that the simulation routine changes the value of the ACF matrix, so the ACF must be recalculated if needed again. The Gaussian fits and the moment calculations should give the exact same values since we are using simulated ACFs and the values should be equal to the values entered for the various parameters. In case of real data, the fit of the ACF using the different algorithms could differ.

**Calculation of the “Modes” of diffusion**

The equation describing the different modes of diffusion is given below

\[ \sigma^2(\tau) = \sigma_0^2 + 4D_{\text{macro}}\tau + \frac{L^2}{3}\left(1 - e^{-D_{\text{micro}}\tau}\right) + V^2 \cdot \tau^2 \]

This equation describes the broadening of the variance of the Gaussian. L is the confinement size, \(D_{\text{micro}}\) is the initial slope of the iMSD in case of confinement (it must be multiplied by \(L^2\) to give a diffusion coefficient) and \(D_{\text{macro}}\) is the diffusion term.

The expansion of the confinement term for small delay times gives an “initial slope” of

\[ \text{Initial slope} = \frac{L^2}{3}D_{\text{micro}} \]

The initial slope could be associated with the diffusion coefficient in the confinement region. The calculation of the initial slope is done directly in SimFCS and the above formula is used to calculate the parameter “Initial slope” in the iMSD page.

The center of the ACF could also move due to the velocity term \(V\). This term adds to the \(\xi\) an \(\chi\) shifts rather than to the variance of the ACF Gaussian. However, it causes a curvature of the iMSD plot derived from the inverse of the amplitude of the ACF. So only the inverse amplitude term is used for deciding if there is directed motion.

For the analysis of different models, you can choose between the linear, the confined or the full model. If you choose the analysis of the full model, the analysis of the different models proceeds by first calculating only the quadratic term (velocity) from the inverse amplitude term. In case the iMSD (from the inverse amplitude term has an upward curvature (positive V) then the linear model (diffusion only) is...
also calculated. If the correlation coefficient of the quadratic fit is better (by at least 1.001) of that of the linear fit, then the directed motion model is accepted and no further calculation is done. Note that the velocity values are always calculated from the shift of the ACF, not from this upward curvature, so this procedure is only used for the rankings of the model.

In case the directed motion is not accepted, then 3 more models are tested. The calculation proceeds with the evaluation of the correlation coefficient of the fit for the following models:

0) The directed motion is detected by the positive curvature of the iMSD plot
1) Diffusion only, the linear term is the only term in the equation
2) Confined only, the exponential term is the only term in the equation
3) Confined and diffusion, both terms are included in the fit.

Note that model 3 will always be better than model 1 and 2 because it includes these two models. So the issue is to rank them and estimate how “better” one model is with respect to the others. For this purpose, we use the correlation coefficient of the fit.

The ranking is examined starting with the diffusion model, which is assumed by default. If the confined model gives a better correlation value (higher value) by at least 1.001, then the confined model is accepted. Then the full model is tested. It is accepted if the correlation parameter for the full model is better by at least 1.001 of the confined model. All the correlation parameters are shown in the screen and the parameters of the fit are listed in the memo control as well as the model that was accepted. The memo can be copied to a spreadsheet for plotting and further statistics. The plots are color coded for the various models.
Commands available in this page

Menu tabs

Open file: You can open a time stack or a previously calculated AFC

Correlation function: It calculates the ACF of the time stack. If the stack is large, it will employ several seconds

Process iMSD: calculate the mode corresponding the ACF that was calculated

Scan image: you must choose a size for the ROI to analyze. The ROI will be automatically scanned across the image and the results of the fit for each ROI are written to the memo and to a series of plots in the Scanform screen. This screen opens automatically when the scan image starts. To stop the analysis, press the STOP button

Analysis if fit: shows the Scanform where various plots and correlations can be obtained

Fields and checkboxes

Correlation points for fit: Once the correlation function has been calculated you can decide how many points in the time axis to use for the fitting of the mode of diffusion. The number of points must be less than half the number of the time slices.

Fame/s, pixel size and PSF: These values are used for converting pixel to microns and frame sequence to time shift. The PSF value is only used for simulation and initialization and its value is not subtracted from variance and is not used for any fit.

ACF points to average: In the calculation of the fir of the full surface you can fit the ACF in subsequent planes simultaneously. This reduces the noise substantially.

Subtract immobile (Checked). Do not change this value since the STICS function should only be sensitive to changes

Show each fit: During fitting of the ACF, will show the data and the fit for each time delay. Use only for testing. It slows down a lot the calculation. You can turn on/off this checkbox during the calculations.

Despike data: when checked, will use a median filter to despike the data. You can turn on and off this check box during calculation

Simulate Gaussian ACF: When checked, the ACF is substituted by a computed ACF according to the parameters of the simulation. The simulated ACF is 2-dimensional so that it can be fit with the different models.

Exclude first point: Using real data, the ACF at time zero contains the noise of the detector as well as of the immobile part. Generally, the fit of the real data ACF should exclude the first time point at zero delay time
**Check for dir motion**: When checked, every model is first tested for directed motion. If there is an upward curvature of the inverse amplitude and the model with the upward curvature gives a better fit than the linear model, then the directed motion model is accepted and no further calculations are done.

**Model to fit**: Forces using a specific model. The ranking of the models is active only if the full expression is used.

**Analysis method**: select the analysis method. The fit of V/H lines is generally more robust but it is slower than the moment analysis. The full surface fit is the slowest but also gives much less noise than the other models.

**Example of use**

Analysis of a simulation of random motion done with the SimFCS simulator without confinement with particles diffusing at 0.02um²/s. The pixel size is 0.05um, the frame time is 4.6frame/s and the PSF is 0.3um. After loading the bin file, the ACF was calculated and the fit was done using the fit of V/H lines. The results are shown below for an ROI of 128
Using the fit V/H or the fit of the surface, all parameters are recovered accurately and the correlation parameter of the fit is excellent. The moment analysis gives less accurate recovery of the parameters. This is due to the ACF that is quite broad and the moment analysis works poorly if the background cannot be determined accurately.
Scan analysis of ROI=64 using 8 time points and the full surface fitting algorithm.

The maps as well as the histograms of the various values found for the parameters are shown below.

While the sigma0 is in the expected range (0.09um2) the diffusion parameter recovered is lower than expected and noisy, probably due to the statistics of this simulation and the small ROI used for this analysis. The smaller value of the D_macro is due to the ACF that is very broad for this simulation and the ACF is partially chopped.

If we use the confinement model (which is not the model that was simulated) particles appear to be confined sometimes with an average confinement of 1um which is 20 pixels. This could be due to the small ROI used for the analysis. There is a strong inverse correlation between the size of the confinement and the diffusion value. Again, this could be due to statistics. Also for this simulated data set, we cannot use more than 8 delays due to the simulation that was done only for 256 frames.
**Analysis of TIRF data of CTx in CHO-k1 cells**

The file `128px zoom 2 bin 1 10ms frame take1.bin` (from Pierre Moens) was used for this demonstration. This file could be found in the SimFCS Tutorials directory in the LFD server.

Load the file and calculate the ACF for an ROI of 64 points. Center the ROI in the image. Below shows the iMSD plot obtained using the various modes of ACF analysis (Moments, VH lines and full surface, without and with average) of the same ACF. No despike was used. The analysis was done using 32 points of the ACF.

Moment Analysis  Fit of V/H  Surface fit (no avg)  Surface fir (avg 3)

The improvement provided by fitting the surface is quite evident. Of course the time required for the fir is substantially longer. So for a quick look the moment analysis could be OK but for accurate analysis the surface fit is much better.

**Scan analysis of the TIRF file**

This file was analyzed with an ROI of 32, using the surface fit and the linear model using 32 points for the time delays. The maps and histograms are shown below.

Sigma0  Dmacro  Correlation coefficient
In this linear analysis the correlation coefficient is pretty good except in the regions of very low intensity outside the cells. The histogram values and the maps are also very smooth. No one point is missing in this fit;

Next we did the analysis using the confined model still using the surface fit with an average of 3 in the ROI of 32 using 32 time delay points.

The Sigma0 has a similar distribution. There are some regions in which the confinement seems to fit better with an average of about 3 um. The D in these regions the average D is about 10um2/s

Finally, we tried the Full model with rankings

So far the analysis has been fully unsupervised. However, we noticed that the analysis depends somehow from the length of the time delays considered. Clearly in some regions the ACF is poorly determined so that long delays are very noisy.
The pie plot shows that only in few instances, 10% there is a need for the partial confinement model and that 51% of the regions have confined model. Below is the color coded map where the various models are found.

Level 3 red-> partial confinement

Level 2 green-> confined

Level 1 blue-> diffusion
Appendices

function myfunc2(xx:single;x:globalparam):double; //this is the iMSD function
begin
   result:=abs(x[4])*xx*xx+(sqr(x[3])/3)*(1-exp(-xx*abs(x[2])))+4*abs(x[1])*xx+x[0];    //use for partially confined  L=x[3],  Dmicro=x[2],  Dmacro=x[1],  V^2=x[4]
end;

function F_sqm8(var x:globalparam):double;  //This is used to fit the term from the msd
var j:Integer;
   sqm:double;
   y,yy:double;
begin
   sqm:=0;
   x[1]:=abs(x[1]);
   x[2]:=abs(x[2]);
   x[3]:=abs(x[3]);
   x[4]:=abs(x[4]);
   for j:=ss to ur_index-1 do
       begin
           y:=sqr(myfunc2(j*sx,x)-myyv[j]);
           sqm:=sqm+y;
           ur^[j]:=y;
       end;
   F_sqm8:=sqm/ur_index;
end;

function F_sqm9(var x:globalparam):double;  //This is also to fit the term from the g0
var j:Integer;
   sqm:double;
   y,yy:double;
begin
   sqm:=0;
   x[1]:=abs(x[1]);
   x[2]:=abs(x[2]);
   x[3]:=abs(x[3]);
   x[4]:=abs(x[4]);
   for j:=0 to ur_index-1 do
       begin
           y:=sqr(myfunc2(j*sx,x)-myav[j]);
           sqm:=sqm+y;
           ur^[j]:=y;
       end;
   F_sqm9:=sqm/ur_index;
end;

function Hline(i:integer;var x:globalparam):double;  //this is the function for the H ACF
begin
\[ x[1] := \text{abs}(x[1]); \quad \text{// must be positive amplitude of gaussian} \]
\[ x[2] := \text{abs}(x[2]); \quad \text{// variance of gaussian} \]
\[ \text{result} := x[0] + x[1] \times \exp(-\text{sqr}(i-x[3])/x[2]); \quad \text{// add background term, x[2] is the variance in} \]
\[ \quad \text{pixels, but myav is in microns} \]
\]
\end{end};

\begin{function} \text{Vline}(i: \text{integer}; \text{var} \ x: \text{globalparam}): \text{double}; \quad \text{// this is the function for the V ACF} \end{function}
\begin{begin}
\[ x[1] := \text{abs}(x[1]); \quad \text{// must be positive amplitude of gaussian} \]
\[ x[2] := \text{abs}(x[2]); \quad \text{// variance of gaussian} \]
\[ \text{result} := x[0] + x[1] \times \exp(-\text{sqr}(i-x[4])/x[2]); \quad \text{// add background term} \]
\end{end};

\begin{function} \text{mysurf}(j,i: \text{integer}; \text{var} \ x: \text{globalparam}): \text{double}; \quad \text{// this is the function for the surface} \end{function}
\begin{begin}
\[ x[1] := \text{abs}(x[1]); \quad \text{// must be positive amplitude of gaussian} \]
\[ x[2] := \text{abs}(x[2]); \quad \text{// variance of gaussian} \]
\[ \text{result} := x[0] + x[1] \times \exp(-\text{sqr}(i-x[3])+\text{sqr}(j-x[4])/x[2]); \quad \text{// add background term} \]
\end{end};

\begin{function} \text{mfuncH}(k,i: \text{integer}): \text{single}; \quad \text{// used for the plots} \end{function}
\begin{begin}
\[ \text{result} := \text{mybg}[k] + \exp(-\text{sqr}(i-\text{velx}[k]) / \text{myyv}[k]) / \text{myav}[k]; \]
\end{end};

\begin{function} \text{mfuncV}(k,i: \text{integer}): \text{single}; \end{function}
\begin{begin}
\[ \text{result} := \text{mybg}[k] + \exp(-\text{sqr}(i-\text{vely}[k]) / \text{myyv}[k]) / \text{myav}[k]; \]
\end{end};

\begin{function} \text{F_sqm10}(\text{var} \ x: \text{globalparam}): \text{double}; \quad \text{// this is the main function that fits the H/V lines} \end{function}
\begin{begin}
\[ \text{var} \ i,j: \text{integer}; \]
\[ \text{begin} \]
\[ \text{result} := 0; \]
\[ i := 0; \]
\[ \text{if} \ x[1] = 0 \text{ then } x[1] := 0.01; \]
\[ \text{if} \ x[2] = 0 \text{ then } x[2] := 0.01; \]
\[ \text{for } j = -\text{siza to siza do} \]
\[ \text{begin} \]
\[ \text{if} \ (j=0) \text{ then continue; // do not include the center point} \]
\[ \text{ur}^[i] := \text{Hline}(j,x) - \text{mhim}^[kk][0+y\_origin, j+x\_origin]); \quad \text{// these values are in pixel units, so} \]
\[ \text{hline must be in pixel units} \]
\[ \text{if} \ \text{abs(\text{ur^[i]})} > 1e15 \text{ then } \text{ur^[i]} := 1e15; \]
\[ \text{result} := \text{result} + \text{sqr}(\text{ur^[i]}) \]
\[ \text{inc}(i); \]
\[ \text{ur^[i]} := \text{Vline}(j,x) - \text{mhim}^[kk][j+y\_origin, 0+x\_origin]); \]
\[ \text{end} \]
\[ \text{end}; \]
\[ \text{end}; \]
\]
if abs(ur^[i])>1e15 then ur^[i]:=1e15;
result:=result+sqr(ur^[i]);
inc(i);
end;
i:=i-1;
if i>ur_par then result:=result/(i-ur_par) else result:=0;  //this is the sum of squares
application.processmessages;
ur_index:=i;
end;

function F_sqm11(var x:globalparam):double; //this is the function that fits the entire surface
var i,j,k:integer;
begin
result:=0;
k:=0;
if x[1]=0 then x[1]:=0.01;
if x[2]=0 then x[2]:=0.01;
for j:=siza to -siza do
for i:=siza to -siza do
begin
if (j=0) and (i=0) then continue; //do not include the center point
ur^[k]:=(mysurf(j,i,x)-mhim^[kk][j+y_origin,i+x_origin]); //these values ar in pixel units,
so hline must be in pixel units
if abs(ur^[k])>1e15 then ur^[k]:=1e15;
result:=result+sqr(ur^[k]);
inc(k);
end;
k:=k-1;
if k>ur_par then result:=result/(k-ur_par) else result:=0;  //this is the sum of squares
application.processmessages;
ur_index:=k;
end;

procedure simulateACF; // this function simulates the 2-dimensional ACF
var s,j,i,k,siz22:integer;
sigma,s0,t,L,Dmicro,Dmacro,V,vv:single;
begin
s:=strtoint(icsform.combobox1.text);
s:=s div 2;
siz22:=s div 2;
sigma:=strtofloat(iMSDform.edit3.text);//this is in microns
L:=strtofloat(IMSDform.Edit8.Text);
Dmacro:=strtofloat(IMSDform.Edit9.Text);
Initial_slope:=strtofloat(IMSDform.Edit14.Text);
If L>0 then Dmicro:=initial_slope*12/sqr(L) else Dmicro:=0;
V:=strtofloat(IMSDform.Edit6.Text);
for k:=0 to 128 do
begin
kk:=k;
t:=k*frametime; //this is in seconds
s0:=(sqr(L)/3)*(1-exp(-t*Dmicro))+4*Dmacro*t+sqr(sigma); //the units here are in microns
s0:=s0/sqr(pixel);
vv:=V*t/pixel;
for j:=s-siz22 to s+siz22 do           //for testing
  for i:=s-siz22 to s+siz22 do
  begin
    mhim^[k][j,i]:=0.0001+(1/s0)*exp(
      -(sqr(i-s+vv)+sqr(j-s+vv))/(s0)); //simulated equation for testing
  end;
end;
ur_index:=sqr(2*(s-siz22)-1);
save_p_fit[0]:=0.0001;                  //this is done to initialize the variables
save_p_fit[2]:=sqr(sigma/pixel);
save_p_fit[1]:=1/sigma;
save_p_fit[3]:=vv;
save_p_fit[4]:=vv;
end;

procedure calculate_moments(kk:integer;var p_fit:globalparam); //This routine calculates the
central moments for the H and V lines of the ACF
var i,j,ii,jj,bkn,s,siz22,b:integer;
ar,m:darray;
d,sd,std,av,mins,maxs,bkg:single;
im:mydatasingle;
const c=5;
begi
s:=strtoint(icsform.combobox1.text);
s:=s div 2;
siz22:=s div 2;
// siz22:=s-16;
fillchar(m,sizeof(darray),#0);
fillchar(ar,sizeof(darray),#0);
move(mhim^[kk],im,sizeof(mydatasingle)); //move this slice of teh ACF to the temo image im

bkg:=0;
bkn:=0;
for b:=2 to c do
  for j:=b+(s-siz22) to 2*s-siz22-b do
    begin
      bkg:=bkg+im[j,(s-siz22)+b];
      bkg:=bkg+im[j,2*s-siz22-b];
      bkn:=bkn+2;
    end;
for b:=2 to c do
for i:=b+(s-siz22) to 2*s-siz22-b do
begin
bkg:=bkg+im[(s-siz22)+b,i];
bkg:=bkg+im[2*s-siz22-b,i];
bkn:=bkn+2;
end;
if bkn<>0 then bkg:=bkg/bkn;

/////////// to here is the calculation of the background
maxs:=0;                 // initialize min,max amplitude and the moments to zero
mins:=10000000;
m[0,1]:=0;
m[1,0]:=0;
m[0,2]:=0;
m[2,0]:=0;
sd:=0;
//moments for the horizontal line
for jj:=s-siz22 to s+siz22 do
begin
  d:=(im[jj,s]-bkg);                       // data
  sd:=sd+d;                               //sum of intensities
  m[0,1]:=m[0,1]+jj*d;                     //first moment
  m[0,2]:=m[0,2]+jj*jj*d;                  //second moment
  if im[jj,s]>maxs then maxs:=im[jj,s];    //get min-max
  if im[jj,s]<mins then mins:=im[jj,s];
end;
if sd<>0 then                                //normalize the first central moment and save
begin
  ar[0,1]:=((m[0,1]/sd)-s);                  //first central moment, this is the velocity along
                                          //centered at the center of teh ACF
  ar[0,2]:=((m[0,2]/sd)-sqr(m[0,1]/sd));     //this is the second moment corrected for the
                                          //change in the center due to velocity
end;

//before exit, save in the common parameter vector
p_fit[0] := bkg;                          // first value is background
p_fit[2] := (abs(ar[2, 0]) + abs(ar[0, 2]));  // the total variance is the sum of the two variances
p_fit[3] := ar[1, 0];                      // velx
p_fit[4] := ar[0, 1];                      // vely
if p_fit1[2] <> 0 then ar[0, 0] := sd/sqrt(p_fit[2]*pi) else ar[0, 0] := maxs; // the amplitude depends
both on the sum of pts and on the variance
// ar[0, 0] := maxs - bkg;
p_fit[1] := ar[0, 0];
end;