Bayesian Clustering for Single Molecule Localisation Microscopy

Sean Baccas
Supervised by Professor Paul French and Dr Andrew Rose

Abstract

In recent years, the advent of single-molecule localisation microscopy (SMLM) [17] has enabled imaging of biological systems beyond the limit of the camera resolutions. In this review, we aim to describe SMLM, and motivate the need for identifying clusters within the data.

1 First Plan

First thing is to outline the plan for this write up.

- introduce the maths tool we need
- explain how the data are generated
  - discuss SMLM
  - discuss storm
  - discuss thunderstorm
- talk about methods for identifying clusters such as DBScan
- Talk about the complexity behind our algorithm - the R and T scans in particular
- Discuss how the code was implemented in depth
- Analyse runtime and memory usage
- What about running some artificially clustered data through it? from eg picasso
- discuss threshold in more depth
- discuss sigma curve in more depth
- mention that Airy functions can be approximated with a gaussian

2 Introduction

3 Fluorescence Microscopy and SMLM

todo:

- mention PALM
- in the storm paper they mention that with enough photons, you can determine the position of a single emitter with arbitrarily high precision, but this does not translate into image resolution.

Fluorescence microscopy is a broad field of study that handles imaging of biological **samples** on a sub-micrometre scale. For a more thorough handling of the topic, please refer to [4, 9]. Whilst the function of modern microscopes is relatively similar to previous eras, what has changed is the ability to examine fluorescence coming directly from molecules of interest. This allows for the study of the spatial and temporal dynamics of the processes that occur within cells. One key **development** has been the use of different *fluorophores*, fluorescent proteins which can be attached to molecules of interest. The wavelengths of light they emit can provide key indicators of the dynamics. In particular, proteins such as Alexa-647[1] and CF5[2]

[1] cite
[2] cite

can be efficiently attached to cellular molecules of interest, producing emissions **as and when necessary**.

Fluorescence microscopy is limited in its resolution. Due to diffraction that occurs as light reaches the **focal plane**, the smallest resolvable objects are around 200nm. Below this, fluorophores appear as a blurry point spread function (PSF)[3]. The pattern observed is an Airy disk [9], with a full width half maximum of approximately $\lambda/2 \cdot NA$, where $0.2 < NA < 1.5$ is the numerical aperture of the microscope.

we don't want to have to use electron microscopes!!

Having discussed fluorescence microscopy **in depth**, we can now turn our attention to the favoured techniques of this project. Stochastic Optical Reconstruction Microscopy (STORM) [27], which would go on to win the 2014 Nobel prize in Chemistry. As the authors mention, individual molecule positions can be determined to high accuracy with sufficiently many photons, but this does not directly translate into higher-resolution imaging techniques as multiple fluorophores nearby each other will be difficult to resolve **cite**. The STORM method evades this by ensuring that fluorescent molecules are stochastically switched on and off, with the eventual image being aggregated from many different frames, with the aim being that only one fluorophore is blinking per frame in each diffraction-limited of the field of view. Each molecule is labelled with a photoswitchable dye, meaning that it can be stimulated into emitting a photon when illuminated with laser of a particular wavelength. Rust et. al. use Cy5, a cynanine dye which they demonstrate can be encouraged to fluoresce, or sent back to a dark state, in a controlled and reversible manner[4] [27, 2].

3.0.1 Photoswitching and Dyes

todo:

- not sure if **stimulated** is the right word
- review the paper in the comment under the fig
- this part should be about dye selection and how to label - might not be that detailed.

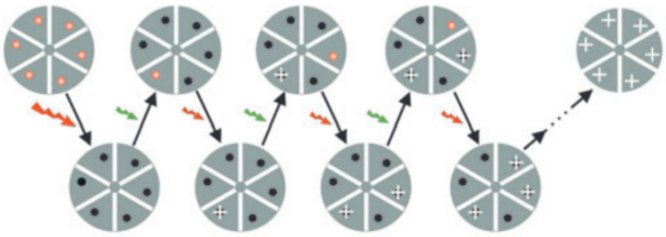


Figure 1: STORM imaging sequence, taken from [27]. For demonstration, Rust et. al. present a hypothetical hexameric object containing red fluorophores. In the first instance, a powerful red laser is used to turn all of the fluorophores into the dark state. In the second, a green laser pulse is used to activate some fraction of the fluorophores, sending them to the emitting state. Illumination from a red laser then stimulates emission from these fluorophores until they are switched off. The white crosses denote molecule positions which have been determined to high accuracy. The overall image is constructed from the amalgamation of many different imaging cycles.

Xu et. al. [33] give a good review of the considerations that must be made when attempting to label photo-switchable fluorophores onto the molecular target of interest. They state that the aforementioned

[3] include one!
[4] too close wording

Cy5 or Alex Fluor 647[5] are the best available fluorophores for *direct STORM*[6], due to their high photon counts, desirable blinking properties[7] and high photon counts. **State also that these dyes can go through many cycles before bleaching occurs.**

stuff to note:

- fiducial markers are used to correct sample drift.
- why don't they all get lit up at once?

3.0.2 Image reconstruction

todo - see paper in comment
Below follows some notes on the "image reconstruction" chapter of [33].

The raw data consist(s) of roughly 10,000 frames are captured over an **state some FOV here** using a **pulsed laser** to trigger photoswitching[8]. Here, we discuss ThunderSTORM [19], and how it reconstructs super-resolved images from these frames.

In brief, it's a software tool that allows for the reconstruction of super-resolved images. It can be used to set a minimum signal-to-noise threshold, so that imprecise localisations are removed[9].

3.1 The Issues Encountered When Attempting to Process Point Cloud Data, or why do we care about clusters

todo:

- attempt to quantify the difference in scales we're looking at here
- discuss the effect of imaging artefacts - note that our method includes the standard deviation from the read-in section as well

stuff to mention

- the output is a point cloud of x and y coordinates
- we want to evaluate the cluster proposals
- [21] talks about why we care about clustering

Having discussed SMLM, we turn our attention now to the problem of quantifying the data that we receive from SMLM processes. Much of the following chapter will follow [17], which is a comprehensive review of the cluster analysis and **quantification** methods. As mentioned by them, the majority of methods[10] are based on second-order statistics, which are sensitive to noise and imaging artefacts. This review also discusses localisations in three dimensions.

Broadly speaking, the analysis of the **size, location** and other characteristics of clusters allows for the extraction of a signature of the underlying biological processes. Indeed, as Pageon et. al. [22] put it, SMLM techniques retain spatial information of each molecule, allowing for granular access to local parameters to establish hierarchical information at various scales, be it at the level of the entire cell, at the level of protein clusters, or at the level of molecules[11]. Much of the methods in this **space** relate to

[5] **cite**
[6] When only a single dye is used, see van de Linde et al., 2011 from this paper
[7] Which are?
[8] see "data acquisition" section of this paper
[9] later, discuss how we can improve their read-in with GPUs.
[10] direct quote here
[11] direct quote please change

protein-protein interactions, and understanding the structure and organisation of their clusters is crucial to understand how they function within the cell [15]. Figure 6 (from [15]) motivates the need to quantify the clusters within biological samples, as this understanding is important to differentiate between the various **paths** that a system may take from one state to the next.

The review from Owen et. al. [20] details some of the important insights that it was possible to garner with this order-of-magnitude improvement in imaging resolution. When discussing **lipid rafts**, the most accurate description that could be given about them was one that **referred** to their small and dynamic nature. Whilst their importance in cellular processes was appreciated, the methodologies used to study them were inherently limited by diffraction-limited microscopy [20]. The advent of SMLM methods, such as PALM **cite** and STORM **cite** then allowed this definition to be updated to reflect that small perturbations can easily tip the balance between cell systems settling in different configurations [20], c.f. [18][12]. Moreover, these new super-resolved imaging methods allow for much deeper insights into the ultimate function of **rafts**, regulating the distribution and diffusion of membrane associated proteins, as well as regulating protein interactions for efficient signalling and trafficking [20]. It is with processes like this in mind that we interest ourselves in the clustering of molecules within biological systems.

The problem now is to efficiently analyse the pointillist data that comes out of static-cell imaging processes like STORM. This representation is fundamentally different to those used in conventional microscopy, namely the intensity grid valued pixel and voxel image representations [15]. These representations are challenging to perform inference upon, largely due to the quantisation errors that may be introduced during the voxelisation, but more broadly analysing them can be challenging to due the sparse nature of their representation [35] **cite this [1] too**. The granular nature of point cloud data can provide much deeper insights, but needs newer, bespoke analysis tools. Identifying and characterising the clusters of fluorophores are the most suitable way to gain insight into SMLM data [15].

The field of cluster analysis is a well-researched topic in computer science is detailed and well-studied. **Some important review papers can be seen here** [14]. The main issue that we encounter when trying to cluster data is the combinatoric complexity, as well as the issue of the assumptions that must be made when deciding what constitutes membership of a cluster. Ideally, the method used should rely on as few priors and user-defined parameters as possible[13]. The data in this context are unlabelled, meaning that the clusters must be elucidated purely from the spatial distribution of the points within them.

3.1.1 A little more specific

Of particular interest of this project is the imaging of nucleosomes and related protein complexes. DNA consists of billions of individual base pairs that express genes, which would stretch to over a metre if laid out in full. When in the nucleus, these base pairs are wrapped around approximately 30 million nucleosomes, forming a complex called chromatin [31, 8]. The compacted DNA can be reduced in length by six orders of magnitude, so that it can fit within the nucleus of a cell. Whilst the DNA sequence is fixed, the structure of chromatin is constantly in flux due to a wide range of processes [31]. Modern advances in microscopy techniques have been very fruitful for uncovering the dynamics of higher-order chromatin structure in a variety of biological processes [31]. In particular, Xu et al [34] went on to demonstrate various direct clinical uses for imaging chromatin structure, as they showed its shape is disrupted as cells undergo carcinogenesis. Furthermore, Ricci et al [24] showed that nucleosomes group together in groups of various sizes, and that the density of their groups was specific to each cell-type. In other words, the *clusters* of various biological cells are important in understanding what occurs at the cellular level, and the advent of SMLM has played a key role in enabling our understanding of such processes.

[12] two direct quotes which i don't understand - important thing to grasp is that before it was just looking all the same
[13] refer to here when talking about DBScan etc

4 What algorithms are there for performing cluster analysis?

- do this outside of a purely biological context
- NEXT - talk about all the other, lesser, clustering algorithms
- why is eg k means not suitable in this sense?

As mentioned, the topic of identifying clusters is a well-studied one in computer science. For this section, we will discuss the problem at hand, as well as many of the most popular methods for identifying and quantifying SMLM clusters, following on from [15].

The output data from ThunderSTORM is a large list of *localisations* with associated metadata. Each localisation consists of x and y coordinates of the fluorophore within the FOV, as well as the frame number in which this photon was detected, the number of photons detected in this position, and the uncertainty in the position of each of the coordinates, which will become important when we come to discussing the Bayesian algorithm in section 5. Also included is the width of the point spread function that produced the given photon, which we can use to remove erroneous data from the localisation table. We expect the PSFs produced by Alexa 647 in STORM imaging to be between 160 and 200nm[14], meaning entries outside of this range can be chalked up to artefacts of the imaging or localisation process. In some instances we will observe phantom clusters produced by particular fluorophores blinking in multiple frames - we can use the **photon count** measure to correct for this.

Identifying the clusters in a (typically very large) dataset is a problem which seeks to determine groups of molecules which appear to be more densely packed than the image in general. As we shall discuss in depth when outlining the Bayesian method for identifying clusters, this can come down to selecting a spatial scale at which the localisations can be considered *clustered*. Getis and Franklin [10] summarise the relationship of localisation i with all the other $j \neq i$ as being represented by the distance between i and all of its neighbours and the spatial scale at which the local density of localisations appears different from the global density. This method identifies clusters by determining a scale at which we consider clusters to be **statistically significant**. We begin by discussing statistical methods of determining clusters.

4.1 Statistical Methods

Ripley's K function [25], as well as similar measures named after him, are popular methods for identifying clusters. Prior to this work, spatial patterns were usually identified by testing the likelihood that a particular pattern could have simply been generated by a random process, such as the Poisson process. Ripley's method was built on the assumption that a full map of the spatial pattern was available (as is more or less the case when dealing with SMLM data), searching for *second-order* methods to test whether a pattern is different from random noise. We call this a second-order method since it concerns second moment properties: the first moment property is simply the number of localisations in a given area, whereas the second moment relates to the expected number of localisations within a fixed distance of another point [16, 15]. We define Ripley's K function here, since it is important background for how we determine clusters at a later stage. Formally it is defined as [25, 16]:

$$K(r) = \frac{1}{n} \sum_{i=1}^{n} N_{p_i}(r)/\lambda,$$

where the sum is taken over all n points in the dataset, and $N_{p_i}(r)$ is the number of points within a distance r of p_i. We have also that our score is normalised by the number of points per area λ. The expected value of $K(r)$ for a random distribution, generated by the Poisson process, is πr^2 [16]. The reason

[14] cite!

for this is that a uniformly random distribution of points will have $N_{p_i}(r) = N_{p_j}(r) = n$, meaning that, replacing $\lambda = \frac{n}{\pi r^2}$, summing from 1 to n will yield $\frac{1}{n} \cdot n^2 \cdot \frac{\pi r^2}{n} = \pi r^2$. We can then use this to measure clustering: $K(r) > \pi r^2$ indicate regions of higher density than a uniform background.

4.1.1 One Method In Particular

It was this idea of analysing the densities around each point that led to the formation of the analysis technique which we focus on in this project. The method from Getis and Franklin [10] was developed following their technique to describe the tree spatial patterns at a number of scales[15]. Developed following Ripley [25], their analysis is **designed** to test randomness hypotheses, by examining the proportion of pairs whose members are within a certain distance of each other[16]. They state that these methods are similar to second order analysis, but rather than producing a measure for the whole dataset, instead focusing on issuing a score for each point i individually. They define the score for each point as follows [10]:

$$L(r)_i = \sqrt{A \sum_{j=1}^{N} \frac{\delta_{ij}}{\pi(N-1)}},$$

where:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i \neq j \text{ and } |i-j| \leq r \\ 0 & \text{otherwise} \end{cases}$$

Here A is simply the area of the ROI. They include also an edge-correction for when the distance $r > |i-j| > e_1$, with e_1 being the distance to the nearest boundary. In this case, δ_{ij} takes the value:

$$\frac{1}{1 - \arccos(e_1/r)\frac{1}{\pi}}, \tag{1}$$

which is at least 1 and is maximised the closer point i is to the edge of the ROI. If i is closer to *both* boundaries than to point j, then δ_{ij} takes the value:

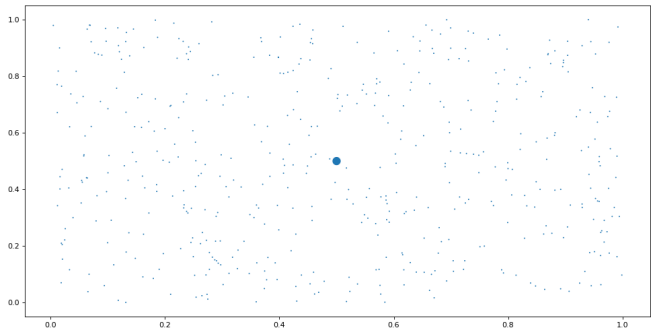
$$\frac{1}{1 - \frac{1}{2\pi}(\arccos(e_1/r) + \arccos(e_2/r) + \pi/2)},$$

where e_2 is the distance from the point i to the other boundary. This allows us to still compute the score for a point, where additional contributions may be obscured by the boundary of the ROI.

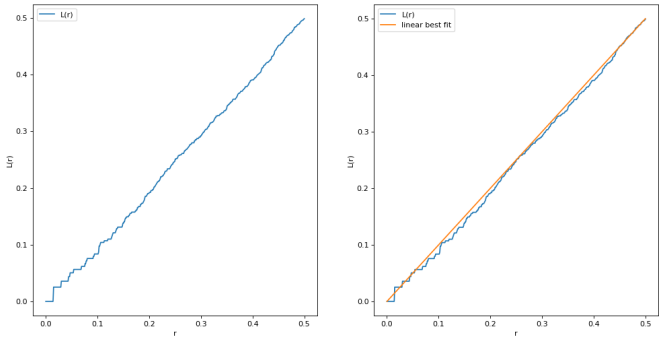
We have here, inspired by Getis and Franklin [10], reproduced some diagrams that illustrate the usefulness of their measure. In figure 2(a), we have highlighted a central point, surrounded by a sea of randomly scattered others. Figure 2(b) demonstrates that, as we increase r, gradually more and more points come within a distance r of our central point, leading to $L(r)$ increasing. Due to the points being scattered randomly, we expect the sum total of the number of points within a distance r to be proportional to the area of our scan, πr^2. The π and the square root were explicitly chosen so that $L(r)$ would be *linear* in r, as we can see when compared with a straight line of best fit.

Figure 3 shows a perhaps unwieldy, but still useful complementary example. Our highlighted central point is now isolated from a dense region of other points by about 0.15 units. This is reflected in the graph of $L(r)$ against r: it remains at 0 until the first member of the cluster comes within range, before rapidly picking up. This steep gradient here, in their words, implies a tendency for clustering [10]. At just under 0.2 units, $L(r)$ passes above the linear line, meaning that we now have denser clustering than a uniform,

[15] get a nice diagram in here
[16] direct quote please fix



(a) **caption goes here**



(b) **caption goes here**.

Figure 2

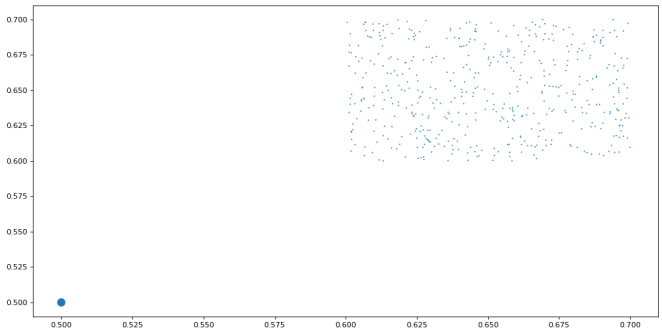
spatially random background[17]. We see further that the curve peaks and levels off at just under 0.3 units, showing us that the clustering is saturated. The authors give also some reasonable numbers for the y - intercept on such a straight line of best fit, that would represent the 5% and 1% line of best fit, namely $\pm 1.42\sqrt{(A)}/(n-1)$ and $\pm 1.68\sqrt{(A)}/(n-1)$. Herein lies the issue that this measure aims to solve: when viewed at a scaled of 0.1 units, the isolated large dot in figure 3(a) appears distinct from the other points. If we instead zoom out and observe at a much larger scale, all the points in this smaller ROI could be considered part of the same cluster. So, which scale do we choose?

Getis and Franklin [10] state that the variance about the observed mean $\bar{L}_i(r)$ for a particular r indicates how much clustering occurs within a particular pattern, and that the greatest contrast in pattern will be seen when the variance of L_i is maximised. In figure 4, we demonstrate this, with our curve showing two distinct peaks, one indicating maximal clustering as r becomes large enough for the three individual clusters to be self-contained, and finally a second peak as r becomes large enough that the all of the data are drawn into one cluster. Part of selecting a scale is some prior understanding of the data itself: this tool from Getis and Franklin is a useful tool for doing so, which we shall revisit when discussing its use in the context of biological clusters. Whilst Ripley's K function can provide insight into the global structure of the data, Getis and Franklin's function provides insight into each localisation [15].

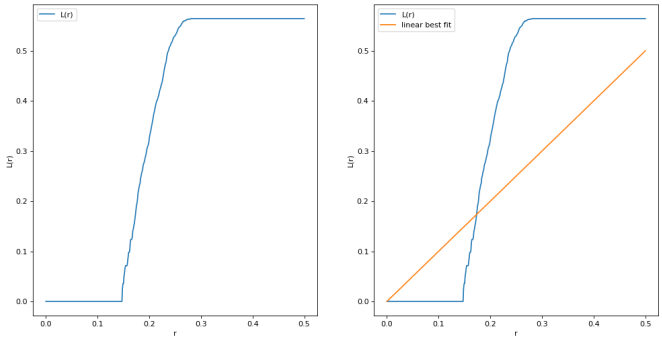
4.2 Other Noteworthy Clustering Methods

The goal of this project was to demonstrate it was possible to efficiently implement the above cluster proposal mechanism, combined with the Bayesian evaluation method outlined in section 5.2.2, to create a

[17] **include their figure, as well as a mention that this is where it becomes statistically significant.**



(a) **caption goes here** molecule.



(b) **cpation goes here**

Figure 3

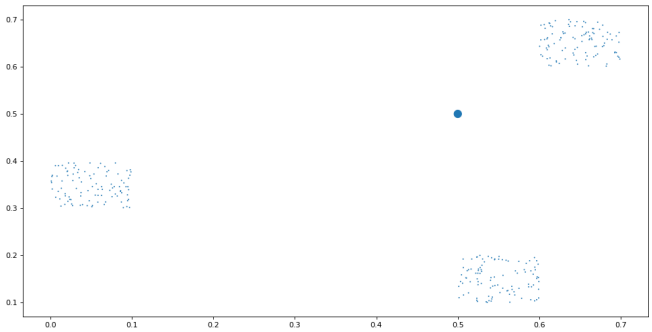
scalable method for processing SMLM data.

5 Bayesian Clustering Algorithm

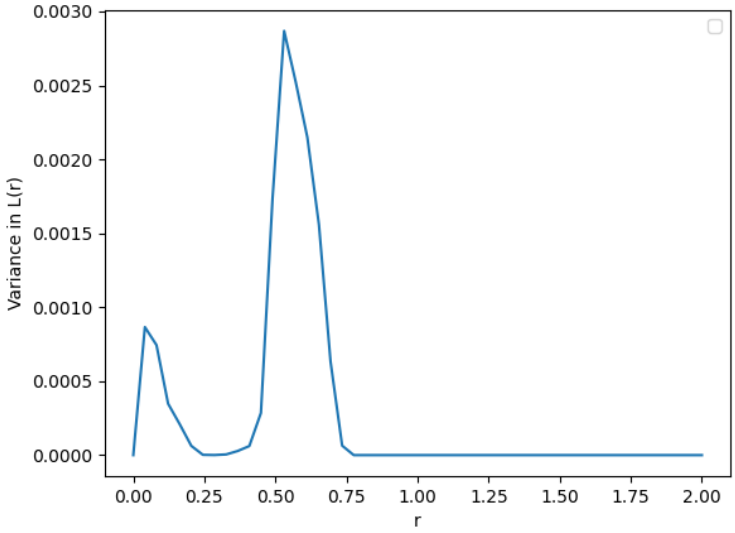
TODO

- discuss Dirichlet process (see their citation number 26 in the paper)
- not sure the dirichlet process is too important - better use of time would be discussing getis and franklin.
- how good is our model? why is it gaussian spread from the centre? see the wiki page for airy disc - we can produce a curve to show they match each other closely
- mention each localisation gets its own error from the localisation process
- essentially we draw up proposals using the second order nbhd of mapped points, and then rate them for how closely they conform to a gaussian

In this section, we will present the methodology described by Rubin-Delanchy et. al. [26], and go on to describe how it has been implemented in our repository. The process has two stages: cluster generation and cluster evaluation. It should be noted that the main contribution of [26] is a model for what the ideal cluster should look like, and a way of assessing how closely the proposed clusters conform to that model. This part of the process is agnostic towards the cluster-generation procedure, and many of the methods **discussed earlier** would also be suitable.



(a) **caption goes here** molecule.



(b) **cpation goes here**

Figure 4

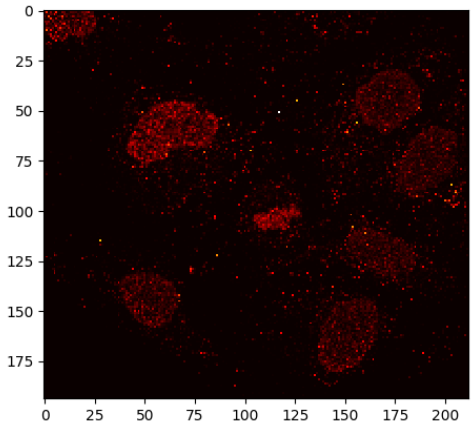


Figure 5: Low-resolution scan of **1_un_red.csv**

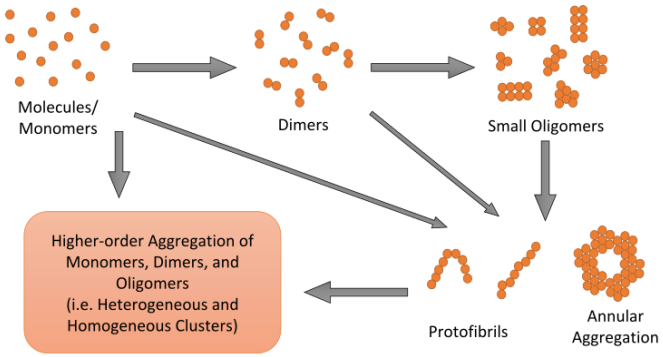


Figure 6: This figure appears as Figure 5 in [15]. It demonstrates how individual molecules can either directly form larger, more complicated structures, or perhaps go through a long chain of other processes. The analysis of the molecule clusters can provide insight into the nature of the process at hand.

The model for the data makes a few key assumptions. Firstly, the localisation process is assumed to disturb by the true positions of the molecules by errors which are Gaussian-distributed. Each localisation i has its own standard deviation value (which we denote s_i), estimated on the basis of of the number of collected photons, PSF width[18], local background noise and camera pixel size [26, 23][19]. Secondly, the clusters themselves are assumed to be uniformly distributed across the ROI, against a completely spatially random (CSR) background, with a probability distribution for their radii being supplied by the user. This distribution should be specific to the situation being imaged, and we give a discussion on this in section 5.3.

Before we begin, we outline some of the terminology and definitions that will be used going forward.

- *Localisation* - localisations are assumed coordinates for a molecule. In the scope of this report, a localisation i will be characterised by its central coordinates in two dimensions, $V_i = (x_i, y_i)$. Each localisation is assumed to be disturbed from its true molecule position Z_i by an error which is Gaussian-distributed. The standard deviation from its centre is labelled s_i.
- $\theta_k = (\mu_k, \sigma_k)$ - for a given cluster k, these are respectively the assumed central coordinates and radius of the cluster.
- $L_i(r)$ - this is the localisation score for each localisation i, as discussed in section 4.1.1.
- R - this is one of the two main parameters we vary when drawing cluster proposals. As we shall see, we determine that any two localisations which are (a) not members of the background and (b) within a distance $2R$ of each other, are members of the same cluster
- T - this is the other main parameter we vary. In each round of cluster proposal generation, any localisation whose score $L_i(R)$ is below this threshold T are determined to be members of the background.
- *Complete Spatial Randomness (CSR)* - CSR is our reference for a typical, un-clustered, uniformly distributed background. Its definition is a background generated by the Poisson point process [10], and we draw cluster proposals by comparing with CSR.
- LABELS
- POSTERIOR PROBABILITY

5.1 Generation of Cluster Proposals

To generate the cluster proposals, we need to define what it means for any individual localisation to be a member of a cluster, and what it means for two localisations to be part of the same cluster.

[18] TODO: ensure this is fully explained

[19] I have not read this second paper - it is [25] from rubin delanchy

5.1.1 Localisation Score

The generation of cluster proposals is based on the localisation-scoring of Getis and Franklin [10], as detailed in section 4.1.1. We build on this, and discuss how clusters are drawn from there.

For each localisation i, this $L_i(r)$ is a function of the number of other localisations within a distance r of V_i, normalised by the total number of pairings where our localisation of interest is one of the members, $N-1$. The value of r is incremented from 5nm up to 200nm. This upper limit is chosen since any clusters larger than 200nm are resolvable with standard fluorescence microscopy[26]. Here the constant A represents the area of the (rectangular) ROI. In our implementation of this algorithm (see section 7), the ROI is scaled down to a square with side length of 2 units.

After this score is calculated for each localisation in the dataset, any points which have $L_i < T$, for some threshold T, are determined to be background points. This is because for this value of r, the localisation i is determined to be in a region that is insufficiently dense to be part of a cluster. The idea behind this method is to compare the density around each localisation with the density expected under CSR. As such, we would expect that $L(r)_i$ being less than r would indicate points that are less clustered than under CSR [26]. This is by design: as demonstrated in section 4.1.1, we see that $L_i(r)$ is linear in r for a localisation amongst a CSR background.

This method for generating proposals was chosen for a number of reasons. Firstly, the score itself is very easy to compute, as for each localisation i, the calculation involves little more than counting its nearest neighbours, and then multiplying by a constant scaling factor. As we shall see in section 7, there are a various ways upon which this can be improved in its implementation. Secondly, this method is agnostic to the scale at which the ROI is being viewed. It's worth reiterating that any cluster-proposal method can be used at this stage of the process, but the low computational cost of this method is attractive, given that we intend to analyse a large number of them to determine the proposals that fit closest to the model we outline in section 5.2.

5.1.2 Cluster Radius

We say further that for some *clustering radius* R, that two localisations with coordinates V_i and V_j are members of the same cluster when their separation $|V_i - V_j|$ is less than $2R$.

This way of assigning molecules either to (a) the background or (b) some cluster, will assign every datum to a cluster of size at least 2, or to the background. To see this, we fix a value for R and T, and let $V_i = (x_i, y_i)$ be a localisation such that $|V_i - V_j| < R$, $\forall j \neq i$. Hence we have that $L(R)_i = 0$, meaning that this V_i is determined to be a member of the background process for all $T > 0$. If we change this so that there is exactly one V_j within a distance R of V_i (and vice versa), then (taking the area $A = 4$) we have that these two points are either members of the same cluster (of size 2) or are both background points $\forall\, T > \sqrt{\frac{4}{\pi(N-1)}}$.

5.2 The Bayesian Model

As mentioned above, the main contribution of the authors of [26] is the calculation of posterior probabilities. More specifically, once proposed set of clusters has been generated for a value of the parameters R and T, we can determine the likelihood that each point in the dataset would have been given its label, based on the supplied parameters. In this subsection, we'll dive into exactly how this is calculated.

5.2.1 Preliminaries

We begin by describing the dataset. The data are a vector of N different localisations V_i. Each of these carry an associated variance s_i^2, which denote the error introduced at the time of localisation, as mentioned in at the beginning of this chapter. We aim to create a vector $L = [l_i,\ i = 1, \ldots, N]$ of N different

labels, each one denoting membership to a particular cluster. If $l_i = l_j$, then the localisations i and j at coordinates V_i and V_j are part of the same cluster. We reserve the label $l_i = 0$ to denote that localisation i is part of the background. In general, if we identify m clusters, we should then have $l_i \in \{0, 1, \ldots, m\}$, i.e. $m + 1$ distinct labels. Furthermore, for a cluster k, let c_k denote the set of localisations i that are a member of cluster k, and n_k be the cardinality of c_k. In particular, if some localisation j is a member of cluster k, then we have that the label $l_j = k$. As we shall see, the most crucial part of this process is calculating, for a proposed set of labels L, and a given vector of localisations $\nu = [V_i,\ i = 1, \ldots, N]$, the calculation of $p(L|\nu)$. We interpret this as the probability of observing a set of labels, given the known set of localisations. It is this measure which determines the quality of a proposed set of clusters, and herein we shall refer to it as the *posterior probability*.

5.2.2 The Model itself

We now can describe the model of [26] in more detail. Our aim here is, given a proposed set of labels L (determining which localisations have been determined to be part of the same cluster, and which are background points), to establish how *likely* it is to observe this set of labels, given a model for forming clusters. We begin by noting a few key assumptions. As mentioned before, we assume that the coordinates V_i of each localisation i are disturbed from the true molecule position Z_i by a Gaussian-distributed error, with its standard deviation being labelled here as s_i. Further to this, we have that each *cluster* is modelled as a Gaussian: for some cluster with label k, we assume that all of the true molecule positions that are located in this cluster are distributed around the cluster centre μ_k with standard deviation σ_k. More specifically, we have that $Z_i \ \sim \mathcal{N}(\mu_k, \sigma_k^2 I_2)$, where I_2 is the two-dimensional identity matrix. With these two assumptions, combined with a linearity argument, we can say that we should expect that a localisation i with position V_i will satisfy:

$$V_i \ \sim \mathcal{N}(\mu_k, (\sigma_k^2 + s_i^2) I_2). \tag{2}$$

The two standard deviations have been added together, encapsulating the Gaussian shape of the cluster and the nature of the assumed localisation errors.

Our aim, as mentioned above, is to calculate $p(L|\nu)$ which, by Bayes' theorem, we can state is proportional to $p(L)P(\nu|L)$, the probability that we have some set of labels, multiplied by the probability that we have a particular set of localisations given a set of labels. As we shall outline, these two quantities can be calculated, but not necessarily in a closed analytic form. The authors of [26], following on from the work of [7], give an expression for the probability (density) that a certain labels will be issued, given the number of points in some cluster, the number of identified clusters, and the number of background points. This measure was developed through the Dirichlet process, which aims to establish some prior knowledge of the distribution of random variables. We quote the result here, from [26]:

$$p(L) = p_B^{n_0}(1 - p_B)^{N - n_0} \frac{\alpha^m \Gamma(\alpha) \prod_{k=1}^{m} \Gamma(n_k)}{\Gamma(\alpha + N - n_0)}, \tag{3}$$

where n_k denotes the number of points in cluster k (with $k = 0$ denoting the background points). The hyperparameter α is known as the concentration parameter [26]. The sensitivity of the algorithm to this parameter is yet to be established.

At this stage, we should note a few key details, following [26]. Firstly, clusters are *independent* of each other, meaning the likelihood that some cluster k has parameters θ_k is independent of any of the other cluster parameters. Secondly, the points within a cluster are identically distributed in a Gaussian pattern around the cluster centre, as described above. Thirdly, background points are uniformly distributed across the ROI. We shall add some clarifications to these points after introducing some more elements of the model.

With all of this, and Bayes' theorem, we can calculate the probability (density) that we would issue a certain set of labels $L = [l_1, l_2, \dots]$, given the set of localisations observed $\nu = [V_1, V_2, \dots]$. In particular, since $p(L|\nu) \propto p(L)p(\nu|L)$, we can state that [26]:

$$p(L|\nu) \propto p(L)\left[\prod_{i\in c_0} p_0(V_i) \cdot \prod_{k=1}^{m} \int p(\theta_k) \prod_{i\in c_k} p(V_i|\theta_k) d\theta_k\right] \tag{4}$$

It's useful at this stage to discuss the meaning of each the terms in the expression $p(\nu|L)$. Firstly, $\theta_k = (\mu_k, \sigma_k)$ are the parameters[20] associated with cluster k. Given that we expect clusters to be uniformly distributed across the ROI, we take $p(\theta_k) = p(\mu_k)p(\sigma_k) = \frac{1}{A}p(\sigma_k)$. $p_0(V_i)$ refers to the background probability density, meaning the probability that some localisation at position V_i was is a member of the background. Since we assume that background points are uniformly distributed across the ROI, we can state that $p_0(V_i) = 1/A$, where A is the area of the ROI. Since we are, at this stage, evaluating the probability that we should observe a set of localisations given a particular set of labels, for every localisation that was labelled as being part of the background, we take the product of the likelihood that any localisation was generated by the background process[21].

The next two terms are slightly more involved. The outer product runs across each of the identified (non-background) clusters. The inner product then runs over each of the points that are assigned to be members of some cluster k, where $c_k \subset \nu$. The term $p(V_i|\theta_k)$ refers to the probability that we would observe a localisation V_i given some cluster parameters θ_k. The novel part here is the integration step. Whilst it cannot be calculated in closed form, this is the step that removes the dependence on user-supplied values for parameters such as cluster centre and cluster radius. As stated in Khater et. al. [15], this is the goal of the Bayesian method - to reduce the reliance on arbitrary user-defined parameters.

Next, we turn discussion to the calculation of $p(V_i|\theta_k)$ and $p_0(V_i)$. For clarity, we will proceed by focusing on one cluster in particular, meaning we drop some subscripts. Let our cluster contain n points, have a mean (cluster centre) μ and a radius σ. This further means that the indices in our cluster are $c = \{1, \dots, n\}$ and our localisations within are $\nu = [V_1, \dots, V_n]$. We can then state that the probability of observing this set of localisations within a cluster ν, given parameters μ and σ to be:

$$\prod_{i=1}^{n} p(V_i|\mu, \sigma) \tag{5}$$

Simply put, we take the probability that a certain localisation V_i is in this cluster, given knowledge of the cluster's parameters. Since we know how the localisations are distributed (see equation 2), we can unpack this expression analytically. We first let $\omega_i = 1/(\sigma^2 + s_i^2)$. The denominator here is the standard deviation of this localisation from the cluster centre. We can state that the probability of observing a localisation with coordinates (x_i, y_i), within cluster centred at $\mu = (\mu_x, \mu_y)$ is:

$$p(V_i|\mu, \sigma) = \frac{\omega_i}{2\pi} \exp\left(-\frac{\omega_i}{2}(x_i - \mu_x)^2 + \frac{\omega_i}{2}(y_i - \mu_x)^2 \right), \tag{6}$$

due to the separability of the Gaussian over the two coordinates we have here. When taking into account the product over all the n points in this cluster, we can state that equation 5 can be written as, quoting [26]:

$$\prod_{i=1}^{n} p(V_i|\mu, \sigma) = \frac{1}{(2\pi)^n}\left(\prod_{i=1}^{n} \omega_i\right) \exp\left(-\frac{1}{2}\sum_{i=1}^{n} \omega_i |V_i - \mu|^2 \right), \tag{7}$$

[20] Central coordinates and cluster radius.

[21] The cluster label 0 is reserved for the background: c_0 is hence the collection of localisations that have been assigned to be part of the background.

where $|V_i - \mu|^2$ denotes the square-distance between the localisation centre and the cluster centre. The authors then define the weighted centre of the points to be:

$$\bar{\nu} = \frac{1}{\bar{n}} \sum_{i=1}^{n} \omega_i V_i,$$

where:

$$\bar{n} = \sum_{i=1}^{n} \omega_i. \tag{8}$$

They go on to define:

$$S^2 = \sum_{i=1}^{n} \omega_i |V_i - \bar{n}|^2, \tag{9}$$

so that the expression inside the exponential can be rewritten as:

$$-\frac{1}{2}\Big(S^2 - \bar{n}|\mu - \bar{\nu}|^2\Big) \tag{10}$$

Since $\int d\theta = \int d\mu d\sigma$, we can write, for fixed k:

$$\int d\theta \; p(\theta) p(\nu|\theta) = \int d\sigma \; p(\sigma) \int d\mu \; p(\nu|\sigma). \tag{11}$$

This allows us to rewrite the right-hand integral as [26]:

$$\frac{1}{A \cdot (2\pi)^n} \cdot \left(\prod_{i=1}^{n} \omega_i\right) \exp(-S^2/2) \int_R d\mu \exp\left(-\frac{\bar{n}}{2}|\mu - \bar{\nu}|^2\right), \tag{12}$$

which can be evaluated with the standard Gaussian cumulative distribution function. The final step is to perform the sigma-integral in equation 11, which is typically done with numerical integration. The domain is a user-supplied distribution of the expected cluster radii. A discussion on this prior follows in section 5.3.

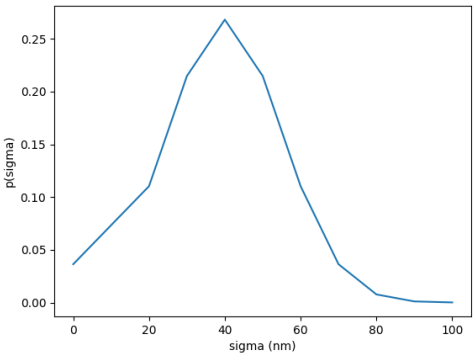
It is the calculation of these integrals which evaluates how closely a proposed set of clusters conforms to the model outlined above. In the implementation of this whole procedure, the complexity of the evaluation of the numerical integrals is reduced by choosing a representation for the data that allows for calculations to be performed with fewer operations. A description of this is given in Appendix A.

5.3 The Sigma Curve

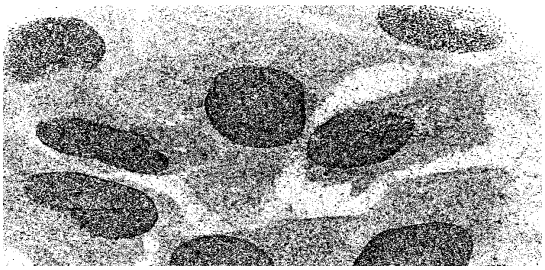
to mention - state explicitly how this penalises/rewards finding clusters of a certain size
One area where this method relies on user-input is the range of values of σ to be integrated over. This should be a probability density curve that will, in some sense, *bias* the results towards identifying a cluster with a particular radius. This requires some understanding of the sample being worked upon to establish reasonable values for this prior. STORM imaging has led to healthy development of information in this regard.

5.4 The Limitations of this model

One key limitation of this model is the assumption that the clusters take the shape of a Gaussian. This is a key component of the mathematics: the fact that we assume the localisations are within a Gaussian-ring of the true molecule centres, *and* that the molecule centres are within a Gaussian-ring of the cluster centres, we can use linearity to combine these "errors" into one Gaussian distribution and evaluate the necessary integrals using well-established numerical techniques. This method could be adapted to take a different probability distribution that describes the shape of non-Gaussian cell clusters, but it's unlikely that such a



(a) **caption goes here**



(b) **caption goes here**.

Figure 7

distribution would have a known cumulative distribution function: without this, the leap taken to arrive at equation 6 would be highly nontrivial, and the result would look entirely different. The authors are aware of this, and state in [13] that any model is likely to have shortcomings, and whilst a Gaussian pattern provides a good fit for a wide range of situations, it's unlikely to be useful in identifying structures that are elongated or non-convex.

what if the background density is different in different places??

6 Analysis of the complexity of the described algorithm

TODO:

1. discuss where the bottlenecks are
2. be consistent about placement of the dimension subscript eg μ_d
3. for consistency, change the index of clusters to "k" (not i)

Having outlined the algorithm in Section 5, we turn our discussion to its practical implementation. A number of elements of the above calculation can be simplified to allow for computation that is efficient with respect to memory and other resources. In subsection **REF**, we shall rewrite the integrals above in terms of parameters which, as we shall demonstrate in section **REF**, allow a much more efficient use of resources when computing over a large dataset.

6.1 The Algorithm, as outlined in the paper

todo:

- find an actual template for the algorithm to be presented in a figure
- discuss the data collection and read-in at some point before this
- ensure localisation precision are discussed in the ThunderSTORM part
- does the above mention the two sources of the uncertainty term??
- mention the instabilities and differences between methods
- mention known limitation - we don't have high enough res to see where the actual maximum might be
- we calculate all of the localisation scores at same time as read-in

In this subsection, we state the algorithm as outlined in the paper, before discussing in subsequent chapters how it can be made more efficient, as well as outlining its practical implementation in the current **codebase**.

The first step is to read the ThunderSTORM data in. As **discussed** above, the data come with three important characteristics for each molecule localisation V_i - a resolved central point (x_i, y_i) and a localisation uncertainty, all in nanometers. Each localisation is given a localisation score, as discussed in section 5.1.1, which is fixed for each point for the rest of the calculation, as it is a function of the number of points in immediate vicinity. Our aim at this stage is to determine which values of the clusterisation radius R and the localisation threshold T are the most **reasonable** given the data at hand. The authors scan each of these parameters from 0 to 200nm[22], and from 0 to 500 respectively, meaning that for each value of R and T in these specified ranges, clusters are drawn, and then scores are calculated. More specifically, the threshold T is first set to some value in the range (0, 500)[23]. Each datum with a score below this value of T is determined to be part of the background at this stage, and is ignored until the next two values of R and T are determined. Each remaining datum is thus determined to be a member of a cluster of size 1 (i.e. containing just itself as a member.) The data are then scanned once more: any two localisations which are within a distance $2R$ of each other are brought together, as members of the same cluster. At this stage, we can calculate the probability that this set of labels[24] would have been issued, given the set of localisations that were observed, using the expression given in equation 4.

6.2 Proofs of correctness

6.2.1 Why is this a good way of drawing clusters? Does it conform nicely to the Gaussian assumptions?

our main tool here may be to compare to another, better clustered dataset.

6.2.2 Does this produce the optimal R,T pair?

7 The Algorithm as implemented, accounting for speedups.

todo:

- describe non-repetition of log score calculation
- add a nice diagram for the clusterisation section
- we could even skip T values that are less than r since these are going to be part of the background anyway
- also mention polynomial edge correction for the localisation score calculation
- mention that we zoom in on just the cell nuclei to produce a bespoke R,T score for each region – we can also look into scanning across the nuclei or the whole image in little windows
- mention that everything was done on lo

The algorithm described above has been implemented in C++, primarily by Dr Andrew Rose, for the purpose of utilising all the most important features of this language, such as multithreading, high-speed memory read and write[25] and the ability to perform large parts of the computational work at compile-time, rather than at run-time. Careful uses of the features of this language, alongside some important algorithmic adaptations, mean that this implementation of the algorithm can process large amounts of data efficiently.

[22] discuss why this is a sensible range
[23] Talk about skipping the early parts of the threshold
[24] Labels here meaning, membership or non-membership of some cluster for each datum.
[25] Due to its contiguous memory allocation

The bulk of the computational work comes from performing the numerical integration described in section 5.2.2, but great improvements in efficiency can be found in careful variation of the R and T parameters.

7.0.1 Calculation of the Integrals

In section 5.2.2, we outlined the method that the authors of [26] presented for evaluation of the posterior probabilities for a set of proposed clusters. In their methodology, the process should go as follows:

- Fix a pair of values (R,T)
- Assign each point to either the background, or to one of m different clusters
- Once the cluster members are fixed, go through each cluster, calculating the two parameters $\bar{n}$ and S^2 from equations 8 and 9.

There are ways to substantially reduce the runtime of this section however. In appendix A we see that the above integral can be rewritten in terms of four variables to track which, crucially, allow us to calculate the mean weighted centre etc. whilst the localisations are being brought into the cluster, rather than having to re-read the list of localisation coordinates *after* the cluster has already been drawn.

Comparing the implementations of the two methods (one as outlined in the paper, the other making use of speed-ups when tracking variables) revealed a slight discrepancy (by a small factor) in the value of the integrand just before performing the integral with respect to sigma, but this was small enough to have negligible effect on the posterior probabilities themselves.

7.0.2 Clusterisation

We turn our discussion first to the way in which clusters are constructed. Starting with a set of non-background points, we group together all the points which are within a distance $2R$ of each other. The naïve method for doing this involves passing through the set of points once, and, for each point in the set, checking which of the others are within a distance $2R$. This requires quadratically many operations before ensuring that the minimum number of clusters are returned[26]. This also doesn't take into account the fact that **the cluster size might grow as we continue**.

To eliminate much of the parallel work, we begin by transforming the representation of the data. The output of ThunderSTORM, as mentioned early, comes in the form of x and y coordinates within the field of view (FOV). Upon read-in we instead transform them into polar coordinates with respect to one corner of the FOV[27], before sorting each of the data-points by radial distance from the origin. This has a few benefits. First, it reduces the amount of redundant works that needs to be done, since a datum's distance from the origin can be used to infer its distance from another datum (or rather, can be used to avoid doing unnecessary work when they are certainly too far apart). Second, it allows for parallelisation. Since the data are sorted by radial distance, points which are closer to each other can be handled by separate threads. Finally, the polar coordinates can again be used to reduce the number of computations needed by using each datum's angle with respect to the corner of the FOV: two data with sufficiently large angles with respect to the x axis would not be able to be neighbours, as demonstrated in figure 8.

One unexpected place in which a noticeable speed-up could be found was in the edge correction of localisation score calculation from equation 1. A Taylor-expansion of this expression in $\arccos(e_i/r)$ leads to, at first order:

$$1 + \arccos()$$

[26] i.e. that we don't create a new cluster every time we observe two points within this distance of each other.
[27] are we sure it isn't the centre?

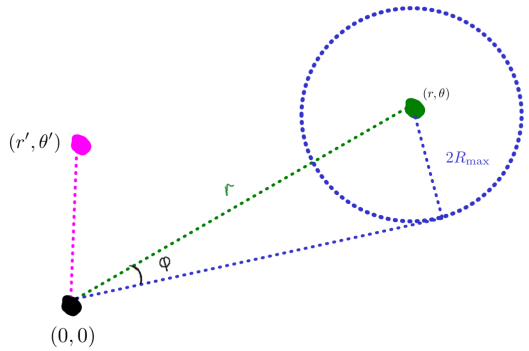


Figure 8: Demonstration of an important speed-up when populating the list of neighbours of a given localisation. When searching for neighbours of the localisation located at the black dot, with polar coordinates (r, θ), we first calculate ϕ as the inverse sin of $r/(2R_{max})$. We can then safely say that any other point with polar angle θ' will only be a candidate for the neighbour of the point at (r, θ) if $\theta - \phi < \theta' < \theta + \phi$, otherwise such a point would not be within a distance $2R$ of the point in question. In particular, this condition on θ' determines whether or not some other localisation could be within the blue circle of radius $2R_{max}$, which is the uppermost distance any other localisation could be from (r, θ) to constitute a nearest-neighbour.

7.0.3 The scan over R and T

todo

- also mention that the localisation scores are fixed only for each R bin

We can now discuss in more detail how we can determine and evaluate the clusters visible in the data whilst performing minimal redundant work during the clusterisation process. Ultimately, we require a two-dimensional scan - a value for $p(L|\nu)$ must be determined for each (R, T) pair in our range of values for these parameters.

First, we fix R. We scan *forwards* through the domain of R, meaning that on each iteration, we increase R by some fixed amount (i.e. we send $R \to R + dR$). For this value of R, we fix a value of the threshold T, starting at its upper limit and working backwards towards the lower limit. The reason for computing in this order (rather than fixing T at each step and then scanning through R) is that it doesn't require the re-drawing of the clusters. When we send T to $T - dT$, data that were previously determined to be members of the background (due to their localisation score being too low), can be simply added to existing clusters, should there be some cluster that contains a datum within a distance $2R$. Moreover, if reducing the threshold doesn't add any new points to an existing cluster, we need not recalculate its integral with respect to the cluster parameters μ_k, σ_k.

7.1 Parallelism

Much of the implementation exploits parallel processing, and as we shall see in section 8.5, as we increase the number of threads available to the program we find that the eventual runtime is bounded below by the single-threaded sections.

We begin by describing the main sections where we exploit **concurrency**. Firstly, reading in the data from ThunderSTORM is sped up massively using **multi-threading**. The data that was used to test and validate the first version of the software contains more than 9 million lines. Each of these must be scanned to extract the central coordinates of the fluorophore, as well as its uncertainty, before determining if such a point was in the **scan area**. Thereafter a `Data` object is created to contain it, which also calculates some useful additional information for each datum, such as the localisation's polar coordinates with respect to

the ROI. This section was a clear candidate to be parallelised: the length of the input file is first determined, so that sections of the data can be apportioned and assigned to individual threads. At this stage, no arbitration is required between threads since the data need only be added to some other container at this time.

Another section is the scan across R and T. As mentioned in section 7.0.3, we fix a value of R, and calculate all posterior probabilities as we vary T from its maximum value to its minimum. Once the data have been read in and pre-processed, the software establishes how many values for R need to be scanned over footnoteMost experiments in this report were conducted by varying R from 5 to 200nm in 100 steps., and crucially the number of threads that the available CPU has to offer. Each of the portions of the range for R are then apportioned and sent to each of the available threads. In section 8.5 we detail the decreases in runtime that can be achieved by doing this compared to a single-threaded approach.

One situation in which parallelism could not be utilised fully is in the calculation of the neighbours of each datum. As a reminder, two data are *neighbours* if the distance between them is less than $2R$, a parameter which varies as the scan continues. To save time, each datum is given a list of neighbours based on the maximum value of $2R$ we expect to use, which is entered when the script is initiated. Since the only criterium is that they are sufficiently close to each other, this can be used to halve the number of calculations. Once it's established that $|V_i - V_j| \leq 2R_{\max}$, we can add V_i to the list of neighbours for V_j and vice versa in one calculation. One issue with that is, as discussed in section 7.0.2, the data are sorted with respect to distance from the centre of the scan area, before fixed-size intervals of **radius from the origin** are passed to separate threads. This means that, at the boundary of an interval, we may find two data which ought to be neighbours, but which are being handled by different threads. Implementing this reciprocal calculation would then require either **arbitration** between threads to establish which data can be added to neighbour lists, or simply passing this section on one thread. Experimentation revealed that brute-forcing was the quickest way of handling this section: we divide up the data, sorted by radius from origin to each thread. Each thread then builds the neighbour list for each localisation in its domain by looking at each other datum (making use of the speed-ups mentioned in section 7.0.2) and adding it to the neighbour list if the right conditions are met.

7.2 Complexity Analysis

to mention:

- number of cluster proposals to evaluate
- where the hot sections are
- mention how time increases with increased number of proposals

The time and memory requirements of this implementation of this algorithm are greatly favourable compared to previous work. Through efficient use of concurrency, and minimising the number of calls to expensive functions, large speed-ups have been produced compared to prior implementations. A region containing 600,000 points takes around 2 minutes to run on a standard 8-core CPU, which is substantially quicker than times quoted in [30].

The most intensive sections are:

- Determining the list of neighbours for each datum
- Performing the scan across the space of allowed R and T values.

One key optimisation that has been utilised in the development of this code was, for each datum, determining its list of nearest-neighbours once, at the start of the runtime, and sorting this list by distance from said datum. The reason for this is twofold. Firstly, it allows for the computation of all localisation scores in the initial stages of the run. Recall that each datum needs to be assigned a localisation score

$L_i(r)$, which is a function of how many neighbours this datum has within a distance r. In accordance with the authors of [26], we test 100 R-values between 5nm and 200nm at evenly-spaced intervals. So, equipped with a sorted list of each datum's nearest neighbours (which are all at most 200nm away), we can quickly generate the entire list of 100[28] localisation scores for each datum.

Secondly, it allows for rapid drawing of clusters. When a datum has been identified as being a member of a cluster[29], adding new points to the cluster is simple; iterate through this datum's list of neighbours, adding each to the cluster, until a neighbour is encountered that is further than $2R$ away; repeat this process for all the neighbours added into the cluster in the previous step.

One might be worried that the process of producing the list of neighbours for each datum would entail great time and space requirements. In the worst case, it would be necessary to perform $\mathcal{O}(N)$ comparisons for each of the N data that were read in from the SMLM point cloud, giving it quadratic time complexity, alongside $\mathcal{O}(N \log(N))$ time to perform the sort with respect to the centre of the ROI[30] and quadratic storage complexity, since it's necessary to store $\mathcal{O}(N)$ neighbours for each of the N data. These are massive overestimates, however. It would not be necessary to perform $\mathcal{O}(N)$ comparisons for each of the N data. As mentioned in section 7.0.2, we only perform comparisons when the two data are sufficiently close to each other[31], and since the list of data is sorted by distance from the origin, we can safely exit execution of the produce to populate the list of neighbours. As mentioned previously[32], the conversion to polar coordinates with respect to the ROI centre allows at least $1/6^{th}$ of the potential neighbours to be ignored, due to them being unable to be within a distance $2R$ of our point of interest due to their position with respect to the origin. It is worth noting further that the list of neighbours is going to be significantly smaller than $\mathcal{O}(N)$ for each datum, due to the relative size of the ROI compared with typical maximum values of R. A full estimate of this algorithm's complexity is beyond the scope of this report.

The scan across R and T takes up the majority of the runtime. As mentioned in section 7.0.3, we increment R *forwards* and decrement T *backwards*. This requires clusters to be re-drawn only when we increment R[33]. Decrementing T will usually increase the size of a cluster, as each cluster member is likely to have neighbours that (a) are within the *clustering distance* of $2R$ units and (b) whose localisation scores are too low to be considered to be a member of a cluster. The complexity of this section is hard to establish: decrementing T could require many operations, bounded above by the sum total of all elements in the list of nearest neighbours for each element already in the cluster. The likely complexity is much lower.

One of the most intensive sections is the calculations of various numerical integrals: fortunately the persistent state of a cluster during a scan means that these posterior probabilities are only re-calculated when the number of elements in a cluster changes, either when new points come above threshold, or when clusters accumulate due to their proximity. The real implementation of these sections relies on calls to non-standard libraries, used for interpolation and integration, the precision of which is increased with increasing resolution on the prior radius, σ. The estimation of the time required to perform these calculations is again beyond the scope of this report[34].

7.3 Yet to be implemented

We could avoid scanning R values below T - these are indistinguishable from CSR. Getis and Franklin [10] designed the localisation score measure to satisfy $L_i(r) = r$ for points generated by the Poisson process.

[28] Or however many proposed values for R are desirable.

[29] i.e. that its localisation score is above the threshold T for a particular value of this parameter.

[30] As mentioned in section 7.0.2

[31] If the difference between the origin-to-datum distances of our point of interest and a potential neighbour is larger than $2R$, then we know they will never be members of the same cluster directly.

[32] be explicit with this part

[33] it's because threshold very low at the end of the T run. note also that localisation scores are fixed for a value of R

[34] [30] gives a little more detail on the complexity considerations.

Unless determining whether a pointillist dataset resembles a CSR, looking for clustering with $T \leq R$ is not necessary and these portions of the computational work can be avoided.

We may also wish to introduce some penalty to the posterior probability to *encourage* the algorithm to fall inside some desirable bounds, such as number of localisations per cluster.

see paper in comment for a measure of biological meaningfulness

7.4 Contributions and Validation

discuss how we found the NaNs

8 Experimentation

todo:

- describe the datasets we run over quantitatively
- describe the hardware we run on
- describe the runtime etc
- include figure for how we reduce the roi to save on runtime
- can we mention the paper in comment?

In this chapter, we present our results from the computational experiments performed using the high-performance tool described above.

8.1 Timings

Our aim was to process a 96-well plate in six colour channels in under 24 hours. We present evidence here that using a supercomputing cluster will certainly facilitate this. Figure[35] shows a scan of some nucleosomes, which have been labelled with Alexa 647. Clearly visible are at least 8 nuclei. One naïve way of performing this is to simply scan over the whole region at once, but there are a number of issues with this. Primarily, the idea of of performing this kind of scan is to produce a pair of parameters that will identify clusters within a given ROI, on a meaningful scale. A reasonable pair of values might be three orders of magnitude smaller than the width of the ROI. Moreover, the characteristics of each cell might be vastly different between nuclei, and perhaps within nuclei as well. Moreover, as we discussed in the section on complexity[36], processing possible tens of millions of data-points at once would be prohibitively costly in memory and time.

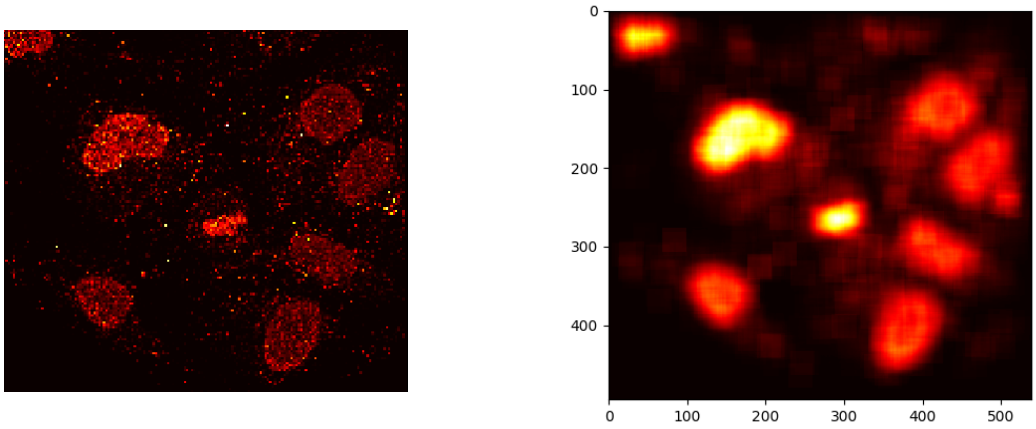
The first approach was to simply cut down the ROI. A separate scan would be run over each nucleus. The nucleus can be identified manually, but for the sake of running automated scans, we outline a tool that can be used to do so automatically. A coarse scan of the image is taken as a two-dimensional vector, before performing a Gaussian blur. The effect of this can be seen in figure 10. This causes regions of higher pixel-density (and hence with more localisations) to be brightened and less dense regions to be darkened. After setting a minimum pixel brightness threshold, the identified nuclei can be seen. From there comes a simple breadth-first search[37] to determine the ROIs necessary to scan over each of the nuclei.

We outline here the results of scanning across the various nuclei shown in figures[38]. The localisations within these regions of interest were scanned, one at a time, on a single 64-core CPU. Cumulatively,

[35] include it
[36] do so concretely
[37] include this as an algorithm
[38] put these figures in

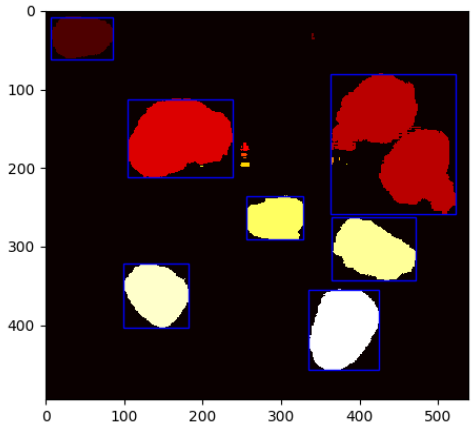
these regions contain over 5 million points, with individual nuclei containing between 680,000 and 2.1 million localisations. The total time taken for all of the processes to run sequentially was less than 15 minutes.

Further to this, we demonstrate the runtime of a variety of different scans, each performed over the same nucleus with the same central focus, but with progressively wider regions of interest and hence progressively more points encapsulated in the scans.



(a) **caption goes here** (b) **caption goes here**.

Figure 9



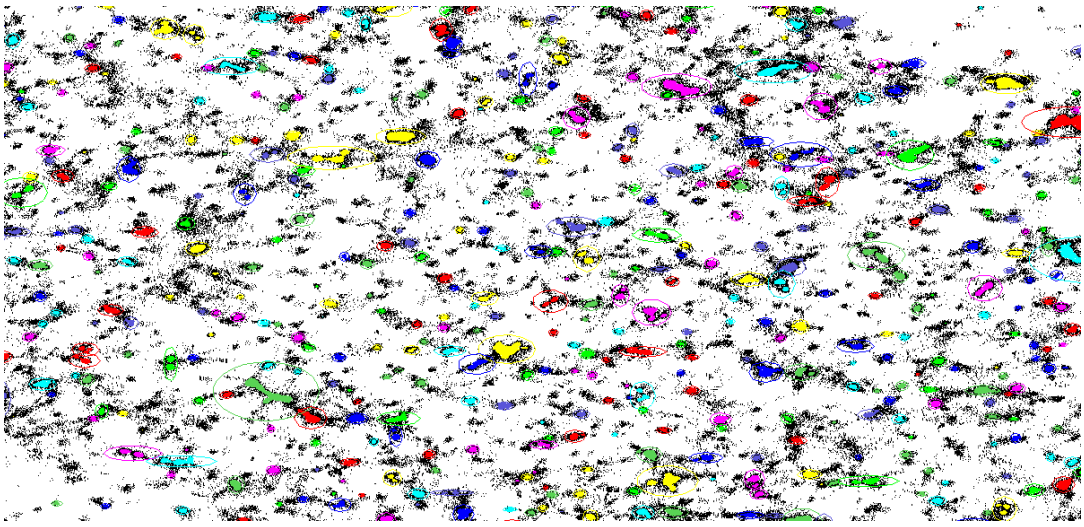
(a) **caption goes here**.

Figure 10

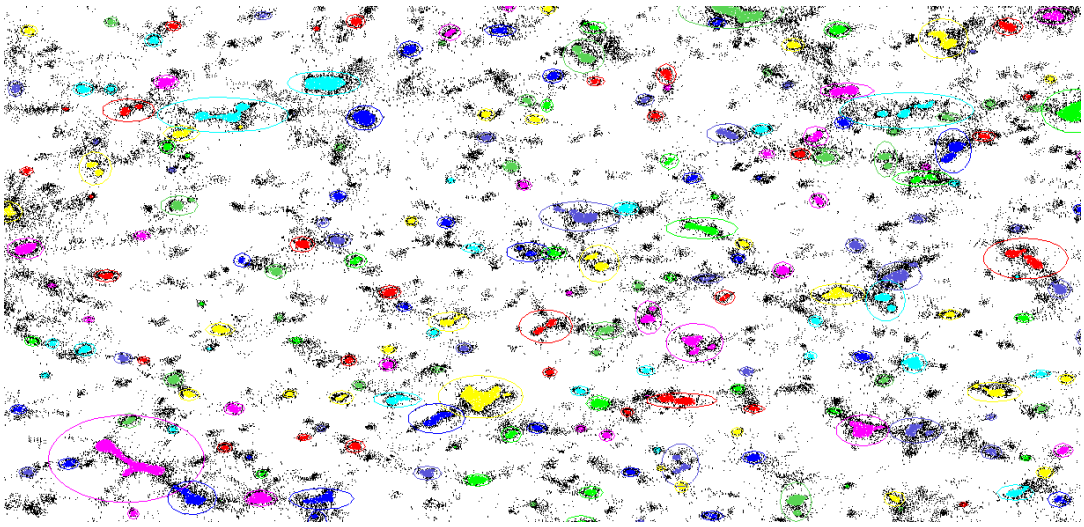
8.2 Memory Usage

We present here a short demonstration that the peak memory usage of the procedure is linear with respect to the number of localisations being analysed at once. Figure 23 shows a scan of a nucleus[39]. The clustering procedure was run on square ROIs, with widths varying from 1μm to 25μm, capturing between 2500 and 930,000 localisations. 23(b) demonstrates a clear linear relationship between the peak memory usage and

[39] which kind



(a) **caption goes here**



(b) **caption goes here**.

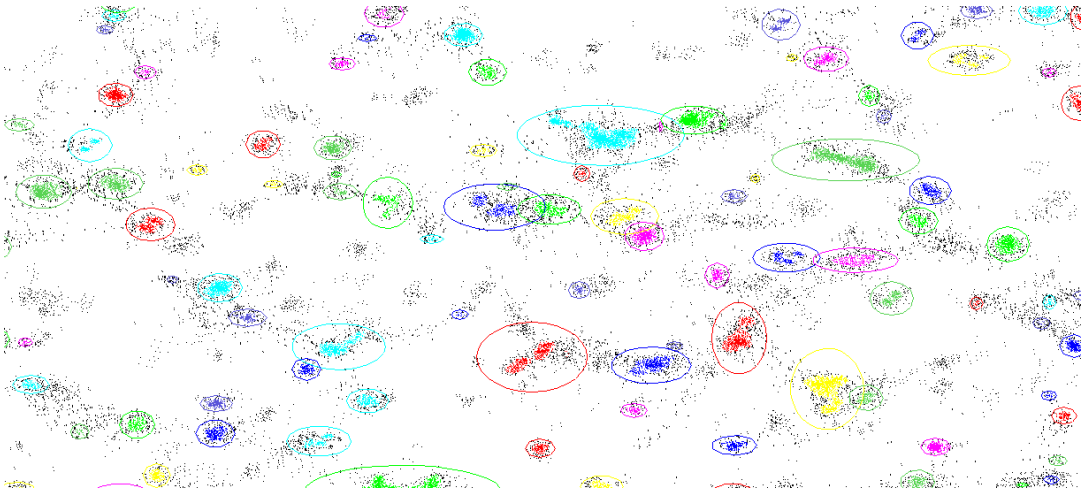
Figure 11

number of points. Processing 930,000 points had a peak memory usage of 63.8GB, which may be important for future sizing of jobs.

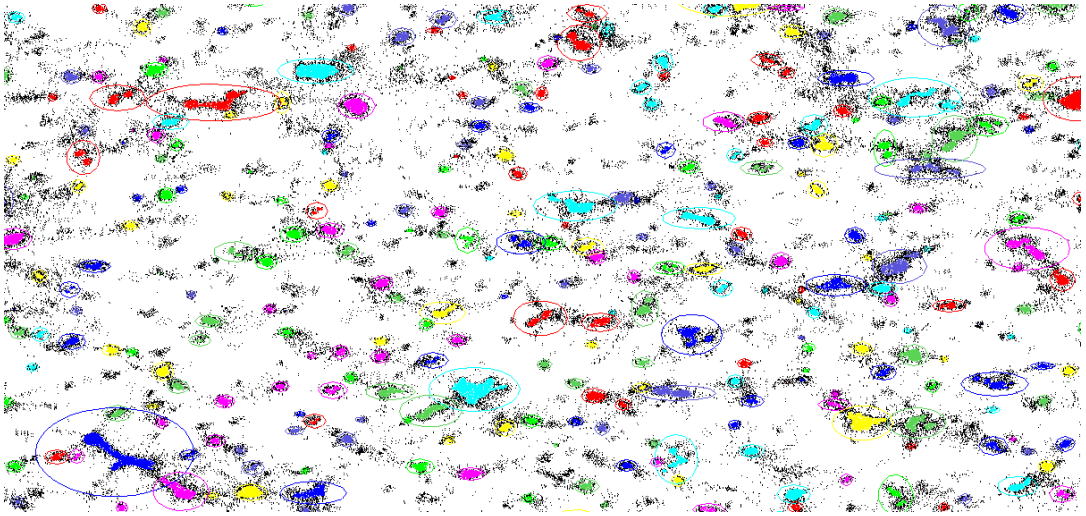
8.3 ROI Sizing

The results from section 8.2, also demonstrated that the parameters R and T returned by the procedure can vary somewhat substantially when altering the region of interest. In particular, as we included more points into a larger ROI, with both values for R and T rising until plateau, before a slight fall as the entire nucleus was brought into the field-of-view. An interesting question to answer for a future work is whether the nature of clusters in biological systems is highly specific to individual regions within nuclei, or whether clusters formed are characteristic of a more macro-process, such as the regions between and around cells. When scanning each of the visible nuclei in section 8.1, each produced a different R-value, but a fairly consistent T-value. We present figures for each in the[40].

[40] appendix



(a) **caption goes here**



(b) **caption goes here**.

Figure 12

8.4 Nuclear Pore Testing

to include: analysis of what r and t combos yield certain results, and all of the rubbish results we had to sift through

In this section we describe using this cluster-identification process on a well-understood dataset which can serve as a ground truth. Given that we know many of the essential characteristics of this dataset, such as expected number of localisations per cluster and expected cluster radius, we can use this to evaluate the performance of this algorithm[41]. We can also, using the clusters we identify, measure the error incurred during the imaging process, by comparing the width of the cluster identified in the image to their known width.

8.4.1 Nuclear Pores as a reference standard

One persistent issue in the study of imaging biological systems is their seeming lack of reproducibility. Many aspects need to be taken into account when analysing samples, such as the precise settings of the optics equipment and the fluorophores used in labelling. Thevathasan et. al. [29] are aware of this, and

[41] move this sentence

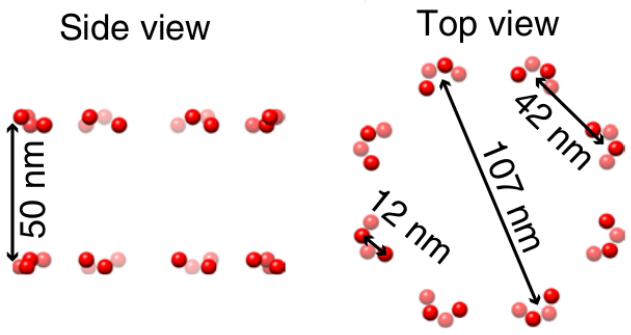


Figure 13: Part of figure 1 from [29]. The nuclear pore complex (NPC) has predictable characteristics - we typically find groups of the protein Nup96 forming octagonal rings, with two octagons stacked atop each other. These rings have predictable characteristics: the corners consist of a proteins with a width of 12nm; the side length of this octagonal shape is consistently 42nm, with diameter of 107nm.

discuss which commonly-imaged cell structures can provide some kind of "ground-truth" when imaging cells with SMLM. They make the case for the nuclear pore complex (NPC) as an ideal reference structure, as it has consistent characteristics that allow for extraneous parameters to be optimised, given the knowledge of what to expect within the region of interest itself. In particular, the NPC contains roughly 30 different proteins, for which a high resolution electron-microscope map already exists.

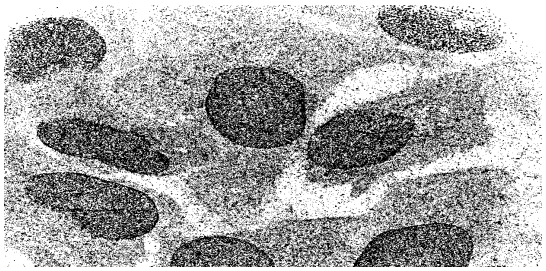
Figure 13 (reproduced from [29]) shows some of the characteristics we can expect from nuclear pore complexes, namely rings of proteins with consistent diameters. This allows us to tailor our prior probability curve for sigma, as described in section 5.3[42]. Figure 14 shows a scan from some SMLM-acquired images in both the wide-field of 40μm by 40μm, as well as a much closer scan of 5μm by 5μm. Highlighted with red boxes are some locations of the nuclear pore rings described above. The images taken from them are not precise - the rings themselves contain too many localisations, as well as their overall pattern has been blurred away from a neat octagon. There are many possible explanations for this. Firstly, the imaging technique at hand takes places stochastically over several minutes, meaning that fluorophores may drift away from their positions at some time between their first emission and the end of the final imaging cycle. This means that a fluorophore that blinks multiple times in different positions cannot be easily filtered from the data, as it may appear as more than one distinct localisation. Secondly, these images were taken from 3D structures, and whilst TIRF[43] microscopy aims to draw an image from a flat plane, this doesn't always prevent some photon emissions from depth being found in the final localisation table.

In the following section we will detail how this implementation of the algorithm performed in the identification of clusters on NPC data.

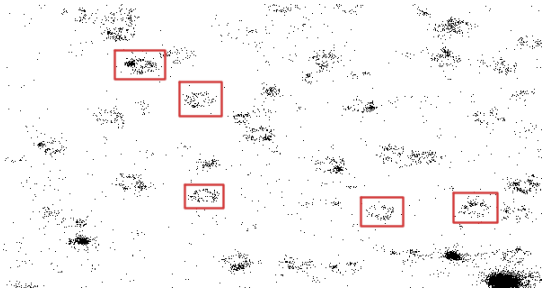
8.4.2 The experiments themselves

For this part of the experimentation, we used a localisation table which was gathered from a **nuclear pore sample**[44]. Figure 15 shows an image taken from the localisation table. The field of view is approximately $80\mu m$ by $80\mu m$. Clearly identifiable are at least 8 nuclei, surrounded by a lot of noise, most of which can be attributed to the **cytoplasm background**. We see here that, in spite of TIRF microscopy being able to image a fairly "flat" plane in 3d space, we see here a lot of localisations from other depths. As we zoom closer in to a section of the image, as shown in **figure x**[45], we can begin to identify the nuclear pores

[42] inculde the sigma curve here
[43] make sure this is explained
[44] Talk about this one in particular
[45] add a zoomed in figure somewhere here



(a) **caption goes here**



(b) **caption goes here.**

Figure 14

themselves. In these images, the distinctive **8-localisation** structure is tricky to identify. Rather than forming an octagon, most of the pores we can see have been blurred into a ring, or loop.

There are a number of possible explanations for this. The SMLM process itself can introduce errors. Over the few minutes of imaging time, molecules are prone to drifting, and later releasing another photon during a subsequent imaging round. In combination with this, the process through which localisations are generated (by detecting the parameters of a Gaussian within the pixelated point spread function) can introduce further errors. Some simple filtering methods were not effective: ThunderSTORM localisation tables come with a "sigma" parameter, which describes the width of the observed PSF. We expect Alexa-647 to produce PSFs of width 160-200nm[46]. Filtering for this still produced noisy, inconsistent data.

The blur of the imaging process can be seen in figure **??**. We have a close-up image of a likely nuclear pore. The centre still has an identifiable "gap", forming the shape of a ring. We have far higher localisations here than expected. Even when taking into account the number of localisations per cluster we see in our ground truth data[47], we have approximately ten times more localisations here than expected from a nuclear pore. Some of these will be background points, as **nuclear pores are three-dimensional structures**, which perhaps highlights the limits of precision available with conventional microscopy.

Performing cluster analysis was much less straightforward on this dataset compared to histone data, such as was discussed in section 8.1. To summarise the method used there, we began by using a Gaussian blur to identify the dense regions (the visible nuclei) before reducing the ROI to enclose only one nucleus and running the scan to produce an optimal R and T value for this nucleus. In the case of the nuclear pore data available, scanning over a region much larger than 25 square μm caused the posterior probabilities[48] to indicate highly unrealistic values for R and T. As a result, the scans were performed with a *sliding window* of $4\mu m$ by $4\mu m$ with the hope of finding an optimal pair of values that can identify clusters over the whole dataset. An important fail-safe when evaluating clusters is to ascertain that areas which are clearly clustered together are identified by the algorithm: eye-balling is often a good place to start if otherwise unsure.

Picking an appropriate probability density curve for cluster radius, sigma was slightly tricky also. As discussed previously, prior study of nuclear pore complexes has revealed a few distinct, predictable substructures visible in the images. Figure 7(b) illustrates this: we aim to prioritise the larger clusters of width 107nm, as well as the smaller clusters of width 12nm. The curve for the first round of scans can be seen in figure 16. Some results were initially promising, with multiple nuclear pore clusters being identified in the FOV. However, the results of the algorithm appeared too heavily biased towards large threshold values. This meant that whilst the values for R led often to sensible clusters being identified

[46] cite and stuff

[47] as mentioned beforehand

[48] Posterior probabilities indicate which (R, T) pair is optimal.

in some regions, many others were missed. This was due to the T parameter being too high: any other potential clusters were simply labelled as background points[49].

To counter this, multiple new scans were done, but instead strictly limiting the scan values for R and T, after observing the cluster behaviour when manually modifying the parameters. I found that threshold T being too high was the main issue. Finding a pair of values that produces a few clusters[50] was not too difficult, however these values would cause the cluster-drawing to avoid many of the clusters which appeared to be obvious to the naked eye. As can be seen in figure[51], increasing the value of R beyond realistic values simply causes multiple pseudo-clusters to be brought together, whereas in figure[52], increasing the threshold allows more clusters to be identified.

Rather than the classic scan ranges, of 0 to 200nm for R and 0 to 500 for T, the ranges for subsequent scans were restricted to 20 to 150nm for R and 0 to 200nm for T. In spite of using a curve which is perhaps non optimal, one particularly good pair of values was R =28nm and $T = 141$. Figure[53] shows the identified clusters at a range of ROI widths. We can see at this values, a good proportion of the clusters have been located, but we search still for more optimal values. 17

These results were built on by adjusting the sigma curve to the one found in figure 17. The idea behind using this, modified curve was to encourage the algorithm to *reward* identifying clusters with those particular radii.

The dataset used was quite tricky to analyse. Correctly tagging nuclear pores with Alexa 647 is time-consuming, and can is prone to failure[54]. Moreover, localisation density within nuclear pores is not uniform, suggesting that this method is not suitable here. Clustering, in this model, is determined by comparison with CSR[55] which may not be possible if rate of change of localisation density with respect to any of the coordinates is too large. This may also preclude the use of a small sliding window to determine optimal (R, T) parameters: how does one generalise the use of a pair of values to any region outside where it was determined. Beyond the scope of this report is a discussion on whether better data may be gathered for such an analysis, and whether the algorithm can be adapted to be more suitable for this scenario, perhaps by introducing some pressure towards a certain number of localisations per cluster[56]. Possible avenues of investigation for a future work include using the width clusters identified by the Bayesian procedure (whose width can be measured computationally) to provide some estimate of the errors that occurred during the imaging and localisation process.

8.4.3 Synthetic Data

A much more clear view of the performance of this model can be seen when looking at synthetically-produced data, carrying on from the work of [30]. Their idea was to analyse the performance of clustering algorithms against a clean dataset with known priors. We present here the results of scanning over 9 datasets, each of which have a few (varying) parameters with which we can assess the performance of the Bayesian method. These parameters include:

- The number of points in the FOV

[49] Include some figures to illustrate what is meant
[50] see figures
[51] add said figure
[52] add this fig
[53] include it
[54] check this
[55] Or in other words, uniform randomness
[56] mention this in the outro - also mention that they found the sigma curve doesn't really matter, so this hampered efforts to makes sure it was good

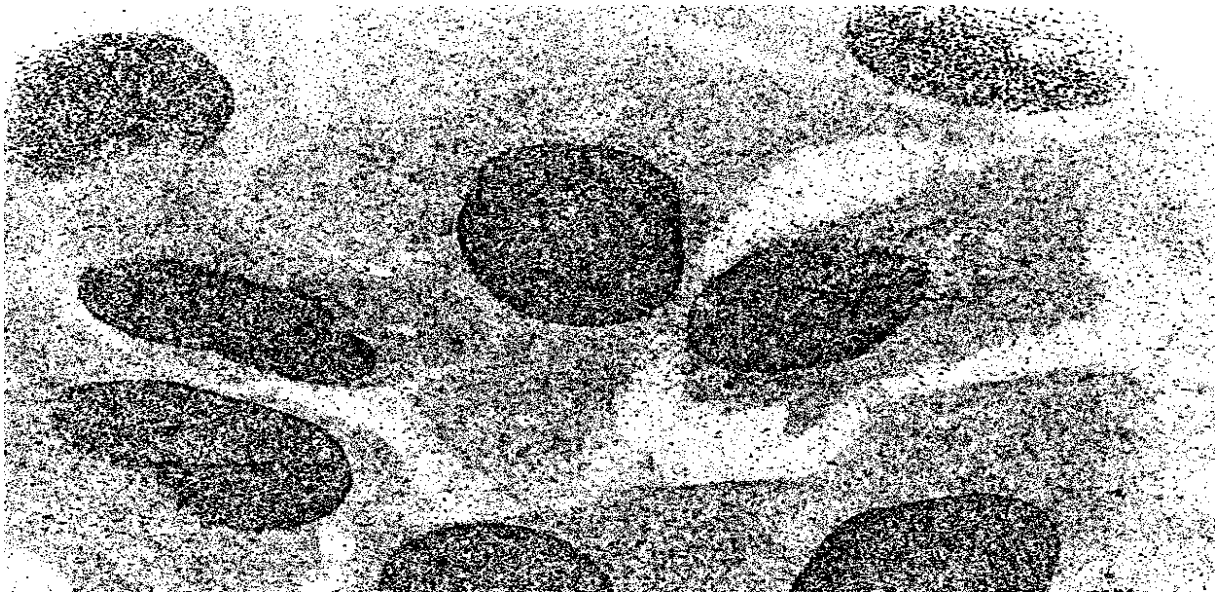


Figure 15: Caption

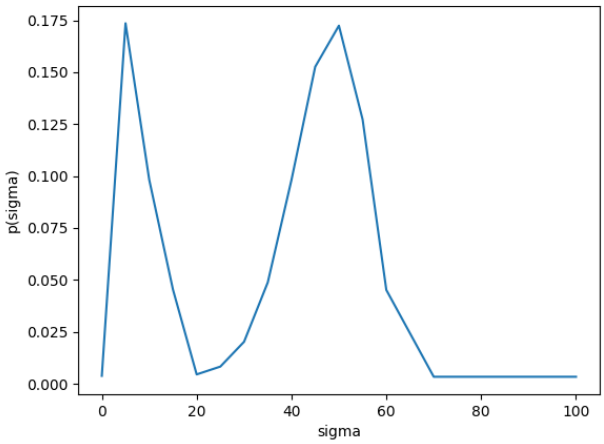


Figure 16: **include a slightly more zoomed out one, not nec the same thing**

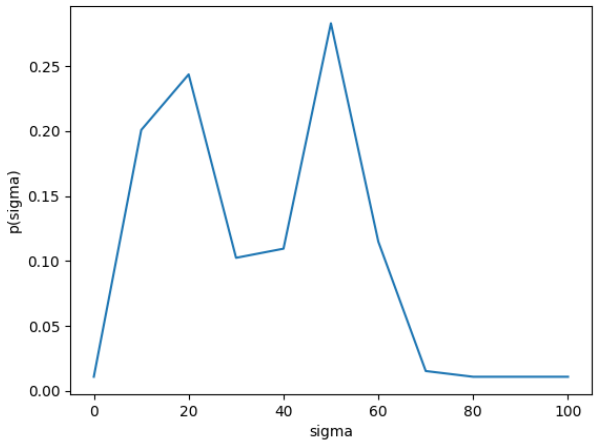
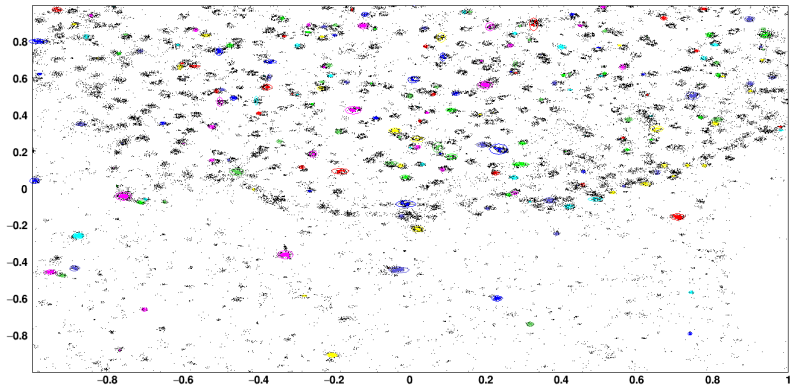
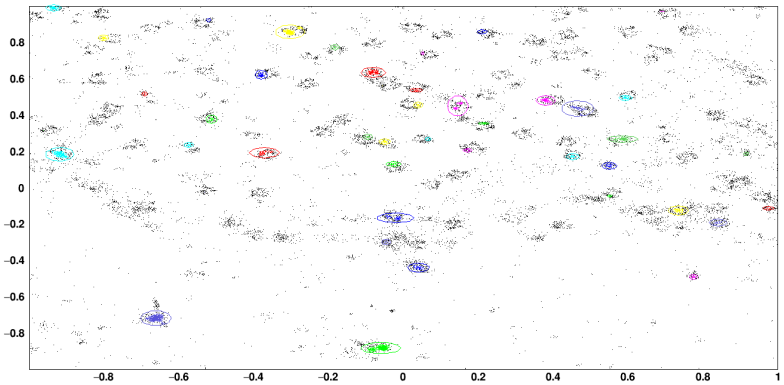


Figure 17: **include a slightly more zoomed out one, not nec the same thing**



(a) **caption goes here**



(b) **caption goes here**.

Figure 18

- The number of points in each cluster
- The proportion of points that belong to the background
- The number of clusters in the ROI
- The maximum cluster radius

The results are summarised in table 19. The efficiency and purity of this method are both high. Each of the datasets contained over 40,000 points, each of which had a target for the percentage of points belonging to clusters set at either 25% or 50%. In every case, the algorithm made an incorrect decision for less than 1% of the members of the simulated dataset. The number of identified clusters was highly accurate too, the error on which was typically less than 5%. For instance, when analysing the seventh dataset, the values we expected to find were as follows[57]:

- Number of points per cluster: 20
- Maximum Radius per cluster: 40nm
- Percentage of points assigned to clusters: 25%

Out of 46064 points viewed by the algorithm, a correct class label (cluster or background) was given in 45917 places, a success rate of more than 99%. Of the ones that were labelled incorrectly, 81 background points, and 66 cluster members, were given the opposite class label. It's worth noting that the model assumes the background points are uniformly distributed across the FOV: given that we have approximately 0.2% of the FOV covered by clusters, we may expect that of the approximately 35,000 background points

[57] change the list: irrelevant info therein

in the dataset, around 70 of them should still appear within an identified cluster. We were able to detect 569 clusters, compared to the target of 584. This is likely attributable to clusters in close proximity being drawn together - this is behaviour that can occur due to the choice of the R parameter, as we shall discuss further[58] in section 8.4.4.

We believe that this demonstrates the algorithm is capable of performing highly accurate clustering on SMLM data. What remains to be seen is how this algorithm copes with a more varied dataset, synthetic or otherwise. One interesting thing to note from [30] is Figure 3(g), which shows the Bayesian method is inaccurate when analysing a CSR background. One suggestion may be to, below a certain posterior probability[59], add some pressure to the algorithm to raise the threshold T, to ensure that points are not erroneously clustered together.

8.4.4 Sensitivity to R and T

Rubin-Delanchy et. al. [30] have discussed the robustness of this algorithm with respect to various parameters. The main contribution of this report is empirical evidence that their procedure can be implemented in a way that allows for high-throughput analysis of SMLM data, and so in the following section we demonstrate how the diagrams of the identified clusters change with respect to the parameters R and T.

8.4.5 Sensitivity to other parameters

8.5 Supposed Amdahl Limit

We present here a brief demonstration that the use of multithreading allows us to maximise the possible gains in runtime. We ran 7 different scans of the nucleus seen in figure 22, the first using a single thread, before each time doubling the number of threads in use to a maximum of 64. Figure 22 shows the results. We see that initial increases in thread counts lead to extremely rapid decreases in runtime: the process of both populating the neighbour lists and scanning across the $R-T$ parameter space halved each time the thread count was increased from 1 to 2, then from 2 to 4 and finally from 4 to 8. Thereafter, the decreases in runtime plateau, as we reach some kind of bottleneck, as the portions of the code which cannot be multithreaded begin to dominate the runtime. Beyond the scope of this project is a thorough investigation into what dominates the runtime of the code when fully parallelised, such as calls to non-standard libraries that interpolate functions and calculate numerical integrals.

[58] please make sure
[59] A low posterior probability would indicate that

	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5	Dataset 6	Dataset 7	Dataset 8	Dataset 9
Total Number of Localisations	43046	45289	44744	43218	43781	44261	46091	44320	44208
True Number of background points	21720	22800	22740	32730	33360	33540	35040	33660	33420
True Number of clusters	1086	1140	1137	1091	1112	1118	584	561	557
Identified Number of clusters	1043	1113	1084	1033	1020	1052	569	542	542
Number of identified background points	21861	22591	22603	32687	32930	33386	34479	33374	33031
Number of misattributed background points	58	149	71	138	218	154	66	51	104
Number of misattributed cluster points	347	70	274	356	251	297	81	145	8

Figure 19: results of the scans

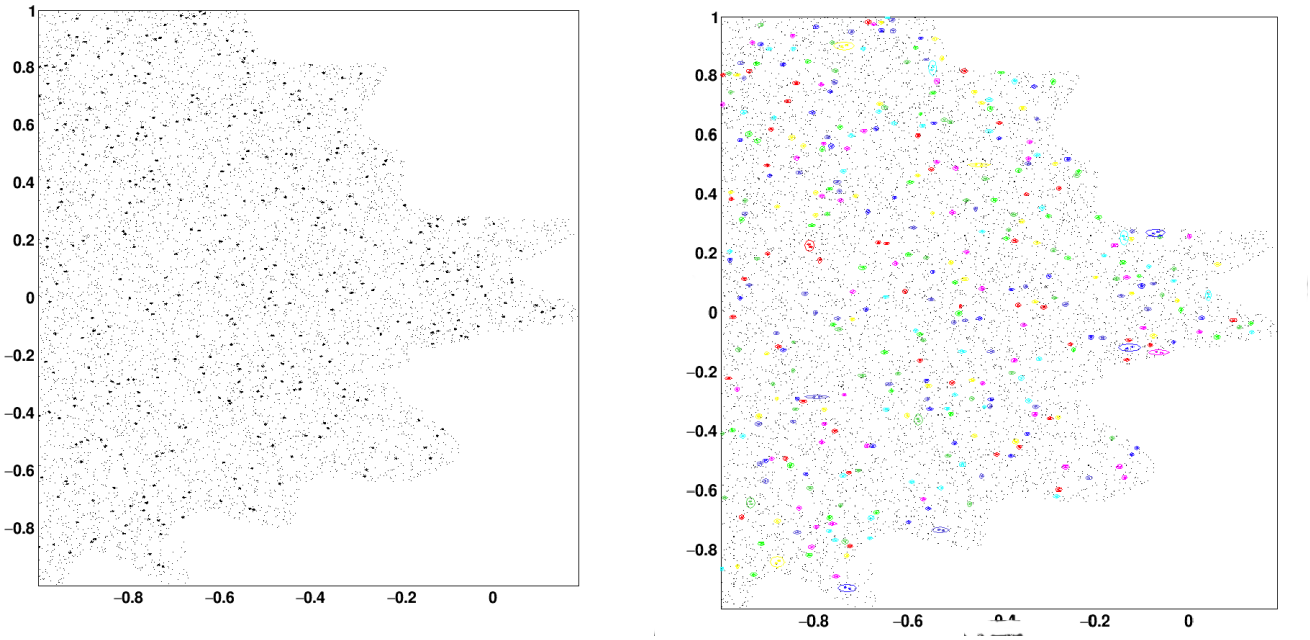


Figure 20

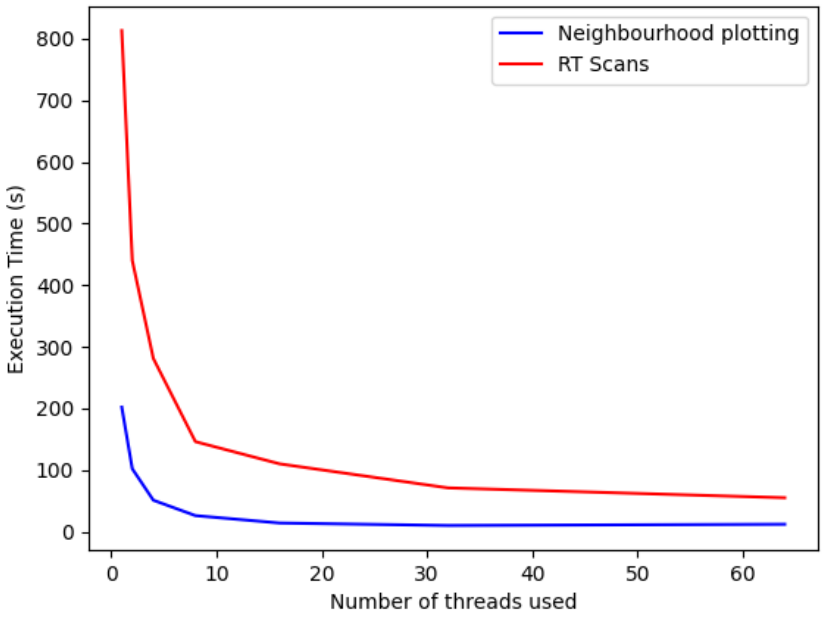


Figure 21: weaad

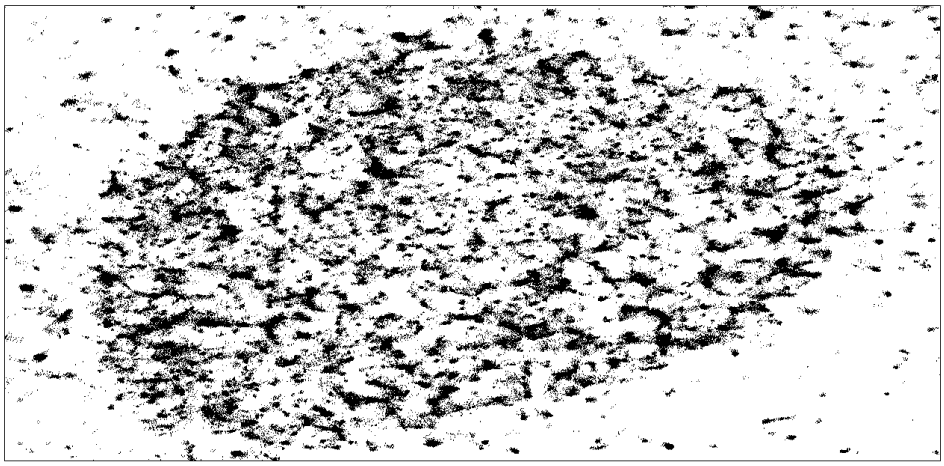


Figure 22: awd

References

[1] Mitko Aleksandrov, Sisi Zlatanova, and David J. Heslop. Voxelisation algorithms and data structures: A review. Sensors, 21(24), 2021.

[2] Mark Bates, Timothy R Blosser, and Xiaowei Zhuang. Short-range spectroscopic ruler based on a single-molecule optical switch. Physical review letters, 94(10):108101, 2005.

[3] Eric Betzig, George H. Patterson, Rachid Sougrat, O. Wolf Lindwasser, Scott Olenych, Juan S. Bonifacino, Michael W. Davidson, Jennifer Lippincott-Schwartz, and Harald F. Hess. Imaging intracellular fluorescent proteins at nanometer resolution. Science, 313(5793):1642–1645, 2006.

[4] CA Combs and H Shroff. Fluorescence microscopy: A concise guide to current imaging methods. 2017.

[5] Ruggero Cortini, Maria Barbi, Bertrand R. Caré, Christophe Lavelle, Annick Lesne, Julien Mozziconacci, and Jean-Marc Victor. The physics of epigenetics. Rev. Mod. Phys., 88:025002, Apr 2016.

[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd, volume 96, pages 226–231, 1996.

[7] Thomas S. Ferguson. A Bayesian Analysis of Some Nonparametric Problems. The Annals of Statistics, 1(2):209 – 230, 1973.

[8] William A Flavahan, Elizabeth Gaskell, and Bradley E Bernstein. Epigenetic plasticity and the hallmarks of cancer. Science, 357(6348):eaal2380, 2017.

[9] P.M.W. French. Overview of fluorescence imaging for biophotonics. 181:1–27, 01 2014.

[10] Arthur Getis and Janet Franklin. Second-Order Neighborhood Analysis of Mapped Point Patterns, volume 68, pages 93–100. 08 2010.

[11] Peter Green and Sylvia Richardson. Modelling heterogeneity with and without the dirichlet process. Scandinavian Journal of Statistics, 28, 07 2000.

[12] Juliette Griffie, Ruby Peters, Garth Burn, Dylan Owen, and David Williamson. Dynamic bayesian cluster analysis of live-cell single molecule localization microscopy datasets. Small Methods, 2, 06 2018.

[13] Juliette Griffié, Michael Shannon, Claire L Bromley, Lies Boelen, Garth L Burn, David J Williamson, Nicholas A Heard, Andrew P Cope, Dylan M Owen, and Patrick Rubin-Delanchy. A bayesian cluster analysis method for single-molecule localization microscopy data. Nature Protocols, 11(12):2499–2514, 2016.

[14] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. ACM computing surveys (CSUR), 31(3):264–323, 1999.

[15] Ismail M. Khater, Ivan Robert Nabi, and Ghassan Hamarneh. A review of super-resolution single-molecule localization microscopy cluster analysis and quantification methods. Patterns, 1(3), Jun 2020.

[16] Maria A Kiskowski, John F Hancock, and Anne K Kenworthy. On the use of ripley's k-function and its derivatives to analyze domain size. Biophysical journal, 97(4):1095–1103, 2009.

[17] Mickaël Lelek, Melina T. Gyparaki, Gerti Beliu, Florian Schueder, Juliette Griffié, Suliana Manley, Ralf Jungmann, Markus Sauer, Melike Lakadamyali, and Christophe Zimmer. Single-molecule localization microscopy. Nature Reviews Methods Primers, 1(1):39, Jun 2021.

[18] Daniel Lingwood, Jonas Ries, Petra Schwille, and Kai Simons. Plasma membranes are poised for activation of raft phase coalescence at physiological temperature. Proceedings of the National Academy of Sciences of the United States of America, 105:10005–10, 08 2008.

[19] Martin Ovesny, Pavel Křížek, Josef Borkovec, Zdenek Svindrych, and Guy Hagen. Thunderstorm: a comprehensive imagej plug-in for palm and storm data analysis and super-resolution imaging. Bioinformatics (Oxford, England), 30, 04 2014.

[20] Dylan Owen, Astrid Magenau, David Williamson, and Katharina Gaus. The lipid raft hypothesis revisited - new insights on raft composition and function from super-resolution fluorescence microscopy. BioEssays : news and reviews in molecular, cellular and developmental biology, 34:739–47, 09 2012.

[21] Dylan M. Owen and Katharina Gaus. Imaging lipid domains in cell membranes: the advent of super-resolution fluorescence microscopy. Frontiers in plant science, 4(N/A):N/A, December 2013.

[22] Sophie Pageon, Philip Nicovich, Mahdie Mollazade, Thibault Tabarin, and Katharina Gaus. Clus-doc: A combined cluster detection and colocalization analysis for single-molecule localization microscopy data. Molecular Biology of the Cell, 27, 08 2016.

[23] Tingwei Quan, Shaoqun Zeng, and Zhen-Li Huang. Localization capability and limitation of electron-multiplying charge-coupled, scientific complementary metal-oxide semiconductor, and charge-coupled devices for superresolution imaging. Journal of biomedical optics, 15:066005, 11 2010.

[24] Maria Aurelia Ricci, Carlo Manzo, María Filomena García-Parajo, Melike Lakadamyali, and Maria Pia Cosma. Chromatin fibers are formed by heterogeneous groups of nucleosomes in vivo. Cell, 160(6):1145–1158, 2015.

[25] Brian D Ripley. Modelling spatial patterns. Journal of the Royal Statistical Society: Series B (Methodological), 39(2):172–192, 1977.

[26] Patrick Rubin-Delanchy, Garth L. Burn, Juliette Griffié, David J. Williamson, Nicholas A. Heard, Andrew P. Cope, and Dylan M. Owen. Bayesian cluster identification in single-molecule localization microscopy data. Nature Methods, 12(11):1072–1076, Nov 2015.

[27] Michael J Rust, Mark Bates, and Xiaowei Zhuang. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (storm). Nature methods, 3(10):793–796, 2006.

[28] Robert A. Schowengerdt. Chapter 3 - sensor models. In Robert A. Schowengerdt, editor, Remote Sensing (Third Edition), pages 75–XIV. Academic Press, Burlington, third edition edition, 2007.

[29] Jervis Vermal Thevathasan, Maurice Kahnwald, Konstanty Cieśliński, Philipp Hoess, Sudheer Kumar Peneti, Manuel Reitberger, Daniel Heid, Krishna Chaitanya Kasuba, Sarah Janice Hoerner, Yiming Li, Yu-Le Wu, Markus Mund, Ulf Matti, Pedro Matos Pereira, Ricardo Henriques, Bianca Nijmeijer, Moritz Kueblbeck, Vilma Jimenez Sabinina, Jan Ellenberg, and Jonas Ries. Nuclear pores as versatile reference standards for quantitative superresolution microscopy. Nature Methods, 16(10):1045–1053, Oct 2019.

[30] David J Williamson, Garth L Burn, Sabrina Simoncelli, Juliette Griffié, Ruby Peters, Daniel M Davis, and Dylan M Owen. Machine learning for cluster analysis of localization microscopy data. Nature communications, 11(1):1–10, 2020.

[31] Jianquan Xu and Yang Liu. A guide to visualizing the spatial epigenome with super-resolution microscopy. The FEBS journal, 286(16):3095–3109, 2019.

[32] Jianquan Xu, Hongqiang Ma, Jingyi Jin, Shikhar Uttam, Rao Fu, Yi Huang, and Yang Liu. Super-resolution imaging of higher-order chromatin structures at different epigenomic states in single mammalian cells. Cell Reports, 24:873–882, 07 2018.

[33] Jianquan Xu, Hongqiang Ma, and Yang Liu. Stochastic Optical Reconstruction Microscopy (STORM), volume 2017, pages 12.46.1–12.46.27. 07 2017.

[34] Jianquan Xu, Xuejiao Sun, Kwangho Kim, Rhonda M Brand, Douglas Hartman, Hongqiang Ma, Randall E Brand, Mingfeng Bai, and Yang Liu. Ultrastructural visualization of chromatin in cancer pathogenesis using a simple small-molecule fluorescent probe. Science advances, 8(9):eabm8293, 2022.

[35] Qiang Zheng and Jian Sun. Effective point cloud analysis using multi-scale features. Sensors, 21(16), 2021.

A The Integral

todo:

- move this into an appendix

In equation 5, we stated that for a given cluster containing n localisations, the probability of observing this set of localisations, given cluster parameters $\mu = (\mu_x, \mu_y)$ and σ was given as a product of probabilities. We begin our **discussion** with the calculation of each of these n terms. We **stated further that** we can write this probability as: **TODO - put the separate fraction as a square root for clarity!**

$$p(V_i \,|\, \mu, \sigma) = \frac{1}{2\pi\overline{\sigma}_i^2} \exp\left(\frac{-(x_i - \mu_x)^2}{2\overline{\sigma}_i^2} - \frac{(y_i - \mu_y)^2}{2\overline{\sigma}_i^2}\right),$$

where we have defined $\overline{\sigma}_i = \sigma^2 + s_i^2$, the sum of our two standard deviations[60]. For each cluster identified, we aim to compute the integral within. Let us now expand this integral to reveal some quantities which can allow for efficient calculation of these probability densities. Due to the separability[61] of **the gaussian**, we can consider integrating over each spatial dimension of the Gaussian independently. Without loss of generality, we consider the x dimension:

[60] ensure this is properly defined somewhere
[61] justify this

$$\begin{aligned}
&\int_{x-}^{x+} \prod_{i=1}^{N} \left(\frac{1}{\sqrt{2\pi\overline{\sigma}_i^2}} \exp\left(\frac{-(x_i-\mu)^2}{2\overline{\sigma}_i^2} \right) \right) d\mu \\
&\quad = \int_{x-}^{x+} \left(\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\overline{\sigma}_i^2}} \right) \left(\prod_{i=1}^{N} \exp\left(\frac{-(x_i-\mu)^2}{2\overline{\sigma}_i^2} \right) \right) d\mu \\
&\quad = \left(\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\overline{\sigma}_i^2}} \right) \int_{x-}^{x+} \prod_{i=1}^{N} \exp\left(\frac{-(x_i-\mu)^2}{2\overline{\sigma}_i^2} \right) d\mu \\
&\quad = \left(\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\overline{\sigma}_i^2}} \right) \int_{x-}^{x+} \exp\left(-\sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\overline{\sigma}_i^2} \right) d\mu. \qquad (13)
\end{aligned}$$

We next examine the sum inside the integral:

$$\begin{aligned}
\sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\overline{\sigma}_i^2} &= \frac{1}{2} \left(\mu^2 \sum_{i=1}^{N} \frac{1}{\overline{\sigma}_i^2} - 2\mu \sum_{i=1}^{N} \frac{x_i}{\overline{\sigma}_i^2} + \sum_{i=1}^{N} \frac{x_i^2}{\overline{\sigma}_i^2} \right) \\
&= \frac{1}{2} \left(A\mu^2 - 2B\mu + C \right) \\
&= \frac{1}{2} \left(A(\mu - D)^2 + E \right).
\end{aligned}$$

Above, we have defined:

$$A = \sum_{i=1}^{N} \frac{1}{\overline{\sigma}_i^2} \qquad (14)$$

$$B_d = \sum_{i=1}^{N} \frac{x_{d,i}}{\overline{\sigma}_i^2} = AD_d \qquad (15)$$

$$C_d = \sum_{i=1}^{N} \frac{x_{d,i}^2}{\overline{\sigma}_i^2} = AD_d^2 + E_d = \frac{B_d^2}{A} + E_d \qquad (16)$$

$$D_d = \frac{B_d}{A} \qquad (17)$$

$$E_d = C_d - AD_d^2 = C_d - \frac{B_d^2}{A}. \qquad (18)$$

Based on our earlier discussion of the necessary calculation that needs to be performed, we can demonstrate how the above variables fit in to the general picture. Taking the integral from equation 13, and removing the constant factor $\frac{1}{\sqrt{2\pi\overline{\sigma}_i^2}}$, we can state that:

$$\begin{aligned}
&\int_{x-}^{x+} \prod_{i=1}^{N} \left(\exp\left(\frac{-(x_i-\mu)^2}{2\overline{\sigma}_i^2} \right) \right) d\mu \\
&\int_{x^-}^{x^+} \exp\left(-\sum_{i=1}^{N} \frac{(x_i-\mu)^2}{2\overline{\sigma}_i^2} \right) d\mu \\
&\quad = \int_{x^-}^{x^+} \exp\left(-\frac{1}{2} \left(A(\mu - D)^2 + E \right) \right) d\mu
\end{aligned}$$

$$
\begin{aligned}
&= \int_{x^-}^{x^+} \exp\left(-\frac{E}{2}\right) \exp\left(-\frac{1}{2}\left(A(\mu - D)^2\right)\right) d\mu \\
&= \exp\left(-\frac{E}{2}\right) \int_{x^-}^{x^+} \exp\left(-\frac{1}{2}\left(A(\mu - D)^2\right)\right) d\mu \\
&= \exp\left(-\frac{E}{2}\right) \int_{x^-}^{x^+} \exp\left(\frac{-(\mu - D)^2}{2\frac{1}{A}}\right) d\mu \\
&= \exp\left(-\frac{E}{2}\right) \int_{x^-}^{x^+} \frac{\sqrt{2\pi\frac{1}{A}}}{\sqrt{2\pi\frac{1}{A}}} \exp\left(\frac{-(\mu - D)^2}{2\frac{1}{A}}\right) d\mu \\
&= \sqrt{\frac{2\pi}{A}} \cdot \exp\left(-\frac{E}{2}\right) \int_{x^-}^{x^+} \frac{1}{\sqrt{2\pi\frac{1}{A}}} \exp\left(\frac{-(\mu - D)^2}{2\frac{1}{A}}\right) d\mu \\
&= \sqrt{\frac{2\pi}{A}} \cdot \exp\left(-\frac{E}{2}\right) \left[\phi\left(\frac{x^+ - D}{\sqrt{\frac{1}{A}}}\right) - \phi\left(\frac{x^- - D}{\sqrt{\frac{1}{A}}}\right)\right] \\
&= \sqrt{\frac{2\pi}{A}} \cdot \exp\left(-\frac{E}{2}\right) \left[\phi\left(\sqrt{A}(x^+ - D)\right) - \phi\left(\sqrt{A}(x^- - D)\right)\right] \\
&= \sqrt{\frac{2\pi}{A}} \cdot \exp\left(-\frac{E}{2}\right) \cdot G.
\end{aligned}
$$

Here we have defined:

$$
G_d = \phi\left(\sqrt{A}(x_d^+ - D_d)\right) - \phi\left(\sqrt{A}(x_d^- - D_d)\right)
$$

with:

$$
\Phi(x) = \int_{-\infty}^{x} (2\pi)^{-1/2} \exp(-y^2/2) dy
$$

being the standard Gaussian cumulative distribution function. We can then continue - the probability of observing a set of localisations $\nu = [V_1, V_2, \ldots, V_N]$ requires taking the product of $p(V_i|\mu, \sigma)$ for each V_i in ν, before integrating over the μ and σ parameters.

$$
\begin{aligned}
p(\nu) &= \int_\sigma p(\sigma) \int_\mu p(\mu) \prod_{i=1}^{N} p(V_i \,|\, \mu, \sigma)\, d\mu\, d\sigma \\
&= \int_\sigma p(\sigma) \prod_d \left[\frac{1}{(x_d^+ - x_d^-)} \left(\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\overline{\sigma}_i^2}}\right) \sqrt{\frac{2\pi}{A}} \exp\left(-\frac{E_d}{2}\right) G_d\right] d\sigma \\
&= (2\pi)^{(1-N)\cdot\tau/2} \cdot V^{-1} \int_\sigma p(\sigma) \cdot \left(\prod_{i=1}^{N} \frac{1}{\overline{\sigma}_i^2}\right)^{\tau/2} \cdot A^{-\tau/2} \cdot \prod_d \left[\exp\left(-\frac{E_d}{2}\right)\right] \cdot \prod_d G_d\, d\sigma \\
&= (2\pi)^{(1-N)\cdot\tau/2} \cdot V^{-1} \int_\sigma p(\sigma) \cdot F^{\tau/2} \cdot A^{-\tau/2} \cdot \exp\left(-\frac{1}{2}\sum_d E_d\right) \cdot \prod_d G_d\, d\sigma \\
&= (2\pi)^{(1-N)\cdot\tau/2} \cdot V^{-1} \int_\sigma p(\sigma) \cdot \left(\frac{F}{A}\right)^{\tau/2} \cdot \exp\left(-\frac{E}{2}\right) \cdot \prod_d G_d\, d\sigma
\end{aligned} \tag{19}
$$

Where:

$$
\begin{aligned}
E &= \sum_d E_d = \sum_d (C_d - B_d D_d) \\
&= \sum_d C_d - \sum_d B_d D_d \\
&= C - \sum_d B_d D_d \\
C &= \sum_d C_d = \sum_d \sum_{i=1}^{N} \frac{x_{d,i}^2}{\overline{\sigma}_i^2} \\
&= \sum_{i=1}^{N} \frac{\sum_d x_{d,i}^2}{\overline{\sigma}_i^2} = \sum_{i=1}^{N} \frac{r_i^2}{\overline{\sigma}_i^2} \qquad (20) \\
F &= \prod_{i=1}^{N} \frac{1}{\overline{\sigma}_i^2} \qquad (21)
\end{aligned}
$$

We can now choose these four variables to track, and we shall rewrite some important equations in terms of them. In later sections, we shall demonstrate how tracking these variables during the cluster proposal process allows for (at least one) fewer passes of the whole set of localisations. Per the suggestion of [26], the large difference in scales means that this computation must be performed on the log scale. In particular, the scale differences can be so large that, as we shall see in later sections[62], computation must be performed carefully to ensure that all variables remain within the confines of the type `double`. We find that the integrand in equation 19 is numerically unstable, and so calculation of these quantities is performed on the log scale, and then exponentiated just before the numerical integration step.

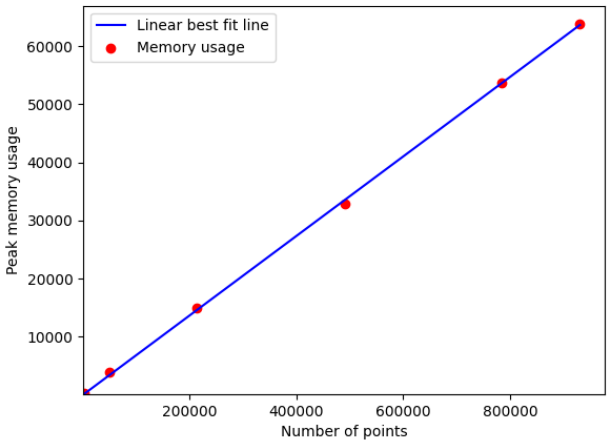
Turning our attention back to the prior probability we noted in equation 3, we state here its log-scale expression, so that we can refer back to it later.

$$
\begin{aligned}
ln\left(p\left(l\right)\right) &= n_B \cdot ln\left(p_B\right) + \left(N - n_B\right) \cdot ln\left(1 - p_B\right) + m \cdot ln\left(\alpha\right) \\
&\quad + ln\left(\Gamma\left(\alpha\right)\right) + ln\left(\prod_{k=1}^{m} \Gamma\left(n_k\right)\right) - ln\left(\Gamma\left(\alpha + N - n_B\right)\right) \\
&= n_B \cdot ln\left(p_B\right) + \left(N - n_B\right) \cdot ln\left(1 - p_B\right) + m \cdot ln\left(\alpha\right) \\
&\quad + ln\left(\Gamma\left(\alpha\right)\right) + \sum_{k=1}^{m} ln\left(\Gamma\left(n_k\right)\right) - ln\left(\Gamma\left(\alpha + N - n_B\right)\right) \\
&= n_B \cdot ln\left(p_B\right) + \left(N - n_B\right) \cdot ln\left(1 - p_B\right) + m \cdot ln\left(\alpha\right) \\
&\quad + \overline{\Gamma}\left(\alpha\right) + \sum_{k=1}^{m} \overline{\Gamma}\left(n_k\right) - \overline{\Gamma}\left(\alpha + N - n_B\right)
\end{aligned}
$$

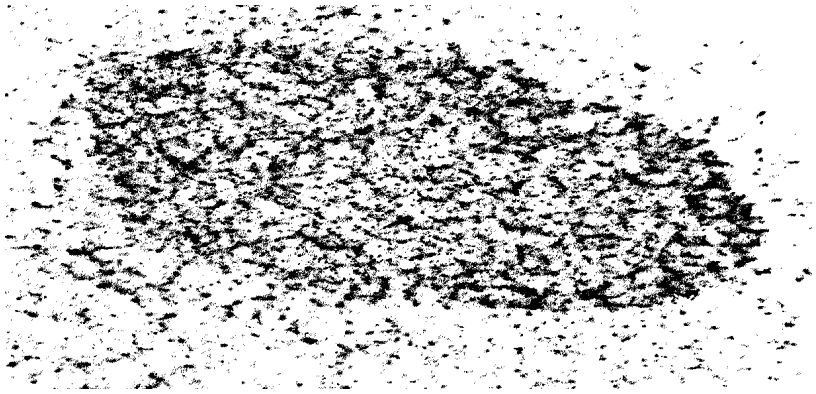
Where $\overline{\Gamma}$ is the log of the Gamma function, the analytical continuation of the factorial function.

B Figures

[62] REF!



(a) **caption goes here** molecule.



(b) **cpation goes here**

Figure 23