Oligomerization Dynamics Using Dynamic Spatial Distribution Analysis

McGill University
Department of Physics

Authors:
David Herrera, Paul Wiseman

Abstract

A new technique called Dynamic Spatial Distribution Analysis (DSpIDA) was developed to measure the diffusion state of an oligomer with a given number of particles using fluorescent labels. The technique is based on fitting the cumulative distribution function of averaged regions of interest around particles sharing a common diffusion state. The theory was confirmed by running simulations for 2D diffusion of particles, with different point spread function radii, intensity amplitudes, and particle densities. The analysis was able to recover the input parameters. Lastly, HEK-293 cells were transected with m3 and mGlu3 receptors tagged with fluorescent labels. For each receptor the cells were treated with two experimental drugs, LY379268 and LY341495. A link between the amplitude factor, which describes the aggregation state of molecules and the diffusion state was found. The experiments led to suggestions on how to improve the technique.

April 20, 2015

1 Introduction

It's possible to isolate proteins but the information gained about the intracellular dynamics is limited for such a system. It is therefore important to study cells in their vivo state so a more complete picture about intracellular interactions can be acquired. Fluorescent microscopy is often used to obtain images of cells in their vivo state, in this process proteins are tagged with fluorescent labels and transfected into cells, images of the emitting fluorophores are obtained from the intensities given of by them. The aggregation state of oligomers and their dynamics are some of the properties obtained from the tagged proteins. Study of cell surface protein receptors has become an important step in understanding many human disorders. For instance, a study done on schizophrenia showed that the dimerization of metabotropic glutamate was altered for patients with the disease when compared to control subjects. [1]
One of the major challenges present in fluorescent microscopy is the relatively small size of the oligomers, (∼4nm) as compared to the resulution of the microscope. The resolution of the microscopes are set by the point spread function (PSF). The CCD's cameras, which are used to acquired the data, have a point spread radius that is at least two orders of magnitude bigger than the oligomers. Instead, study of intensity values is performed as particles tend to conglomerate into clusters while diffusing through the cell. The intensity given off by the cluster is linearly proportional to the number of fluorescent particles inside the cluster.
Previous methods developed such as FRAP and SPT had been able to measure protein transport properties in cells.[2][3] Other methods such as Brightness Analysis and Photon Counting Histogram have been developed to measure the protein aggregation. No methods though exist to measure both quantities simultaneously thus attempting to cross correlate these two metrics.[4][5]
This paper presents a new technique called Dynamic Spatial Intensity Distribution Analysis (DSpIDA), it was derived from a previously developed technique SpIDA.
SpIDA fits super-poissonian distributions to intensity histograms, and finds aggregation states of proteins using the quantal brightness of the fluorophores.[6] DSpIDA has the advantage over SpIDA in that it uses smaller regions of interest around the particles thus allowing it to capture not only aggregation states of the clusters, but also information about the dynamics of the cluster.

2 Theory

DSpIDA is based on fitting the cumulative distribution function to pixel intensities of an averaged region of interest around a tracked particle cluster. DSpIDA returns information on the number of particles in a cluster based on and their respective diffusion state based on the intensity given off by the clusters.

2.1 Mathematical Basis of DSpIDA

2.1.1 Intensity of a Pixel

Any intensity value $i(\vec{r}_{xy})$ in a CLSM[1] image or a TIRFM [2] image can be described by the convolution between the particles present in the image, each represented by a point source and the point spread function of the microscope (PSF) $I(\vec{r})$. The point spread function of the microscope is approximated to be a Gaussian beam with a e^2 PSF radius, σ, and a focal point intensity amplitude I_o.[7]

$$I(\vec{r}) = I_o e^{-2\frac{|\vec{r}|^2}{\sigma^2}} \tag{1}$$

In an system of N particles the intensity at any point in space is given by:

$$i(\vec{r}_{xy}) = W(\vec{r_{xy}}) \times \left[\sum_{j=1}^{N} I(\vec{r_{xy}}) * \delta(\vec{r_{xy}} - \vec{r_j}) \right] q_j \Theta_j(t) \tag{2}$$

The $W(\vec{r_{xy}})$, is a window function that defines the boundaries of the ROI, it takes the value of

[1]Confocal Laser Scanning Microscopy
[2]Total Internal Reflection Fluorescent Microscopy

one if $\vec{r}_{xy}$ is part of the ROI and zero otherwise. The q_j represents the ratio of photons emitted by the fluorophore particles vs. the photons absorbed by them. Lastly the $\Theta_j(t)$ parameter represents the emission state of the fluorophore. It is set to one if the particles is emitting and zero if is not emitting. This phenomenon has a variety reasons, mostly related to the stochastic nature of quantum mechanics.[8] For simplicity it is assumed that the value of the q_j parameter and the $\Theta_j(t)$ parameter are set to 1. Taking these assumptions into account and carrying out the convolution:

$$i(\vec{r}_{xy}) = W(\vec{r_{xy}}) \times \sum_{j=1}^{N} I(\vec{r_{xy}} - \vec{r_j}) \tag{3}$$

Equation 3 represents the contribution in intensity to a point in space of all the particles interacting with the microscope inside the ROI. Since the image is made up of pixels, equation 3 must be integrated over the area of a pixel in order to be able to obtain the pixel intensity: [9]

$$i(n_x, n_y) = \int_{n_x\Delta x - \frac{\Delta x}{2}}^{n_x\Delta x + \frac{\Delta x}{2}} \mathrm{d}x \int_{n_y\Delta y - \frac{\Delta y}{2}}^{n_y\Delta y + \frac{\Delta y}{2}} \mathrm{d}y i(\vec{r}_{xy}) \tag{4}$$

Where n_x and n_y are the pixel number with respect to the centered pixel of the region of interest.

2.1.2 Average Intensity Values of a ROI

The expected value of a pixel intensity over time, $\langle i(n_x, n_y)\rangle_t$, will give the intensity value likely to be obtained for each pixel intensity inside the region of interest. By assuming ergodicity, the time average can be transformed into an ensemble average.[10][11]

The ensemble average of a mean image intensity can be obtained using the probability density, $P_j(\vec{r}_j)$, of a particle j with a position r_j within the sample area:

$$\langle i(\vec{r}_{xy})\rangle = \int_{-\infty}^{\infty} \mathrm{d}\vec{r}_j i(\vec{r}_{xy}) P_j(\vec{r}_j) \tag{5}$$

There are two averaged ROI of particular interest. The first one is calculated for a static ROI on the image, and the second one is for a dynamic ROI tracking a particle cluster.

For the static ROI, assuming a homogeneous system, the probability density of a single particles is given by $P_j(\vec{r}_j) = A^{-1}$, where A^{-1} is the inverse of the sampled area. Using Eq.1 and Eq. 3 the ensemble average for a static ROI is:

$$\langle i(\vec{r}_{xy})\rangle_{sROI} = \int_{-\infty}^{\infty} \mathrm{d}\vec{r}_j W(\vec{r_{xy}}) \sum_{j=1}^{N} I(\vec{r_{xy}} - \vec{r_j}) \frac{1}{A} \tag{6}$$

$$= \frac{I_o W(\vec{r_{xy}})}{A} \sum_{j=1}^{N} \int_{-\infty}^{\infty} \mathrm{d}\vec{r}_j exp\Big(-2\frac{\mid \vec{r}_{xy} - \vec{r}_j \mid^2}{\sigma^2}\Big)$$

$$= \frac{N I_o W(\vec{r_{xy}}) \pi\sigma^2}{2A} \tag{7}$$

Eq.7 is independent of the position in space and represents a constant offset intensity given by all the particles in the sample area.

For the dynamic ROI tracking a particle cluster, there are two probability densities considered: Either the particle j^{th} is located in center of the ROI or the particle is located anywhere else in the sample area, the corresponding probability densities are:$P_j(\vec{r}_j) = \delta(\vec{r}_j)$ and $P_j(\vec{r}_j) = A^{-1}$ respectively. Assuming there are α particles in the center of the ROI and $N - \alpha$ elsewhere the expected intensity is:

$$\langle i(\vec{r}_{xy})\rangle_{dROI} = \int_{-\infty}^{\infty} \mathrm{d}\vec{r}_j W(\vec{r_{xy}}) \Big[\alpha I(\vec{r_{xy}} - \vec{r_j})\delta(\vec{r}_j) + \sum_{j=1}^{N-\alpha} I(\vec{r_{xy}} - \vec{r_j}) \frac{1}{A}\Big]$$

$$= \alpha I_o W(\vec{r_{xy}}) \Big[exp\Big(-2\frac{\mid \vec{r}_{xy} \mid^2}{\sigma^2}\Big) + \frac{(\frac{N}{\alpha} - 1)}{A} \frac{\pi\sigma^2}{2}\Big] \tag{8}$$

Using Eq.4 to obtain pixel intensity values and assuming that N¿¿α[3]:

$$\langle i(n_x,n_y)\rangle_{dROI} = \int_{n_x\Delta x-\frac{\Delta x}{2}}^{n_x\Delta x+\frac{\Delta x}{2}} \mathrm{d}x \int_{n_y\Delta y-\frac{\Delta y}{2}}^{n_y\Delta y+\frac{\Delta y}{2}} \mathrm{d}y \Big[\alpha I_o W(\vec{r_{xy}}) exp\Big(-2\frac{\mid \vec{r}_{xy}\mid^2}{\sigma^2}\Big) + \frac{(N-1)}{A}\frac{\pi\sigma^2}{2}\Big]$$

$$= \Big\{erf\Big(\frac{\sqrt{2}(n_x\Delta x+\frac{\Delta x}{2})}{\sigma}\Big) - erf\Big(\frac{\sqrt{2}(n_x\Delta x-\frac{\Delta x}{2})}{\sigma}\Big)\Big\} \Big\{erf\Big(\frac{\sqrt{2}(n_y\Delta y+\frac{\Delta y}{2})}{\sigma}\Big) - erf\Big(\frac{\sqrt{2}(n_y\Delta y-\frac{\Delta y}{2})}{\sigma}\Big)\Big\}. \Big(\frac{\pi}{8}\sigma\alpha I_o\Big) + \frac{(N-1)I_o\pi\sigma^2\Delta x\Delta y}{2A} \tag{9}$$

In Eq.9 for notation simplicity the window function was dropped, a valid assumption as long as n_x and n_y stay inside the ROI. There are three parameters of interest: The PSF radius, σ, the amplitude factor αI_o, where α is the number of particles in the cluster and the constant offset.

2.2 CDF Model

Using equation 9, a cumulative distribution function is obtained using the size of the total area of the ROI, XY, as the normalizing factor:

$$cdf(I) = \frac{\sum_{n_x,n_y}\theta(\langle i(n_x,n_y)\rangle \le I)}{XY} \tag{10}$$

Eq. 10 will be used as the model equation for the data, the three parameters described in Eq.9 will be the fitting parameters for the model.

2.3 Intuition

The idea of DSpIDA is to create small enough regions of interest so that mostly the cluster in the middle of the ROI is present in the region of interest at any point in time[4], the contribution in intensity of the first term in Eq.9 would then be determined by this cluster, while the offset term is determined by all the other particles around. Therefore the parameter αI_o is the value of interest, since α represents the number of particles in the cluster.

3 Experimental

3.1 Simulations

A software generating particles undergoing 2D free diffusion was made. The simulations had as parameters, the total number of particles, the PSF radius, the number of frames, the diffusion coefficient, and the constant offset. Additionally a background noise was added to each image. The dimensions of the frames were 64 by 64 pixels. The particles were distributed randomly initially. From there, the random movement of a particle from one frame to the next was determined by a Gaussian distribution with a standard deviation of $\sqrt{2D\Delta t}$ in each of the two coordinates. In the equation D is the diffusion coefficient, and Δt is the time step between frames. Once all the particles had been position in all the frames [5], an image series was generated for the system using a PSF radius of 4 pixels and an intensity amplitude I_o at the focal point of 0.0398(arb.). Each particle was then placed at the focal point and the intensity of each pixel was recorded as the contribution of each particle's interacting with the point spread function. Lastly an offset of 1(arb.) was added to the image series.

3.2 Real Data

The images were obtained from the collaborators in Japan[6] through TIRF. HEK-293 cells, embryonic kidney cells were transfected with two different proteins, a EGFP protein tagged with a TMR fluorescent label and a GPRC5B receptor protein, which is a green fluorescent protein.[12]

[3]Reasonable assumption for the systems considered

[4]The ROIs have to be big enough compared to the point spread function, as the PSF sets the resolution of the image.[7]

[5]No need to use a tracking software to track the particles as the position of every particle at each point in time is known.

[6]Data obtained from Osaka University, at Professor's Masataka Yanagawa Cellular Informatics Laboratory

The proteins were further treated with two experimental drugs namely with 1nM of LY341495 and 1nM of LY379268. A basal control trial was kept for comparison. There were around 60 independent trials for each receptor, around 20 trials [7] for each treatment. The images obtained per trial had 2000 frames each. For each image series a rolling ball filter and a 2 frame average was performed using the running z-projector image plug-in.[13]

A home made particle tracking software based on Hidden Markov Models was made by the collaborators, 500 particles in the images were tracked throughout the image series from the initial moment to the moment when the fluorophores bleached. To each particle the software assigned an specific diffusion state assuming four different diffusion states, 1 being the slowest and 4 being the fastest[8].

4 Analysis

The analysis performed on the data is divided as follows:

1. Upload images to MATLAB
2. Single particle spatial and temporal information is obtained from the tracking software results.
3. Localize a region of interest around each particle in all the frames.
4. Take the average of all the ROIs containing a common diffusion state.
5. Create histograms and obtain a cumulative distribution function from each averaged ROI corresponding to a particular diffusion state.
6. Fit the model to CDFs and obtained parameter values that minimize the non-linear least square function for the model[9].

[7]Some actually had 19 or 18, others had 20, or 21

[8]Speeds were in the range of 1 μm/s to 100 μm/s

[9]The software minimizes the function $\chi^2 = \sum(model - Data)^2$

4.1 Software Considerations

The software does not find the global minimum always, it finds ten local minimums and takes the one with the lowest least square function.

The intensity units returned by the software do not matter and are arbitrary since the parameter of interest is α.

5 Results & Discussion

5.1 Simulations

The analysis outlined in section 4 was applied to the simulations of 2D free particle diffusion with the exception of step 2 as explained previously.

Figure 1 is a typical frame from an image series obtained from the 2D diffusion simulation of 100 particles.

Figure 2. shows the resulting image from an av-

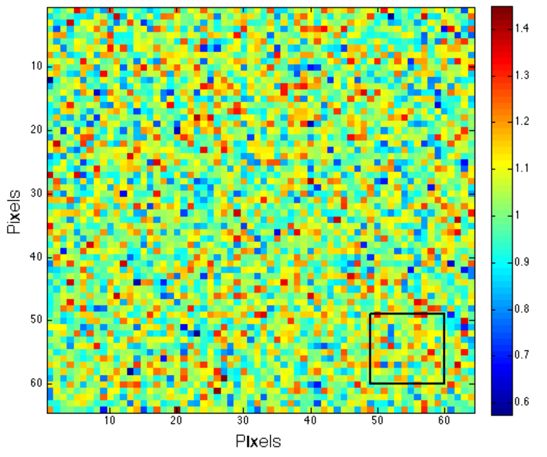

Figure 1: Image frame of 64 by 64 pixels^2 obtained from the simulation of 100 particles undergoing free diffusion, the black rectangle is a region of interest tracking one of the particles

eraged static ROI.The image is mostly constant in accordance with Eq.7, the discrepancy arrives from the noise added to the image. A circular average filter was applied to the image.

Figure 3. shows the resulting image from an averaged ROI tracking a particle, the image takes the shape of a single particle cluster interacting with the PSF, the result was as expected according to

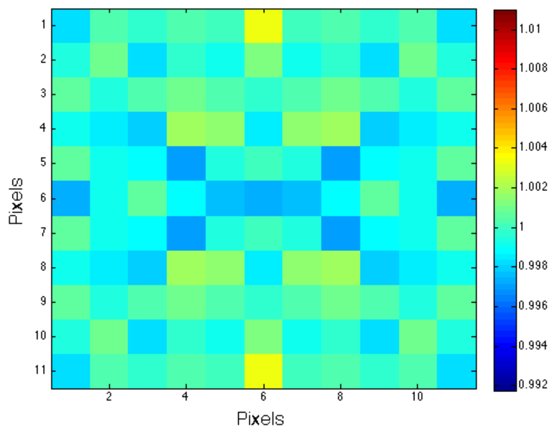

Figure 2: Averaged ROI static throughout all the frames, the result is in accordance with offset predicted by the theory

Eq.9 where the contribution of intensity inside the averaged ROI was mostly from the cluster in the middle.

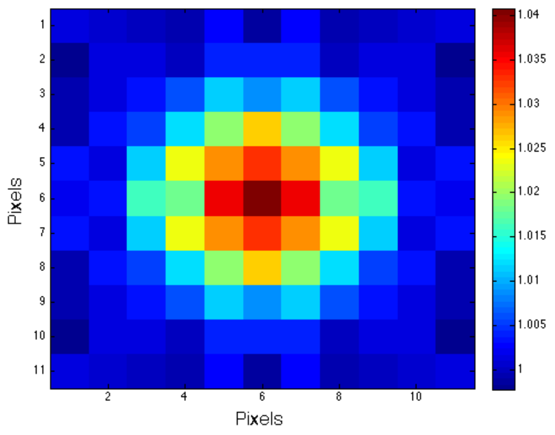

Figure 3: Average dynamic ROI obtained from a particle tracking, as expected the intensity values take the shape of a particle cluster interacting with the PSF of the microscope.

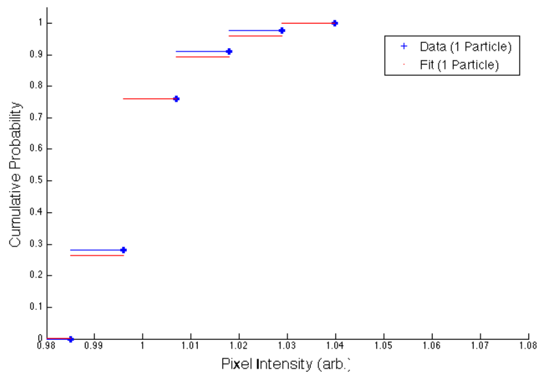

Figure 4: The CDF obtained for one trial of the simulation for a particle undergoing 2D diffusion plotted against the model obtained from the analysis on this data. The parameter values obtained were $\sigma = 4.000$ pixels, $offset = 1.0000$(arb.), and $\alpha I_o = 0.0400$(arbs.), the least-square value was of 0.0003.

Figure 4 and Figure 5 show the CDFs obtained for simulations of 1 particle and 100 particles respectively on one trial.
For each simulation 20 independent trials were made. The parameters obtained are depicted in Table 1. The input parameters for the simulations were $\sigma = 4.0$pixels, $offset = 1.0$(arb.), and $\alpha I_o = 0.0398$(arbs.). All the parameters agree within a 95% confidence interval for both

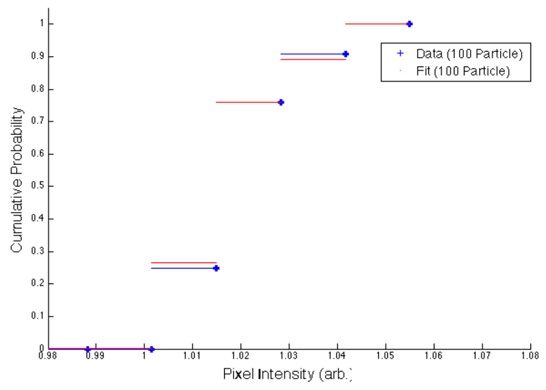

Figure 5: CDF obtained for one trial of the simulation of 100 particles plotted against the model obtained from the analysis on this data. The parameter values obtained were $\sigma = 4.00$ pixels, $offset = 1.000$(arb.), and $\alpha I_o = 0.040$(arbs.), the least-square value was of 0.0008.

simulations. The error obtained came from randomness of the noise simulated in each image, and the search of the global minimum.

Simulation	σ(pixels)	Offset (arb.)	σI_o(arb.
1 Particle	4.000(8)	1.0002(5)	0.040(6
100 Particles	4.00(9)	1.013(9)	0.046(5

Table 1: Values of the parameters of the model function obtained over 20 trials for each simulation.

5.2 Real Data

The analysis outlined in section 4 was performed on the EGFP protein and the GPRC5B protein data for the three treatments (two drugs and the control). The data for the EGFP protein was taken over two days. The basal had 18 independent trials, the LY341495 and the LY379268 treatments had 20 independent trials each. The data for the GPRC5B protein was taken over three days. The basal and the LY341495 treatments had 20 independent trials each, while the LY379268 treatments had 19 independent trials.

Figure 6 shows the results of the PSF radius

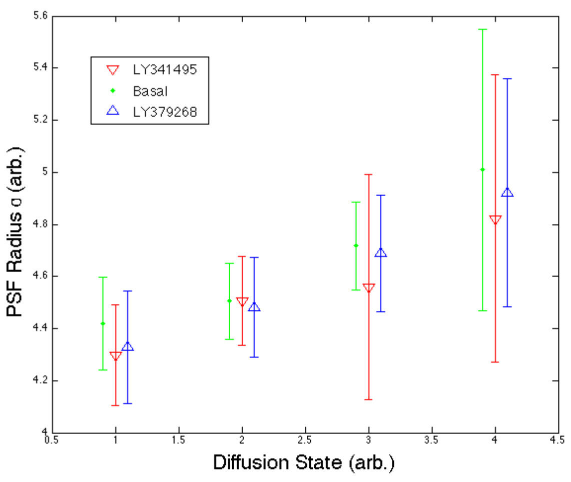

Figure 6: PSF radius vs. diffusion state graph for the EGFP protein and its three treatments.

for the EGFP protein, all the values obtained for the 3 treatments were within two standard deviations of each other.

Figure 7 shows the results of the PSF radius for the parameter for the GPRC5B protein, again the values obtained for each treatment are in agreement as they are within two standard deviations of each other.

For both proteins it can be noted though that the radius seems to increase as well as the error as the particle cluster gets faster. The broadening of the PSF radius could be due to errors in the particle tracking, if the particle tracking assigned the center of the particle to be in the wrong pixel, then the rotational average would

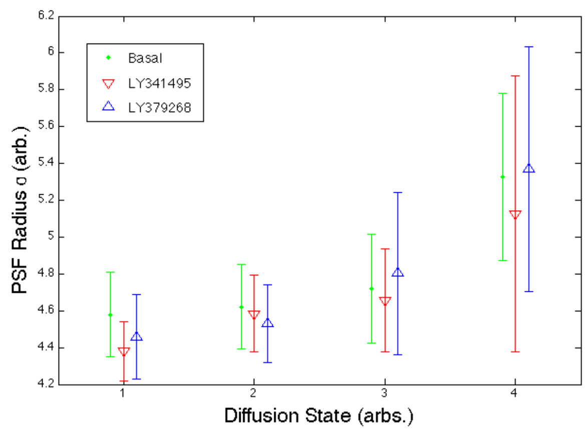

Figure 7: PSF radius vs. diffusion state graph for the GPRC5B protein and its three treatments.

have the effect of broadening the PSF radius. This could come from a defocus due to the high speed diffusion of the tracked particles, since the tracking was based on a two frame average.[13] A better tracking or smaller time steps would be required to improve the results for higher speeds.

Figure 8 and Figure 9 show the results of

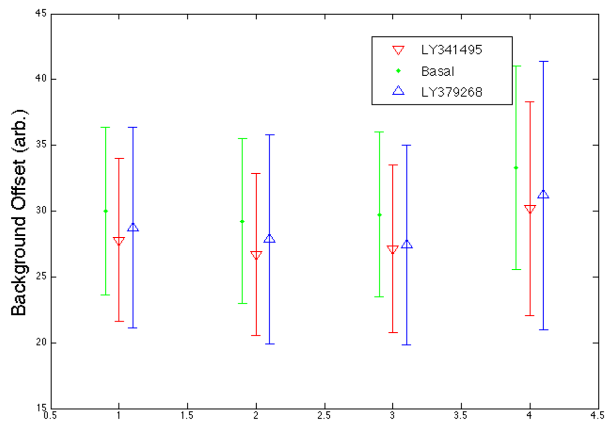

Figure 8: Background offset vs. diffusion state graph for the EGFP protein and its three treatments.

the offset parameter for the EGFP protein and the GPRC5B protein respectively. The values obtained for the 3 treatments for each protein were within one standard deviation of each other. The results for the offset parameter proved to be the most stable along all the parameters, there

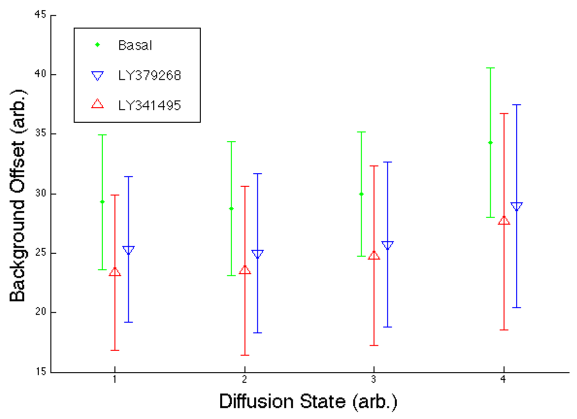

Figure 9: Background offset vs. diffusion state graph for the GPRC5B protein and its three treatments.

was no discrepancy between diffusion states, as described by Eq.7 since the offset parameter did not depend on the diffusion state. (i.e It did not depend on the number of particles in the middle of the ROI.). The stability of the offset parameter suggests that an improvement in the software could be done by running the analysis and obtaining a value for the offset parameter first, and then running it again with only the other two parameters to find.

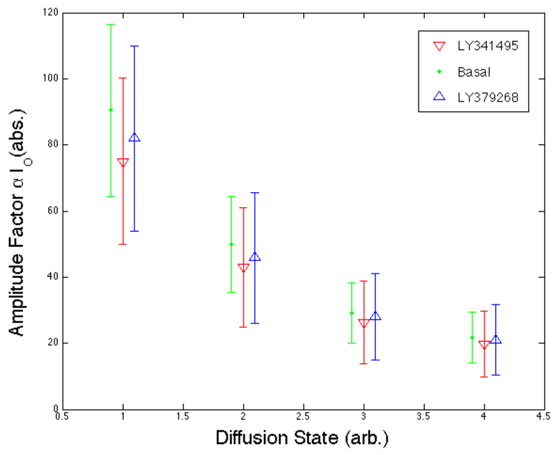

Figure 10: Amplitude factor vs. diffusion state graph for the EGFP protein and its three treatments.

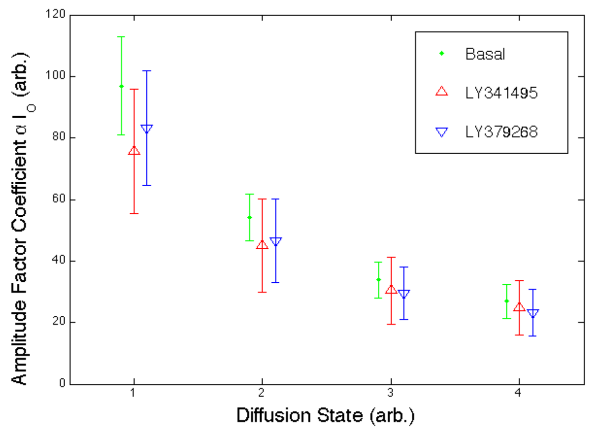

Figure 11: Amplitude factor vs. diffusion state graph for the GPRC5B protein and its three treatments.

	1st State	2nd State	3rd State	4th State
Basal	96(15)	54(7)	33(6)	26(5)
LY379268	83(18)	46(13)	29(8)	23(7)
LY341495	75(20)	45(15)	30(11)	24(9)

Table 2: Amplitude Factor Values in arbitrary intensity units obtained for each diffusion state for the GPRC5B protein.

	1st State	2nd State	3rd State	4th State
Basal	90(26)	49(15)	29(9)	21(7)
LY379268	86(28)	47(15)	27(9)	20(8)
LY341495	74(25)	42(18)	26(12)	19(10)

Table 3: Amplitude Factor Values in arbitrary intensity units obtained for each diffusion state for the EGFP protein.

Figure 10 and 11, and the two tables above show the results of the amplitude factor for the EGFP and the GPRC5B proteins. Two things become evident, the first one, which may seem obvious, is that the particles travelling slower have a larger number of particles in the cluster, (the exact relationship requires knowledge of the intensity given off by a single particle)[10]. The second is that the graph of the diffusion state vs. amplitude factor seems to flatten out as the particles get faster. Rather than representing the real relation between aggregation state vs. diffusion state, this seems to suggest that the values for the last two diffusion states are not precise based on the results obtained for the other two parameters, which as described previously could be due to tracking errors made by the particle tracking software.

Other effects such as the blinking of the fluorophores, which was assumed to be non-existent in the theory, could have affected the results although to a much smaller degree. The theory also assumed that the dynamics of the particles were only affected by the oligomerization of the protein receptors with themselves, this is not a real experimental condition as the protein receptors could be interacting with other particles around. To verify this a two color fluorescence microscopy analysis could be perform to observe the interactions with other molecules.[14]

The need for simulations containing binding rate between molecules and different diffusion states is necessary to improve the analysis, the precision, the accuracy of the results.

6 Conclusions

The analysis performed on the 2D free diffusion simulations showed that the averaged static and dynamics regions of interest resulted in a constant image and a PSF shape image respectively in accordance to the theory. Moreover the analysis returned values for the three parameters of interest for the 1 particle and the 100 particle simulations which were in agreement with the input values set within one standard deviation, cementing the theory behind DSpIDA.

The analysis performed on the data showed that the results for the offset parameter were the most stable for both experiments and each the diffusion state, which could be used in the future for improvements in the stability of the values of the other two parameters. Results on the PSF radius and the Amplitude Factor led to believe that in order to obtained the exact relationship between the number of particles and their respective diffusion states a better tracking

[10]It was assumed that it was the same as the focal point intensity of the microscope but this is not guaranteed and it therefore needs to be measured experimentally

technique will be needed for the faster diffusion states. Furthermore to complete the relation, experiments to determined the value of I_o given off by a single fluorophore are required.
The need for simulations involving binding rate and different diffusion states is necessary to make assumptions about the validity of the model under those circumstances which might lead to more precise and more accurate results in the future. As a last statement, it is hoped that the results obtained through this technique will be used in the future to make inference about properties of the protein receptors examined and ultimately biological systems.

Acknowledgments

I'd like to thank, first Professor Paul Wiseman for giving me the opportunity to work on this project, opening the door to this exciting field. Secondly, Professor Sako and his lab group at Osaka University in Japan for all the data provided.
Lastly, I'd like to thank Mr. Raphael Theberge for all his support and consideration explaining the work that had been done.

References

[1] Corti C et al. Altered dimerization of metabotropic glutamate receptor 3 in schizophrenia. *Biol. Psychiatry*, 62(7):747–755, October 2007.

[2] Axelrod D. Koppel D.E. Schlessinger et al. Mobility measurement by analysis of fluorescence photobleaching recovery kinetics. *Biophys. Journal*, 16:1055–1069, 1976.

[3] Ken Jacobson Michael J. Saxton. Single-particle tracking:applications to membrane dynamics. *Annual Review of Biophysics and Biomolecular Structure*, 76:373–399, June 1997.

[4] Gratton E Digman MA, Stakic M. Raster Image Correlation Spectroscopy and Number and Brightness Analysis. *Methods Enzymology*, 518:121–144, 2013.

[5] Yan Chen. Joachim D. Muller et al. The photon counting histogram in fluorescence fluctuation spectroscopy. *Biophysical Journal*, 77:553–567, 1999.

[6] Jean-Martin Beaulieu et al. Paul W. Wiseman. *Methods in Enzymology*, volume 522. Elsevier, 2013.

[7] P.W. Wiseman. Image correlation spectroscopy. *Comprehensive Biophysics*, 2:246 – 259, 2012.

[8] Scott C. Blanchard Qinsi Zheng. *Encyclopedia of Biophysics*. Springer, 2013.

[9] Petra Schwille Jonas Ries, Eugene P. Petrov. Total internal reflection fluorescence correlation spectroscopy: Effects of lateral diffusion and surface-generated fluorescence. *Biophysical Journal*, 73(1):390–399, July 2008.

[10] Yury A. Kutoyants. *Statistical Inference for Ergodic Diffusion Processes*. Springer, 2004.

[11] Hamsih Johnston. Ergodic theorem passes the test. *Physics World*, 2011.

[12] Nobuhiro Kurabayashi et al. The g protein-coupled receptor gprc5b contributes to neurogenesis in the developing mouse neocortex. *Development*, 140:4335–4346, 2013.

[13] Charlie Holly. *Grouped ZProjector*. ZProjector, 2004.

[14] F.J. Meyer-Almes P. Schwille. Dual-color fluorescence cross-correlation spectroscopy for multicomponent diffusional analysis in solution. *Biophysical Journal*, 72:1878–1886, 1997.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Appendix

MATLAB Code

Mean Rois:

```
% This script is used to calculate the histograms of the windows around the
% particles. The Data struct and imageSeries variables needs to already be in the
% environment.
% Raphael Theberge - Summer 2014
%Modified by David Herrera - Fall 2014

halfWinSize = 5;% now defined in SimulateAndAnalyze.m

side = 2*halfWinSize+1; % side length of the ROI
numpixels = side^2;
maxBin = double(max(imageSeries(:))); % maximum image intensity
minBin = double(min(imageSeries(:)));% minimum image intensity
bins = linspace(minBin,maxBin,100);
% intensity bin
numBins = length(bins);
skip = 0;
tau = 1;
NumberOfGSRuns = 10; % Number of GlobalSearch runs
bestguess = zeros(4); % 3 params + best least squares fit

for i=1:Data.NumberOfDStates %Initialize the final cdf matrix and numbers for each DState
    eval(['D' num2str(i) ' = 0;']);
    eval(['D' num2str(i) '_ROIS4Mean = [];']);
end

%Calculate the histograms
for i = 1:length(Data.Particle)

    %Cumulate histograms for all frames of the particle
    for j = 2:length(Data.Particle{i}.X)
        if Data.Particle{i}.DState(j) %Checks if non zero DState, can probably be removed
            xmin = round(Data.Particle{i}.X(j)) - halfWinSize;
            xmax = round(Data.Particle{i}.X(j)) + halfWinSize;
            ymin = round(Data.Particle{i}.Y(j)) - halfWinSize;
            ymax = round(Data.Particle{i}.Y(j)) + halfWinSize;
            if ymax > size(imageSeries,2) || ymin < 1 || xmax > size(imageSeries,1) || xmin
                < 1
                skip = skip + 1;
                continue % skip this frame, rectangle is out of bounds
            end
            [~,DPosition] = ismember(Data.Particle{i}.DState(j)...
            ,Data.AllDStates);
            DPosition = num2str(DPosition);
            eval(['D' DPosition ' = D' DPosition ' + 1;']); %Calculate total number of that
                DState we used
            eval(['condition = mod(D' DPosition ',tau);']);
            if ~condition;eval(['D' DPosition '_ROIS4Mean(:,:,end+1) =
                imageSeries(ymin:ymax,xmin:xmax,j);']);
            end
        end
    end
end
for j = 1:Data.NumberOfDStates %Check results for all DStates

        %Circular mean of all ROIS at once and put each ROIS as column
        %vectors now for the batch histogram count

        eval(['rois = D' num2str(j) '_ROIS4Mean;']); %Make a working matrix so code is
            readable
        eval(['Data.D' num2str(j) '_ROIS4Mean = D' num2str(j) '_ROIS4Mean;']);
        meanImg = (rois + flipdim(flipdim(rois,1),2) + flipdim(rois,1) +
            flipdim(rois,2))/4;
        rois = reshape(meanImg,side^2,size(rois,3));
        c = mean(rois,2);
        H = hist(c(:), bins);
        c = cumsum(H)/numpixels;
        ps = zeros(5,4);
        %% Fitting Params
        firstpt = 1;
        endpt = find(c >= 1.0000,1,'first');
        model = @(a) sum( (c(firstpt:endpt) - seek([a(1) a(2) a(3)], bins(firstpt:endpt))
            ).^2 );
        options = optimset('maxIter',10000,'MaxFunEvals',10000,'TolFun'...
        ,1e-30,'TolX',1e-30,'display','off','UseParallel','never');
        gs = GlobalSearch; gs.NumTrialPoints = 10000;

        guess = [4 bins(find(c > 0,1,'first')) .04];
        if find(c > 0,1,'first') > 1
            offsetlb = bins(find(c > 0,1,'first') - 1);
        else
            offsetlb = bins(1);
        end
        offsethb = bins(find(c-c(find(c > 0,1,'first')) > 0,1,'first'));
        lb = [1 0 0];
        ub = [side offsethb max(meanImg(:))-min(meanImg(:))];

                    cfmin = inf;
        %% Fitting
        for k = 1:5
            problem = createOptimProblem('fmincon','x0',guess,...
                'objective',model,'lb',lb,'ub',ub,'options',options);
            [xmin,fmin,~,~,~] = run(gs,problem);
            ps(k,:)=[xmin fmin];
            if fmin < cfmin
                Data.bestguess(j,:) = [xmin fmin];
                guess = xmin;
                cfmin = fmin;
            end
        end
end
```

BatchAnalysis2:

```
%This script obtains the file from the folders and categorizes the data
```

```
%into the Data Structure Struck used to describe the particles and
%and their properties
tic;
%Batch calculator
BatchCalculation = 1;
FoldersToAnalyze = {};
FolderNumber = 0;
AnalyzeTiffs = 0;
AnalyzeAvis = 1;

Ambiguities = {};
NumberOfAmbiguities = 0;
TxtNotFound = {};
NumberOfTxtNotFound = 0;


while 1
    FolderToAdd = uigetdir;
    if ~isstr(FolderToAdd)
        questdlg('No folder selected','','Ok','Ok');
    elseif ~sum(strcmp(FolderToAdd,FoldersToAnalyze))
        FolderNumber = FolderNumber + 1;
        FoldersToAnalyze{FolderNumber} = FolderToAdd;
        celldisp(FoldersToAnalyze)
    else
        questdlg('Folder was already part of the list','Duplicate name','Ok','Ok');
    end
    answer = questdlg('Do you want to add another folder in the batch ?');
    if strcmp(answer,'No')
        break;
    elseif strcmp(answer,'Cancel')
        msgbox('Analysis was cancelled');
        return;
    end
end

for folder = 1:FolderNumber
    fprintf('\n\n\n\n****Now processing folder : '); disp(FoldersToAnalyze{folder});
    cd(FoldersToAnalyze{folder})
        if AnalyzeAvis
            videotype = 'AVI';
        filesInFolder2 = ls('*.avi');
        filesInFolder=strsplit(filesInFolder2);
        filesInFolder=char(filesInFolder);
        filesInFolder(size(filesInFolder,1),:)=[];
        for file = 1:size(filesInFolder,1)
            if file>1
                    continue;
            end
            cd(FoldersToAnalyze{folder})
            [cutName, remain] = strtok(filesInFolder(file,:),'.');
            txtName = ls([cutName '*.txt']);
            txtName=strsplit(txtName);
            txtName=char(txtName);
            txtName(size(txtName,1),:)=[];
            if size(txtName,1) == 0
                NumberOfTxtNotFound = NumberOfTxtNotFound + 1;
                TxtNotFound{NumberOfTxtNotFound} = [FoldersToAnalyze{folder} filesep
                    filesInFolder(file,:)];
                continue;
            elseif size(txtName,1) > 1
                NumberOfAmbiguities = NumberOfAmbiguities + 1;
                Ambiguities{NumberOfAmbiguities} = [FoldersToAnalyze{folder} filesep
                    filesInFolder(file,:)];
                continue;
            end
            fprintf([filesInFolder(file,:) '\n was associated to \n' txtName '\n'])

            %Loading imageSeries from .avi
            disp('Loading .avi movie into imageSeries')
            disp('Beginning')
            vid = VideoReader([FoldersToAnalyze{folder} filesep filesInFolder(file,:)]);
            imageSeries = zeros(vid.height,vid.width,vid.NumberOfFrames);
            for i = 1:vid.NumberOfFrames
                imageSeries(:,:,i) = flipud(read(vid,i));
                if ~mod(i,500)
                    disp(['Frame ' num2str(i) ' of ' num2str(vid.NumberOfFrames) ' loaded'])
                end
            end
            clear vid
            disp('Done loading the video')

            %Building the Data structure
            disp('Loading the data found in the .txt into the Data struct')
            DataStructureGenerator;
            disp('Done loading the .txt file into the Data struct')

            %Calculating the cumulative distributions of the particles
            disp('Calculating the cumulative distributions of the particles')
            MeanRois;
            %            CalculateCDFs;
            disp('Done computing the cdfs!')
            toc;
            cd('/Users/davidherrera/Documents/phys 449')
            save([Data.Name date() 'avi'], 'Data','-v7.3')
            end
        end
        if AnalyzeTiffs
            videotype = 'TIFF';
            filesInFolder = ls('*.tif');
            filesInFolder=strsplit(filesInFolder);
            filesInFolder=char(filesInFolder);
            filesInFolder(size(filesInFolder,1),:)=[];
        for file = 1:size(filesInFolder,1)
            cd(FoldersToAnalyze{folder})
            cutName = strtok(filesInFolder(file,:),'.');
            txtName = ls([cutName '*.txt']);
            txtName=strsplit(txtName);
```

```
            txtName=char(txtName);
            txtName(size(txtName,1),:)=[];
            if size(txtName,1) == 0
                NumberOfTxtNotFound = NumberOfTxtNotFound + 1;
                TxtNotFound{NumberOfTxtNotFound} = [FoldersToAnalyze{folder} filesep
                    filesInFolder(file,:)];
                continue;
            elseif size(txtName,1) > 1
                NumberOfAmbiguities = NumberOfAmbiguities + 1;
                Ambiguities{NumberOfAmbiguities} = [FoldersToAnalyze{folder} filesep
                    filesInFolder(file,:)];
                continue;
            end
            fprintf([filesInFolder(file,:) '\n was associated to \n' txtName '\n'])

            %Loading imageSeries from .tif
            disp('Loading .tif movie into imageSeries')
            disp('Beginning')
            FileTif = [FoldersToAnalyze{folder} filesep filesInFolder(file,:)];
            InfoImage = imfinfo(FileTif);
            mImage = InfoImage(1).Width;
            nImage = InfoImage(1).Height;
            NumberImages = length(InfoImage);
            imageSeries = zeros(nImage,mImage,NumberImages,'uint16');

            for i=1:NumberImages
                imageSeries(:,:,i) = flipud(imread(FileTif,i));
                if ~mod(i,500)
                    disp(['Frame ' num2str(i) ' of ' num2str(NumberImages) ' loaded'])
                end
            end
            clear nImage mImage NumberImages InfoImage
            disp('Done loading the video')

            %Building the Data structure
            disp('Loading the data found in the .txt into the Data struct')
            DataStructureGenerator;
            disp('Done loading the .txt file into the Data struct')

            %Calculating the cumulative distributions of the particles
            disp('Calculating the cumulative distributions of the particles')
            MeanRois;
            %            CalculateCDFs;
            disp('Done computing the cdfs!')
            cd('/Users/davidherrera/Documents/phys 449')
            save([Data.Name 'tiff'], 'Data','-v7.3')
            toc;
        end
     end
    disp(['*****Folder ' num2str(folder) ' of ' num2str(FolderNumber) ' done'])
end

if ~isempty(Ambiguities)
    fprintf('\n\n***There were some ambiguities resulting in some data not processed')
    celldisp(Ambiguities);
end

if ~isempty(TxtNotFound)
    fprintf('\n\n***There were some videos we could not link to data')
    celldisp(TxtNotFound);
end

toc;
```

Seek:

```
function [ cdfvector ] = seek( param, bins )
%Function used to fit the data
side = 11;
half = floor(side/2);

ubounds = 0.5;
lbounds = -0.5;
x = -half:half;

%Int_Pix =pi/8 * param(1) * param(1) *
    (erf(sqrt(2)*(ubounds-x)/param(1))-erf(sqrt(2)*(lbounds-x)/param(1)));
Int_Pix =sqrt(pi/8)* param(1) *
    (erf(sqrt(2)*(ubounds-x)/param(1))-erf(sqrt(2)*(lbounds-x)/param(1)));

Int_Pix = param(3).*Int_Pix'*Int_Pix;
Offset = ones(size(Int_Pix)).*param(2);
P = Int_Pix +Offset;

H2 = hist(P(:), bins)';
H2(H2==0) = eps(sum(H2));
cdfvector = cumsum(H2) / side^2;

end
```

DataStructureGenerator:

```
% This script is used to build the data structures from the raw data txt
% files from the single particle tracking software.
% Raphael Theberge - Summer 2014


if ~exist('BatchCalculation','var') || ~BatchCalculation
    [filename,filepath,~] = uigetfile('*.txt');
    cd(filepath)
else
    filepath = FoldersToAnalyze{folder};
    filename = txtName;
end

% Open the raw data file
fid = fopen([filepath filesep filename]);

tline = fgetl(fid); %Reading first line
```

```
Data.NumberOfDStates = 0; %Initializing the field
Data.AllDStates = [];
Data.Name = strtok(tline,'.'); % Storing the name of the data we are working with
while isempty(strfind(tline,'frame')) || isempty(strfind(tline,'X')) ||
    isempty(strfind(tline,'Y'))
    %   disp(tline) % Display starting lines, if you feel like it
    tline = fgetl(fid);
end

% Getting rid of the "FRAME X Y" line so we are ready to build our structures
tline = fgetl(fid);

% Initializing the values for the loops
ParticleNumber = 0;

% Building the particle structures from the following data lines.
while ischar(tline)
    [Frame,Remain] = strtok(tline); % Getting the frame number
    if str2double(Frame) == 0 %Checking if new particle
        ParticleNumber = ParticleNumber + 1;
        Data.Particle{ParticleNumber}.X = [];
        Data.Particle{ParticleNumber}.Y = [];
        Data.Particle{ParticleNumber}.DState = [];
        i = 1;
        if ~mod(ParticleNumber,200)
            disp(['Particle ' num2str(ParticleNumber-1) ' was just loaded']);
        end
    else
        i = i + 1;
    end
    [temp, Remain] = strtok(Remain); % Getting the X value
    Data.Particle{ParticleNumber}.X(i) = str2double(temp); % Storing the X value
    [temp, Remain] = strtok(Remain); % Getting the Y value
    Data.Particle{ParticleNumber}.Y(i) = str2double(temp); % Storing the Y value
    [~, Remain] = strtok(Remain); % Getting rid of Intensity
    [~, Remain] = strtok(Remain); % Getting rid of Data
    [temp, ~] = strtok(Remain); % Getting the State Value, getting rid of the Remain
    Data.Particle{ParticleNumber}.DState(i) = str2double(temp); % Storing the State Value
    if ~ismember(str2double(temp),Data.AllDStates) && str2double(temp)%Check if new DState
        Type, if so update the list
        Data.AllDStates(end+1) = str2double(temp);
        Data.NumberOfDStates = Data.NumberOfDStates + 1;
    end
    tline = fgetl(fid); % Getting next line
end
Data.AllDStates = sort(Data.AllDStates);
fclose(fid); % Closing the text file
disp([num2str(ParticleNumber) ' particles were detected and stored in the Data variable'])
clear Remain temp Frame tline fid ParticleNumber i ans
```

SimulateAndAnalyze

```
% clc; clear all; close all;
simulator; % Simulate the experiment
onParticle = 1; % Set the ROI on or off particle
```

```
halfWinSize = 5; % Self explanatory

if (onParticle == 1)
    for i=1:size(particles.pos,1) %Go through all particles
        if i == 1 %For the first particle, note the coordinates
            Data.Particle{i}.X = squeeze(particles.pos(i,1,:));
            Data.Particle{i}.Y = squeeze(particles.pos(i,2,:));
            Data.Particle{i}.DState = ones(length(particles.pos(i,2,:)),1);
            Data.AllDStates = [1];
            Data.NumberOfDStates = 1;
        end
    end
else %If off particle, calculate mean image
    Data.Particle{1}.X = ones(size(particles.pos(1,1,:),3),1) *
        size(particles.pos(1,1,:),1)*size(imageSeries,1)/2;
    Data.Particle{1}.Y = ones(size(particles.pos(1,1,:),3),1) *
        size(particles.pos(1,1,:),1)*size(imageSeries,2)/2;
    Data.Particle{1}.DState = ones(size(particles.pos(1,2,:),3),1);
    Data.AllDStates = [1];
    Data.NumberOfDStates = 1;
end
Data.Name = 'hi'; %because the code requires a name somewhere
MeanRois; %Go calculate the cdf's
```

Simulator:

```
% created by H.B. on 03/01/2014
% modified on 23/03/2014 diffusion direction error fixed
% program simulates free diffusion in 2D with binding out of plane
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Variable parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

N = 100; % particle number
T = 1000; % detector time samples
sizeX = 64; % pixels
sizeY = 64; % pixels
PSFsize = 4; % pixels in e^-2 radius
pad = 0; % (yes)no (i.e. (non)periodic boundary condition)
nT= 1; % (number of simulation steps per detector time sample)

D = 1; % pixels^2/frame

includePhotophysics = 0; % do simulations with blinking/binding
ka = 1/7; % association rate
kd = 1/7; % dissociation rate
Pon = ka/(ka+kd); % probability that particle is bound at t=0


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Particles structure: initialize
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
particles = struct('pos',zeros(N,2,T*nT),'state', zeros(N,T*nT));
% initialize field: .pos (x,y) positions
particles.pos(:,:,1) = rand(N,2,1).*repmat([sizeX,sizeY],N,1);
```

```
% initialize field: .state (on/off)
particles.state(:,1) = 1.0*(rand(N,1)<=Pon);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Particles positions: update
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for f=2:T*nT
  % generate new positions
  particles.pos(:,:,f) = particles.pos(:,:,f-1) + sqrt(2*D/nT).*randn(N,2);
  % apply periodic boundary
  particles.pos(:,1,f) = mod(particles.pos(:,1,f),sizeX);
  particles.pos(:,2,f) = mod(particles.pos(:,2,f),sizeY);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Particles state: update
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if(includePhotophysics == 1)
    for n=1:N
      particles.state(n,:) = gillespie(particles.state(n,1),ka,kd,nT,T);
    end
else
    particles.state = ones(N,nT*T);
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Save particles simulation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Create video images
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

imageSeries = zeros(sizeY,sizeX,T);

kernelSize = ceil(6*PSFsize); % makes Gaussian kernel with radius 3*PSFsize
% kernelSize = round((kernelSize-1)/2); %
[x,y] = meshgrid(-kernelSize:kernelSize,-kernelSize:kernelSize);
factor = 1/(2*pi*PSFsize);

thisFrame = 0;
for f=1:T*nT
  if(mod(f-1,nT)==0)
    thisFrame = thisFrame + 1;
    %disp([f thisFrame]);
  end
  for n=1:N
    dx = -round(particles.pos(n,1,f))+particles.pos(n,1,f); % dx shift from pixel center
    dy = -round(particles.pos(n,2,f))+particles.pos(n,2,f); % dy shift from pixel center
    arg = -((x-dx).*(x-dx) + (y-dy).*(y-dy))/(PSFsize^2/2);

    % create PSF Gaussian kernel
    kernel = exp(arg).*factor ;
    kernel(kernel<eps*max(kernel(:))) = 0;
%     sumk = sum(kernel(:));
%     if sumk ~= 0,
%       kernel = kernel/sumk;
%     end
    nonZeroEl = find(kernel);
    % find kernel index on image
    xcoor = mod(x(nonZeroEl) + round(particles.pos(n,1,f)),sizeX);
    ycoor = mod(y(nonZeroEl) + round(particles.pos(n,2,f)),sizeY);
    % fix MATLAB index of 1
    xcoor(xcoor==0) = sizeX;
    ycoor(ycoor==0) = sizeY;
    % make image
    imgKernel = full(sparse(ycoor,xcoor,kernel(nonZeroEl),sizeY,sizeX));
    % add image to video
    imageSeries(:,:,thisFrame) = imageSeries(:,:,thisFrame) +
        particles.state(n,f).*imgKernel;
  end
end

%Line to add constant offset
imageSeries = imageSeries +1;

%%Line to add noise

imageSeries = imageSeries + randn(size(imageSeries))*.1*max(imageSeries(:));
```

meanAnalysisTrials

```
%Calculates the error in the parameters, based on the mean between
%bestguess contains the lowest least square obtained and the parameteres
%experiments, chlaritin

 filesInFolder2 = ls('*.mat');
 filesInFolder=strsplit(filesInFolder2);
 filesInFolder=char(filesInFolder);
 filesInFolder(size(filesInFolder,1),:)=[];
 for file = 1:19
    load(filesInFolder(file,:))
    %file is the trial, each column represents a diffusion state
    psfRadius(file,:)=transpose(Data.bestguess(1:4,1));
    offset(file,:)=transpose(Data.bestguess(1:4,2));
    amplitudeFactor(file,:)=transpose(Data.bestguess(1:4,3));
    chiSquare(file,:)=transpose(Data.bestguess(1:4,4));


    %Choose the diffusion stat
 end
 meanPSFLY379268=mean(psfRadius)
 stdPSFLY379268=std(psfRadius)
 meanAFLY379268=mean(amplitudeFactor)
 stdAFLY379268=std(amplitudeFactor)
 meanOFFLY379268=mean(offset)
 stdOFFLY379268=std(offset)
 meanCHI=mean(chiSquare);
 stdCHI=std(chiSquare);
```

SimulatorResults:

```
%% Does the simulation for 1 particle in 2D difussion and analyzes it using dSpida
%The parameters for the psfsize, the amplitude factor, and the offset are specified
%The analysis will validate the accuracy of dSpida

for run=1:10
    SimulateAndAnalyze;
    total(run)=guess;
    clear;
end
    psfMean=mean(total(:,1));
    amplitudeMean=mean(total(:,2));
    offsetMean=mean(total(:,3));
    std=std(total);
```

Graph Code Results

fittingCDF:

```
%CDF + Fit Function to find CDF of data and model
firstpt = 1;
endpt = find(c >= 1.0000,1,'first');
psa=psTot(10,:);
figure(1);hold on;
scatter(bins(firstpt:endpt),c(firstpt:endpt),'blue+','LineWidth',3)
model1=seek([mean(psa(:,1)) mean(psa(:,2)) mean(psa(:,3))],bins(firstpt:endpt));
model=seek([4.0 1.0 0.0398],bins(firstpt:endpt));
plot(bins(firstpt:endpt), model1,'.red','LineWidth',2)
%plot(bins(firstpt:endpt), model,'.green','LineWidth',2)
legend({'Data (100 Particle)','Fit (100 Particle)'},0)
xlabel('Pixel Intensity (arb.)','FontSize',16)
ylabel('Cumulative Probability','FontSize',16)
ylim([0 1.05])
bins(51)
bins(52)
for i=1:endpt-1
   plot([bins(i) bins(i+1)],[c(i+1) c(i+1)]);
   plot([bins(i) bins(i+1)],[model1(i+1) model1(i+1)], 'r');
   %plot([bins(i) bins(i+1)],[model(i+1) model(i+1)], 'g');
end
xlim([0.98 1.08]);
hold off
```

RunsMeanRois:

```
%%Mean Rois Simulation to obtained various trials
for counter=1:10
    MeanRois;
    psTot(counter,:)=min(ps);
end
```

GraphMeanDiffState:

```
%Graph for the different parameters vs diffusion state
errorbar(meanAFLY341495, stdAFLY341495, '.','Color', 'r')
hold on
errorbar(meanAFBasal, stdAFBasal, '.','Color', 'g')
hold on
errorbar(meanAFLY379268, stdAFLY379268, '.','Color', 'b')
```

ImagePlusRoi:

```
%Image of Simulations, and ROI of image
frame=400;
xorigen= round(Data.Particle{1}.Y(frame)) - halfWinSize;
yorigen = round(Data.Particle{1}.Y(frame)) - halfWinSize;
figure;
xlabel('Pixels','FontSize',16)
ylabel('Pixels','FontSize',16)
imagesc(imageSeries(:,:,frame));
colorbar;
rectangle('Position',[xorigen yorigen side side],'LineWidth',2)
```

staticROI:

```
%Static ROI image obtained.
figure;
imagesc(mean(meanImg,3))
xlabel('Pixels','FontSize',16);
ylabel('Pixels','FontSize',16);
colorbar;
caxis(handle);
```

A1Q1 a,b:

```
%Function Calculates mutations under the Darwin model, simulating
%the Luria and Dellbruck experiment
function [times, r_T]=assignment1Phys519
a=5E-8;
N_o=10;
max_time=7*log(10);
dt_max=log(2);
lambda=0.1;
t_r=log(lambda/(a*N_o*dt_max));
i=1;
for j=1:1E6
  steps=1;
  r=0; % Number of resistant bacteria.
  time=0;
while( time < max_time)

    mutation = 0;  % No mutation occurred in dt.
     if time<t_r
         dt=log(2);
     else
         dt=lambda/(a*N_o*exp(time));
     end
    if ( a*N_o*exp(time)*dt > unifrnd(0,1) )
        mutation = 1;        % One mutation occured in dt.
    end

    r =r + dt*r + mutation;
     if r ==1
        times(i)=time;
        i=i+1;
     end
    % Growth and mutations.
    time =time + dt;
    steps=steps+1;
end
   if r > 0
    r_T(i-1)=r;
   end
end
[counts, centers]=hist(r_T, 850000);
counts=counts/993540;
plot(centers, counts, '.')
%end
```

A1Q2c:

```
function ass1Q2Phys519
%This function finds the intersection between the curves of degradation
%and production for a self-activating gene for different values
%of sigma.
xintVector=[];
simgaVector
x=[0:1000];
p=66.11;
sigmaInc=[6:-0.1:0];
sigma=4;
z1=p*(x.^5)./(x.^5+10^5)
z2=sigma*x;
%Intersection function finds the intersections between two curves
[xint yint]=intersections(x,z1,x,z2)
s=size(xint,1);
xTemp=xint;
xintVector=vertcat(xintVector,xTemp);
sigTemp=sigma*ones(s,1);
sigmaVector=vertcat(sigmaVector,sigTemp);
%Plotting of the intersections as a function of sigma
plot(x,z1)
xlim([-1 30])
ylim([-2 75])
hold on
%Plot of curves for production and degradation
% plot(x,z2)
% hold on;
% plot(x,z1)
```

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.