Quantifying the Golgi

Gert-Jan Both

Supervised by:
P. Sens
C. Storm

Technical university of Eindhoven
January-November 2018

Abstract

Title: Quantifying the Golgi

Abstract: The Golgi apparatus is a key component in intracellular trafficking, maturing and directing proteins essential to the cell. Despite years of research, a model coupling Golgi size and function to the cells' transport properties is lacking. In this thesis we develop such a model, describing the Golgi as an active droplet. New experimental data sheds more insight in the spatial organization of the trafficking and I have also devised two new methods relying on image gradients and neural networks to analyze this data and confront it with our model.

Contents

1

Introduction

The cell uses thousands of proteins and lipids to function and many of these are produced in the Endoplasmic Reticulum (ER), an organelle found in eukariotic cells. Upon exiting the ER, these proteins and lipids are then transported throughout the cell in a process known as intracellular transport. Key component in this intracellular trafficking is the Golgi apparatus, an organelle responsible for biochemically maturing proteins and directing them to the right location. Intense research over the last years has identified key players,[1],[2] but an integrated model coupling Golgi size and function to the intracellular transport is lacking[3],[4]. In this thesis, we seek to propose such a model.

1.1 The secretory pathway: biology 101 for physicists

$$f(x) = a$$

Proteins produced in the ER exit the organelle at specific locations referred to as ER Exit Sites - ERES. At these sites, cargo is packaged into a lipid bilayer and this package, known as a vesicle, buds off into the cytoplasm[5]. ERES are located throughout the cell and thus the vesicles need to be transported to their destination: the Golgi apparatus. In general, we can recognize two different trafficking modes: diffusive and directive[6]. In the directive mode, molecular motors pull vesicles along microtubules by hydrolysing ATP. Microtubules (MTs) are long tubular polymers spread throughout the cell and form a network which acts as the backbone for intracellular transport. They are organised around objects known as MicroTubular Organisation Centers (MTOCs). The primary MTOC is the centrosome,

an organelle located next to the nucleus, but strong evidence exists that the Golgi apparatus acts a MTOC too[7],[8].

Microtubules are polarized and have two distinct ends, indicated as the (+) and (-). Different molecular motors are utilized for transport towards each end, with dynein being (-)-directed and kinesin (+)-directed[9]. Vesicles are often attached to multiple motors of both types, binding and unbinding constantly, making this active transport a stochastic process which can, for example, be described by a tug-of-war model[10]. Furthermore, cargo can also completely detach from all molecular motors. The vesicle will then move through the cytoplasm in a diffusive way, until it reattaches to a microtubule. Note that diffusive mode is a deceptively simple name, as the cytoplasm is not a simple fluid; it is packed with other cellular components, giving rise to effects such as anomolous diffusion or crowding.

The intracellular trafficking transports the vesicles towards the Golgi, where the cargo undergoes biochemical maturation and is sorted before being sent to their destination. The Golgi thus acts as a sort of post-office of the cell, receiving cargo, repackaging and sending it to the right destination[11]. Although the function of the Golgi is similar for different cell types, its appearance is strongly dependent on it. In plants for example, the Golgi is distributed throughout the cell in separate but fully functional subunits known as stacks[12] , whereas in mammals all these stacks are localized close to the nucleus in a single organelle known as the *Golgi Ribbon*[7]. A stack consists of a number of stacked compartments of a disk-like shape called *cisternae*. These are membrane enclosed objects containing the enzymes responsible for biochemically altering proteins, a process generally referred to as maturing. Proteins move through the Golgi in a particular direction and the Golgi thus has distinct entry and exit faces. These are known respectively as the cis and trans face, with the cisternae being labeled analogously. The cisternae in the middle of the stack are referred to as medial compartments.

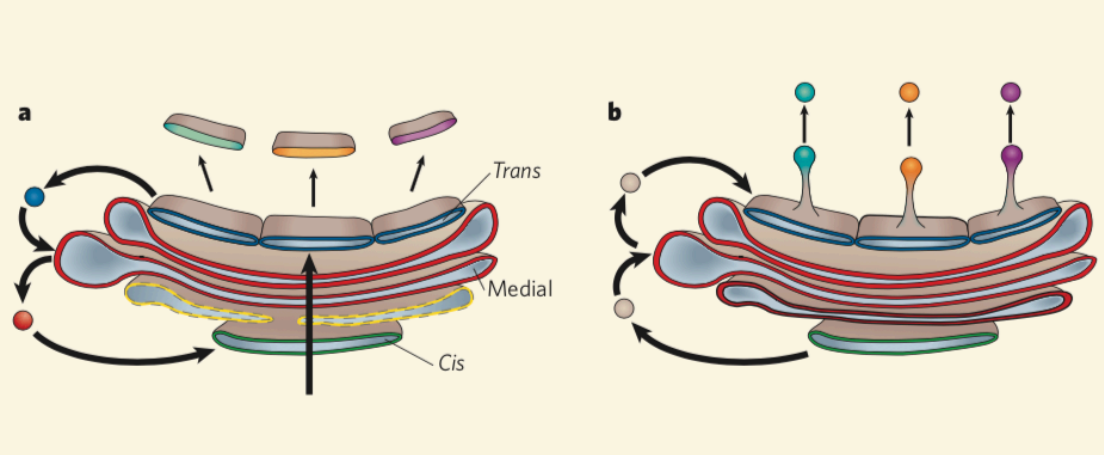


Figure 1.1: Left panel: In the cisternal maturation model, compartments mature as a whole and thus change identity. **Right panel**: In the vesicle transport model, compartments are static objects and cargo is being transported from compartment to compartment by vesicles. Image taken from 13.

At the cis-face vesicles fuse with the golgi and release their cargo into a compartment, while their lipid bilayer becomes part of the compartment membrane. Exactly how maturation then happens is hotly debated[14],[13]. The two main competing explanations are the *cisternal maturation* and *vesicular transport* models. In figure fig. 1.1 we show the structure of the Golgi and and a schematic view of each model. In the cisternal maturation model (left panel of fig. 1.1), the compartments mature as a whole, changing identity from cis to medial and finally to trans. Trans compartments are recycled into cis compartments by retrograde vesicular transport. In the vesicular transport model (see right panel of fig. 1.1), the vesicles move in the opposite direction. Rather than constantly changing identity, in this model cisternae are static entities with a defined task and cargo is moved from one compartment to the next by vesicles. The debate could thus be settled by analysing the direction of the vesicles, but so far this has proven elusive.

At the trans-face, the cargo is encapsulated again in a lipid bilayer and is transported to its destination, similar to pre-Golgi intracellular transport.

1.1.1 Quantitative models of the Golgi

The Golgi has been intensively studied by biologists for many years, but very few attempts at quantification appear to have been made: our research only turned up a single attempt by Hirschberg et al[15], where the authors present a model for the trafficking of VSVG virus from the ER to the plasma membrane. The secretory pathway is modeled by dividing it

into populations connected by a first order rate equation, i.e. $d\varphi_2/dt = k_{1\to 2}\varphi_1$. Assuming no flowback (i.e. $k_{i+1\to i} = 0$) and a population for the ER, Golgi and Plasma Membrane, they find that such a model is sufficient to describe their experimental data, as shown in figure #fig:ratemodel.

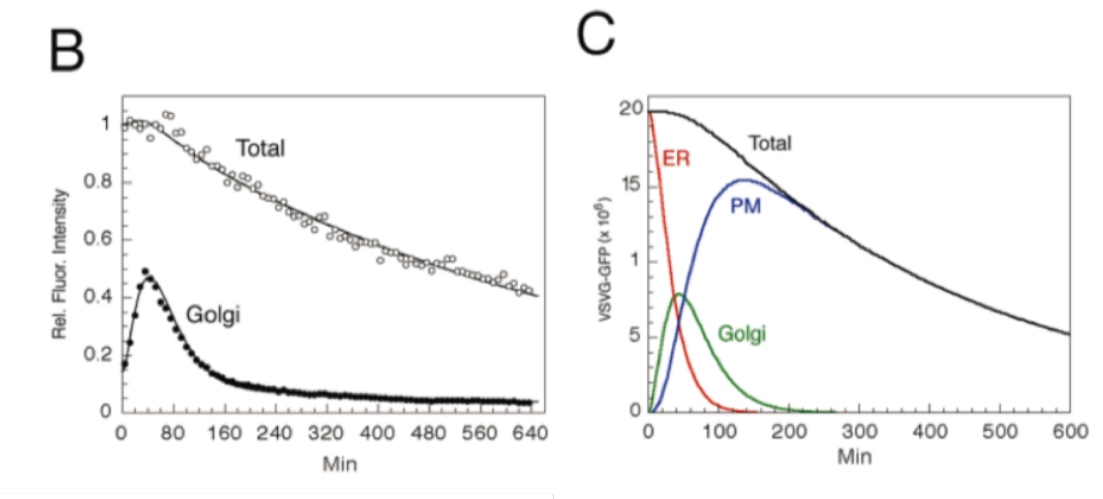


Figure 1.2: Left panel: First order rate model fitted to experimental data by 15 **Right panel**: Inferred concentration in ER, Golgi and PM using the fitted parameters from the left panel and their model. Image reprinted from 15.

We reprint their main result in figure fig. 1.2. Although this model describes the experimental data, it is a phenomological model and reduces the entiry system to a few rate parameters $k_{i\to i+1}$. These rate parameters are not coupled to any of the underlying processes and hence this model doesn't offer any insight into the system. Furthermore, the model lacks any spatial dependence of the concentration.

1.2 This thesis

In this thesis we propose a model which couples intracellular transport to the Golgi. We hypothesize that we can describe the Golgi as an *active, phase separated droplet* and the intracellular transport with an advection-diffusion equation. We will confront our model with experimental data. The group of Frank Perez at Institut Curie has developed a new technique called RUSH[16], which allows us to study the intracellular transport from the ER to the Golgi and beyond using fluorescence microscopy. In the next sections, we justify the description of the Golgi as an active phase-separated droplet and how we intend to perform the data analysis of the experimental data.

1.2.1 Golgi as an active phase-separated droplet

Many biological processes and reactions require a high concentration of some protein or lipid to occur. This can be achieved by physically separating proteins inside a membrane (consider the lysosome), but the cell contains several membraneless organelles. These organelles thus require a different means of reaching high concentrations and the prime candidate is liquid-liquid phase separation. In this process a mixture of two different liquids A and B separates into two phases, one rich in A and one rich in B, due to the interactions between them. Phase separation can thus produce membraneless domains with a high concentration. It has been proposed as a model for early protocells[17] and is able to correctly describe several phenomena such as P-granules[18] and centrosome growth[19].

We hypothesize we can use a similar description for the Golgi, as its biogenesis contains strong clues which point towards phase separation. First and foremost, the Golgi is able to form *de novo*, meaning that in cells from which the Golgi has been removed, it will reappear without any specific action. Ronchi et al[8] studied this in detail and found three phases of growth. In the first phase, vesicles are released from the ER, but no larger structures are formed and the vesicles disappear either due to fusion with the ER or degradation. In the second phase larger stack-like structures are formed, while in the third phase all these structures are clustered in a single location; the Golgi Ribbon is formed. Phase two has the markings of a concentration-dependent phase separation. Once a critical concentration is reached, the mixed state becomes unstable and the mixture phase-separates into a dilute and dense phase. The dense phase would then correspond to a stack.

One crucial point we've neglected so far is that the Golgi isn't a membraneless organelle and neither is the cargo. Hence, a phase-separation description should be inappropriate. Using a description in which we consider the vesicles to be the phase-separating liquid, we think we can use a phase-separation description however. The dense phase would then correspond to the Golgi, while the dilute phase would correspond to the cytoplasm with the vesicles. Such models predict droplets growing to infinity and research has shown that in reality the Golgi has a finite size. By accounting for the maturation of the cargo, making our droplet *active* in the process, finite radii can be obtained. We thus model the Golgi as an *active phase-separated droplet*.

1.2.2 Biological image analysis

Image analysis is a lively and ongoing subject in cell biology, with many new methods being developed constantly, especially with quantization in mind[20],[21],[22],[23]. Techniques for quantifying intracellular transport roughly fall into two categories: single particle tracking (SPT)

or correlation spectroscopy. SPT tracks fluorescent proteins or beads moving through the cell on a frame-to-frame basis, so that each particle's trajectory can be reconstructed. These trajectories can then be analyzed to obtain information about the transport. The fluorescent movies obtained from the RUSH experiments are not clear enough to accurately localize the vesicles, so that SPT can not be used to analyze the transport. Methods based on correlation spectroscopy[24],[25],[26] rely on a general relationship between the fluctuations and the underlying density of particles and transport properties. These techniques are most suitable for a stationary signal, whereas the RUSH experiments are highly dynamical. Thus, none of the techniques we found are directly applicable to the RUSH data.

As stated, we hypothesize that we can describe the intracellular process by an advection-diffusion equation and we wish to confront this with the RUSH data. The question we're thus asking is a rather general one: how do we fit some spatiotemporal (nD+1) data to a model? More specifically, since a model is most often presented in the form of a partial differential equation (i.e. $df/dt = \alpha df/dx + \beta d^2f/dx^2 + ...$), for what parameters $\alpha, \beta...$ is the temporal evolution of a given dataset best described? We have developed and evaluated two different methods. Our first method approaches the problem rather directly by calculating spatial and temporal derivatives directly from the data using a technique known as image gradients. Our second method is based on a recently developed technique based on neural networks[27]. We will show that by encoding physics into the neural network, we are not only able to infer the optimal parameters, but even an optimal parameter *field*.

1.2.3 Structure and main questions

The rest of this thesis is divided into two parts. In the first part we show the two model fitting methods we have developed and apply them to the RUSH experimental data. The second part discusses the model we have developed for the Golgi. In a chapter-by-chapter breakdown, we have the following:

- Part I - Model fitting and data analysis
  - Chapter 2 introduces the framework we have developed for model fitting spatiotemporal data using image gradients.
  - Chapter 3 applies the method developed in chapter 2 to experimental data.
  - Chapter 4 shows an alternative method for model fitting based on neural networks.

- Part II - Golgi as an active phase-separated droplet
  - Chapter 5 introduces the Cahn-Hilliard equation, which describes phase separation, an approximation of it known as effective droplet theory and develops our model.
  - Chapter 6 contains the predictions the model developed in chapter 5 and investigates the biological implications.

- Chapter 7 is the concluding chapter and summarizes all the findings from the previous chapters.

2

Model fitting

In this chapter we introduce the method we have developed for fitting a model in the form of a PDE to spatiotemporal data. We start with a section describing the general idea and subsequent sections elaborate on each step. The method principaly works for any type of data and model, but was developed originally to analyze data from the RUSH experiments. We have chosen to illustrate the effects of each step with RUSH experimental data instead of synthetic data.

2.1 The concept

Assume we have acces to experimental data of some process $f(x, t)$. Parallely, we have also developed a model describing this process, but it is the form of a PDE:

$$\partial_t f(x,t) = \lambda_1 \nabla^2 f(x,t) + \lambda_2 \nabla f(x,t) + \lambda_3 f(x,t) + \lambda_4 \tag{2.1}$$

We now wish to investigate if this model fits the data $f(x, t)$ and find the optimal value of coefficients λ_i. To do so, we consider each spatial term in $f(x, t)$ in eq. 4.9 as some variable x_i but $\partial_t f$ as y, so that we can rewrite it as:

$$y = \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 + \lambda_4$$

If we thus can find the variables x_i and y, we can perform some fitting procedure such as least squares to obtain the coefficients λ_i. In other words, if we can calculate the spatial and

temporal derivatives of our data, we can fit the model. Although the concept seems trivial, its implementation is all but. Data is rarely noiseless and obtaining accurate derivatives from noisy data is notoriously hard, but forms the heart of our method. It's also possible to have distinct models in different areas of the data, so that we need to segment the data. Furthermore, the coefficients λ_i might not be constant but could be space- and time- dependent. The process of fitting the data thus has several steps:

1. Denoising and smoothing
2. Calculating derivatives
3. Segmenting
4. Fitting

In the next sections, we describe each step separately. Note that the method we present here has been developed empirically: there's no theoretical background as to why this particular combination should work optimally. Instead, it's been developed while analyzing the data, adapting each step on the go. However, we have found several parallel to another method from machine vision known as optical flow[28].

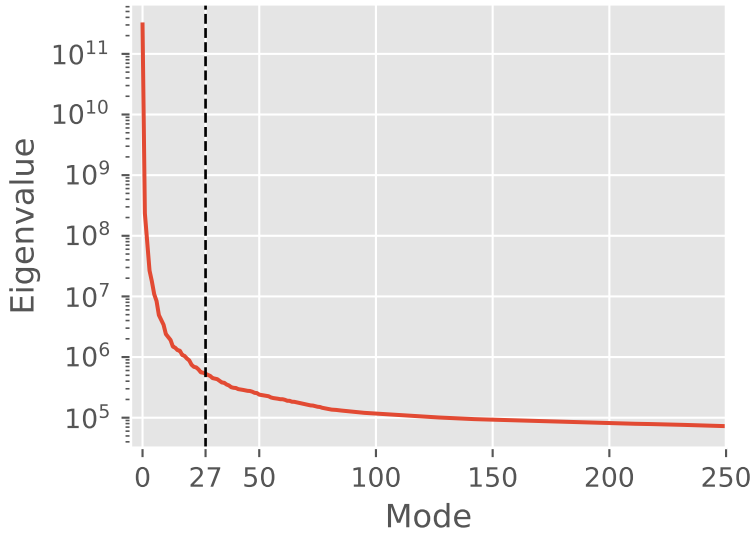
2.2 Step 1 - Smoothing and denoising

The first step in our pipeline is to denoise and smooth the data. The smoothing is necessary for accurately calculating the derivatives. Denoising still is a very active area of research (especially in life sciences) and dozens of different methods exist[31]. For example, one could Fourier transform the signal and use a high pass filter, but this would also get rid of small and sharp features. After evaluating several methods, we have settled on the so-called 'WavInPOD' method, introduced by 32 in 2016. In 33 they show that this methods outperforms several other advanced methods. WavInPOD combines Proper Orthogonal Decomposition (POD) with Wavelet filtering (Wav). Both subjects are vast (especially Wavelet transform) and we're only interested in the result of the technique, so we only present a short introduction here, adapted from 32.

POD is a so-called dimensionality reduction technique and is very closely related to the more well-known Principal Component Analysis (PCA) in statistics. In physics it's ofen used to analyze turbulent flows[34]. In POD we wish to describe a function as a sum over its variables:

$$f(x,t) = \sum_{n}^{r} \alpha_n(x)\varphi_n(t)$$

where α_n and φ_n are called respectively the spatial and temporal modes. Associated with each mode n is an energy-like quantity E_n. Modes with a higher 'energy' E_n contribute more to the signal f than modes with a lower energy and we can thus approximate the signal by selecting the k modes with the highest energy. A typical log10 energy spectrum has a 'knee' in the values, as shown in figure fig. ??. Modes with an energy below the knee are noise, and modes above signal.



The wavelet transform is very similar to the Fourier transform, but uses wavelets as its basis. A fourier transform gives the frequency domain with infinite precision, but tells nothing about the locality of the frequencies (.i.e when each frequency is present in a signal). By using a wavelet (a wave whose amplitude is only non-zero for a finite time), we sacrifice precision in the frequency domain but gain information on the locality instead. Performing a wavelet transform transforms the signal into the sum of an approximation and its details and we can filter this analogous to a fourier filter. Due to its locality however, noise is filtered out, by sharpness is retained.

WavinPOD combines these two techniques by applying wavelet filtering to the POD modes. In detail, one first decomposes the problem with a POD transformation. The energy spectrum of this transformation is shown in figure fig. ?? and we select a cutoff of 27. All retained modes are wavelet filtered and are then retransformed to give the denoised and smoothed signal. In figure fig. 2.1 we show the results of the smoothing in the time and spatial domain. In the left panel we show the signal of a single pixel in time, while we plot a line of pixels in a single frame in the right panel. The red lines denote the original (unfiltered) signal, the blue line the effect of just applying a POD and the black one the

result of the WavInPOD technique. Note that the effect of the wavelet filtering is to smooth the signal significantly and in comparing the original data to the filtered data that we've retained the sharpness of the features whilst obtaining a smooth signal.

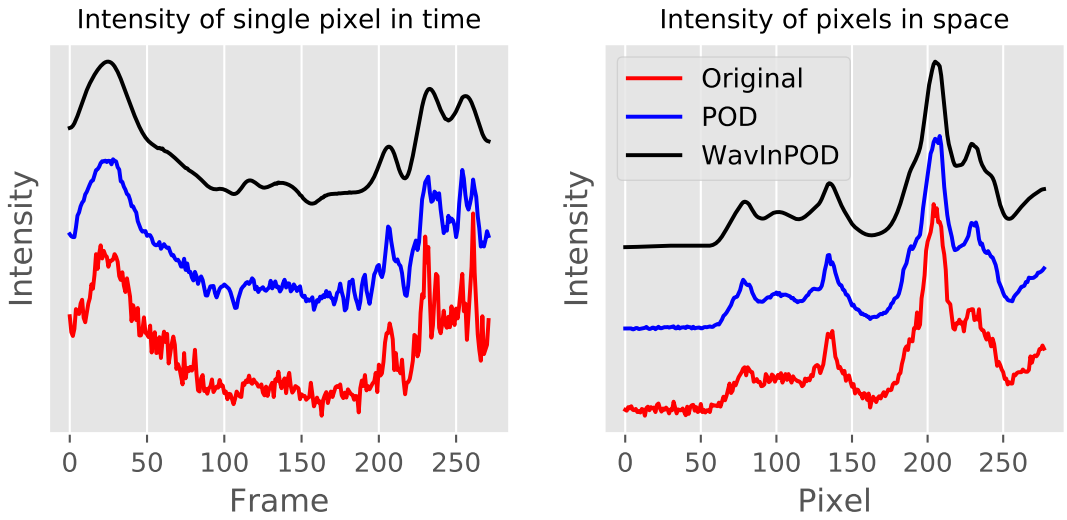


Figure 2.1: Effect of POD with a cutoff of 27 and wavelet filtering with a level 3 db4 wavelet. Left panel shows the result in the time domain, right panel in the spatial domain. Lines have been offset for clarity.

2.3 Step 2 - Derivatives

After having denoised the images, we calculate the spatial and temporal derivatives. Obtaining correct numerical derivatives is hard and becomes much more so in the presence of noise[36]. Next to a finite-difference scheme, one can for example (locally) fit a polynomial and take its derivative or use a so-called tikhonov-regularizer[37]. The computational cost of these methods is high however and they don't scale well to dimensions higher than one. For our spatial derivatives these methods are thus not available. In fact, obtaining the gradient of a 2D discrete grid has another subtlety which we need to adress.

Naively, one could obtain the gradient of a 2D grid by taking the derivative using a finite difference scheme with respect to the first and second axis. If there are features on the scale of the discretization ($\sim$ few pixels), such an operation will lead to artifacts and underestimate the gradient. These issues have long been known and several techniques have been developed to accurately calculate the gradient of an 'image'. The most-used image-gradient technique is the so-called Sobel operator. It belongs to a set of operations known as *kernel*

operators. Kernel operators are expressed as a matrix and by convolving this matrix with the matrix on which the operation is to be performed, we obtain the result of the operator. We show this for the Sobel operator.

Consider a basic central finite difference scheme:

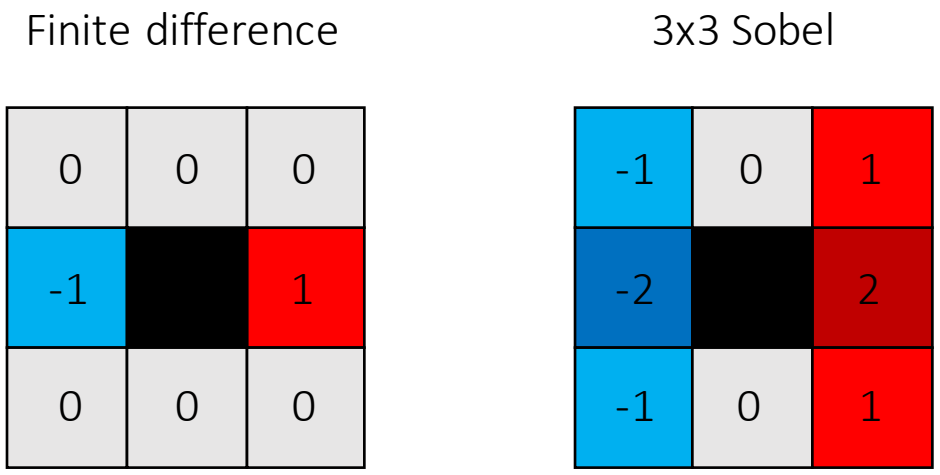
$$\frac{df}{dx} \approx \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$$

where h is defined as $x_{i+1} - x_i$. In terms of a kernel operator, we rewrite this as:

$$\frac{1}{2} \cdot \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

where we have set $h = 1$, as the distance between pixels is one by definition. By convolving this matrix with the matrix $\mathcal{A}$ we obtain the derivative of $\mathcal{A}$:

$$\partial_x \mathcal{A} \approx \mathcal{A} * \frac{1}{2} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$



As stated, this operation is inaccurate and introduces artifacts. To improve this, we wish to include the pixels on the diagonal of the pixel we're performing the operation on as well (see figure fig. ??). The distance between the diagonal pixels and the center pixel is not 1 but $\sqrt{2}$ and the diagonal gradient also needs to be decomposed into $\hat{x}$ and $\hat{y}$, introducing another factor $\sqrt{2}$. The kernel thus obtained is the classic 3×3 Sobel filter:

$$G_x = \frac{1}{8} \cdot \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

Increasing the size of the Sobel filter increases its accuracy and we've implemented a 5x5 operator. Implementing the derivative operation as a kernel method is also beneficial from a computational standpoint, as convolutional operations are very efficient. The Sobel filter is usually applied to an image and hence is often said to calculate the image-gradient, but due to its separability is possibe to scale this method to an arbitrary number of dimensions.

2.4 Step 3 - Segmentation

In the case of the RUSH data, obtained images and movies often contain multiple cells. Each of these cells can be further segmented into two more areas of interest: the cytoplasm, which is were we want to fit our model and the Golgi apparatus. We wish to make a mask which allows us to separate the cells from the backgroud and themselves and divide each cell into cytoplasm or Golgi. Figure fig. ?? shows four typical frames in the MANII transport cycle. Note that no sharp edges can be observed, especially once the MANII localizes in the Golgi. No bright field images were available as well, together making use of techniques such as described in 38 unavailable. We have thus developed two methods which allow us to segment the image and the cells, based on Voronoi diagrams and the intensity.

2.4.1 Voronoi diagram

Consider again the frame on the left of figure fig. ??. Note that in early frames such as this one, the cargo (i.e. fluorescence) is spread circumnuclear. Applying a simple intensity based segmentation gives us a number of separate areas, which *very* roughly correspond to a cell. We can then pinpoint each cells' respective center. Given n points, Voronoi tesselation divides the frame into n areas, where point i is the closest point for each position in area $\mathcal{A}_i$. The hidden assumption here is thus that each pixel belongs to the cell center it's closest too. Although this is a very big assumption, in practice we've found this to be reasonable. Furthermore, one can add 'empty' points to make the diagram match observations. Assuming small movements of the cell, this isn't an issue either for this technique, as we are assigning an area to each cell instead of very precisely bounding it. This also allows us to calculate the Voronoi diagram in the early frames and apply the segmentation to the entire movie. The result of this segmentation for MANII is shown in figure fig. 2.2. Each cell centre is denoted by a dot, while the lines denote the border between each voronoi cell.

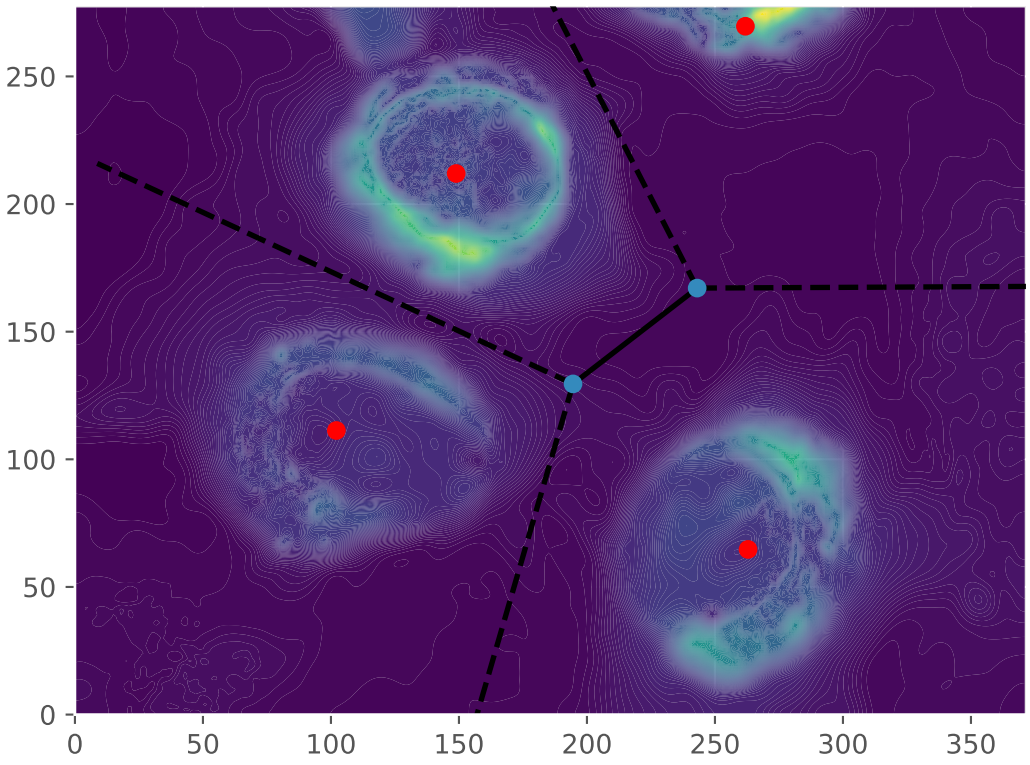


Figure 2.2: The obtained mask. Red dots are cell centers, dashed lines infinite edges and solid lines finite edges.

2.4.2 Intensity

The Voronoi technique works very well for an area-based approach, but for analyzing our fitting data we would like a more precise mask - although we still don't require pixel-level accuracy. From the movies, the Golgi is clearly visible and we can separate the cytoplasm from the background, with a big 'gray' area inbetween. We thus turn to an intensity based approach. We have developed the following approach for localizing the Golgi:

1. Normalize the intensity I between 0 and 1.
2. Sum all the frames in time: $\sum_n I(x, y, t_n)$. A typical result is shown in figure fig. 2.3 .
3. Threshold the image to obtain the mask. This is either done automatically through an Otsu threshold or by manually adjusting the threshold until desired result.
4. The mask is postprocessed by filling any potential holes inside the mask.

This procedure was unable to reliably separate the background from the cytoplasm. Noting that while the cytoplasm might not have the intensity as the golgi, its time derivative should be higher than the rest of the areas. We replace step two by $\log_{10}\left(\sum_n I(x, y, t_n) \cdot \partial_t I(x, y, t_n)\right)$, where the time derivative has been normalized between 0 and 1. Figure fig. 2.3 shows our final results. The upper two panels show the images obtained after performing the summing operation for the Golgi and cytoplasm respectively, while the lower left panel shows the final mask obtained after thresholding these two images. For comparison, we plot frame to compare the mask to.

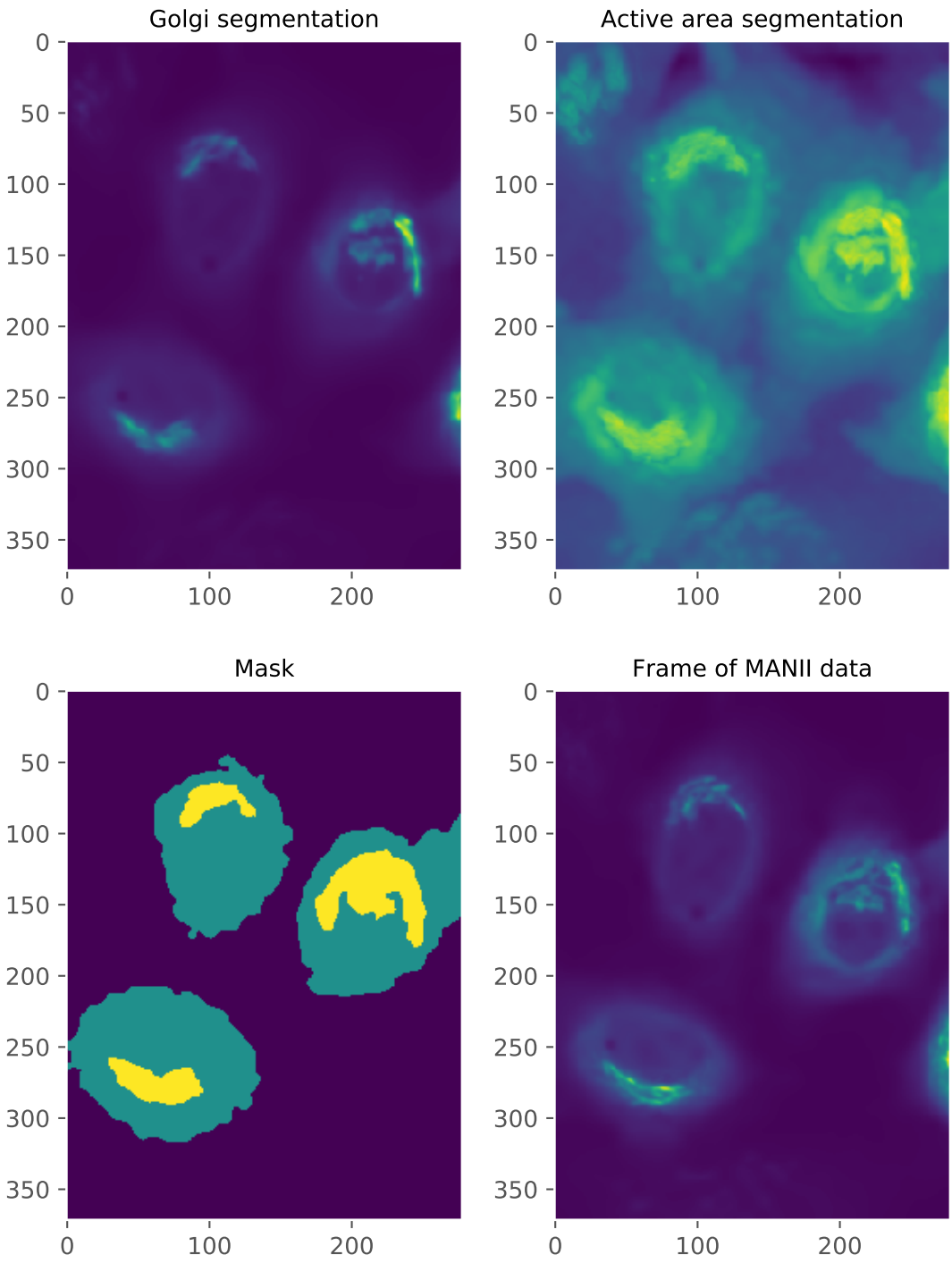


Figure 2.3: Four panels showing the different stages of making the mask. From segmenting the upper two panels we determine the golgi and active area, leading to the mask in the lower left. Compare the to the lower right.

2.5 Step 4 - Fitting

The final step in our method is to fit the our model to the data. By calculating the derivatives, we have reduced our PDE-model to a generic model, which allows us to use virtually any fitting method. For simplicity, we use least-squares, but one could use a Bayesian method to obtain not only the fit, but also the uncertainty.

Equation eq. 4.9 assumes a model with constant coefficients. In reality, coefficients will be spatially and even temporally varying. To circumvent this issue, we assume the coefficients can be assumed to be locally constant. We thus assume that for a small area we can fit the model using constant coefficients. We perform this operation for every datapoint in a sliding-window manner, as shown in figure fig. 2.4, thus ending up with a coefficient field.

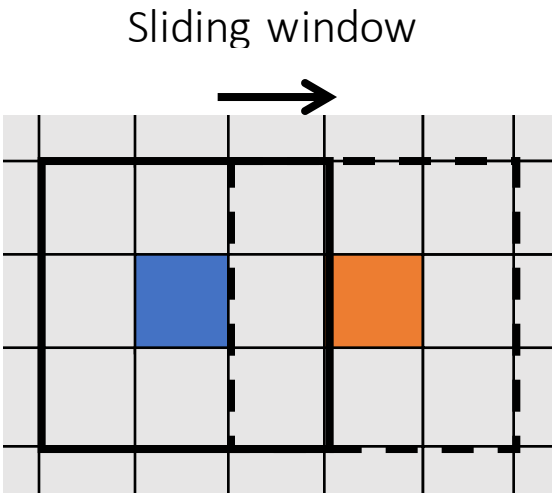


Figure 2.4: Schematic overview of the sliding window technique. The solid black line encompasses an area around its blue coloured central pixel and the fit output is assigned to that pixel. We then move the window (dashed black line) and perform the fit for the orange coloured pixel.

In the next chapter we apply this method to the RUSH experimental data.

3

Data analysis

In this chapter we apply the method developed in the previous chapter to experimental data obtained using the RUSH technique. Our first section introduces the RUSH technique and discusses our model for the intracellular transport. In the following section we discuss the experimental data, investigate the fluorescence curves of several areas of interest. and gain more insight into the data by studying its time derivative. We then present a linear least squares fit and show that this can lead to unphysical results. We end with a short section of both recommendations for the experimentalists as well as a number of ways to improve the method.

3.1 Experimental data

The transport of vesicles from the ERES to the Golgi is both diffusive and directive and a technique known as RUSH (Retention Using Selective Hooks) has recently been developed[16] in the team of Frank Perez at Institut Curie to study this trafficking. RUSH allows precise timing of the release of proteins from the ER and can be used to follow the secretory pathway from the ER to the Golgi and even post-golgi using fluorescent live-cell imaging. Several other methods have been developed (???–golgi_1997, ???–golgi_2018), but lack the non-toxicity, timing and versatility of the RUSH technique.

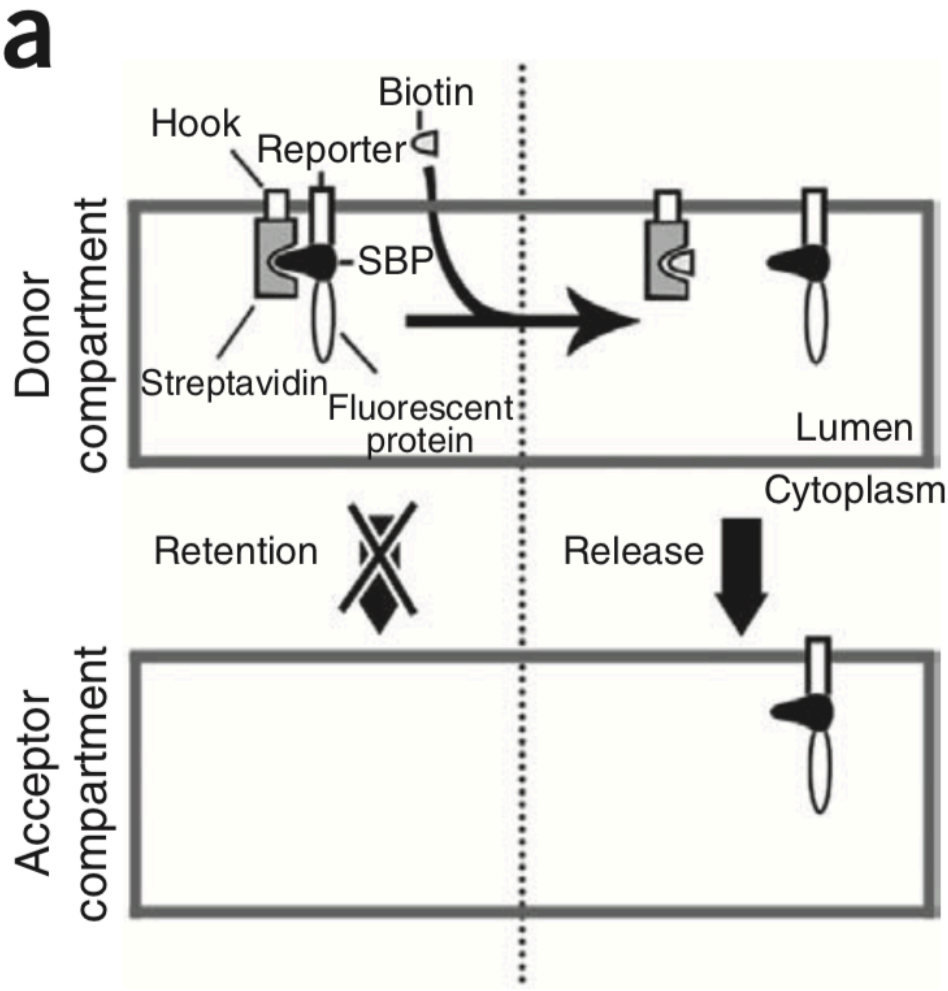


Figure 3.1: Schematic overview of the RUSH system. Image taken from 16

Figure fig. 3.1 shows the principle of the RUSH system. Inside the ER, a core streptavidin is fused to it using a hook protein. Another protein known as a streptavidin-binding-protein (SBP) binds to streptavidin, but connected to the SBP are also the protein to be transported ('reporter') and a fluorescent protein. Upon the addition of biotin, the SBP is released from the streptavidin as the biotin binds to it. The SBP-reporter-fluorescent complex then exits the ER and can be followed the entire secretory pathway with fluorescence microscopy.

The RUSH technique can be used for many different proteins, but in this thesis we mainly focus on the α-mannosidase-II, generally referred to as ManII. The ManII protein resides in the Golgi apparatus and thus upon reaching it will remain there. This means that the data we obtain will only contain transport *towards* the golgi, greatly simplifying the analysis as we won't have to post and pre-golgi traffic. Figure fig. 3.2 shows two frames in a typical RUSH experiment of ManII trafficking. The left panel shows an image obtained just after the addition of the biotion, so that most of the cargo is still retained in the ER. A later frame is shown on the right: we can observe the localization of fluorescence in the Golgi, while there's still fluorescence in the rest of the cells.

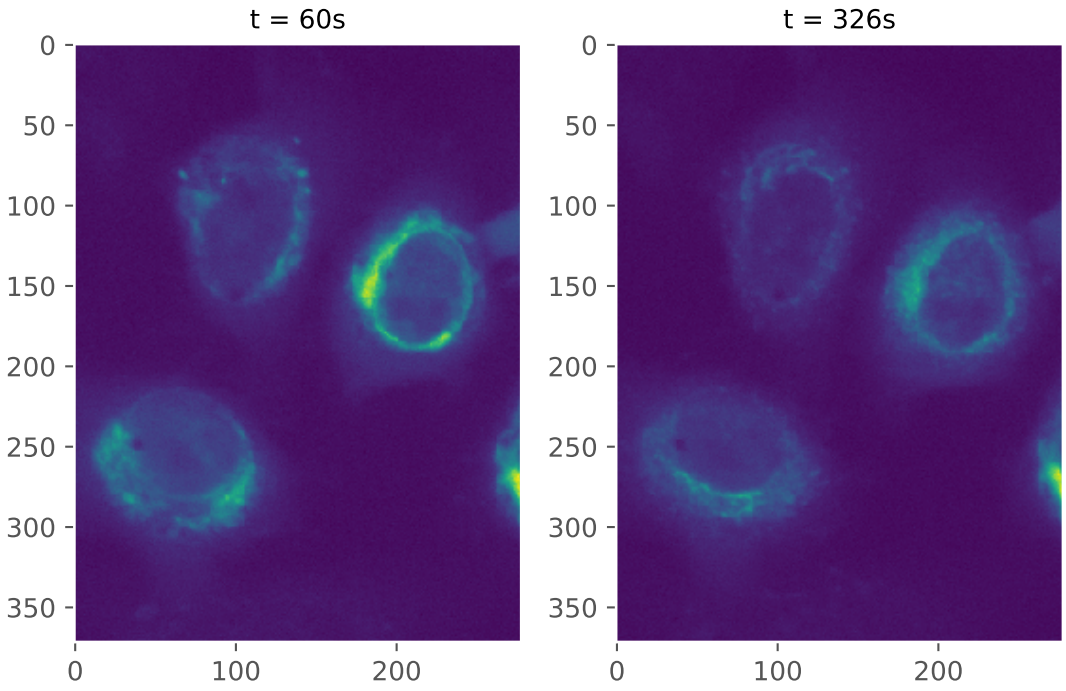


Figure 3.2: Two frames of the ManII transport images using the RUSH technique.

3.1.1 Model

Vesicles exiting the ERES are transported towards the ER over the microtubules. This is a stochastic process with the proteins detaching from and (re-) attaching to the microtubules randomly, while the vesicles move diffusely once detached. Several models have been developed to describe such intracellular transport processes (6, 39), many in the light of virus trafficking (40, 41, 42). In general, these models assume a two population model, with one

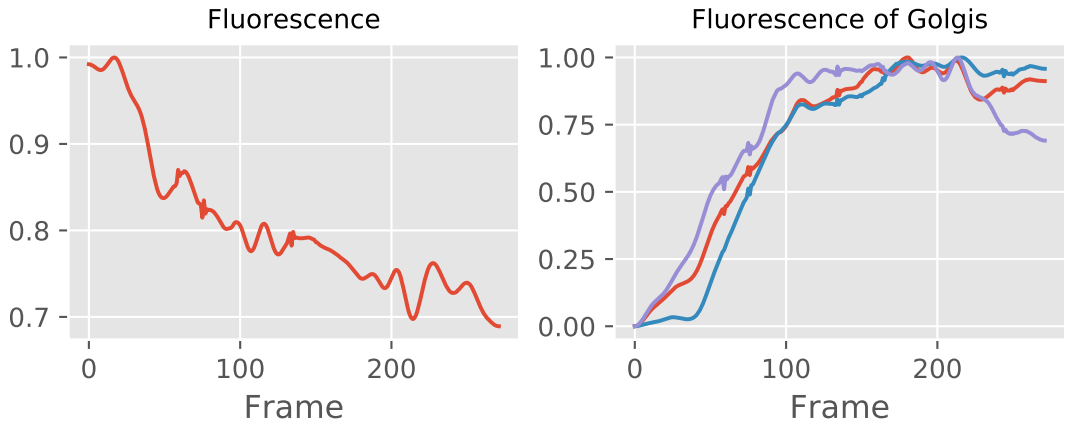
population being cargo attached to a microtubule and another cargo freely diffusing in the cytoplasm. If one assumes that the timescale for attaching and detaching from the microtubules is much smaller than the transport timescale, the two populations can be assumed to be in equilibrium. In this assumption, known as a quasi-steady-state reduction, the two population model reduces to a Fokker-Planck equation. As the Fokker-Planck equation is functionally equivalent to an advection-diffusion equation, we hypothesize that we can model protein transport using an advection-diffusion equation:

$$\partial_t c = D\nabla^2 c - v\nabla c \tag{3.1}$$

where c is the concentration of the cargo, D a diffusion coefficient and v an advection velocity. Equation eq. 3.1 is thus the model we fit our data to. Note that the fluorescence images obtained from the RUSH experiment return an intensity I and not a concentration c, and hence we make the assumption $c \propto I$.

3.2 Initial analysis

We first study the evolution of the fluorescence in two ways, plotting both the mean fluorescence for each frame and the mean fluorescence in the golgi region. To get rid of the background in our statistics, we normalize the fluorescence between before computing the mean. The left panel of frame #fig:fluorescence shows the average fluorescence of each frame and shows a significant drop of almost 30% between the initial and final frame. We observe a strong initial drop and a slower decay after roughly frame 100, which can probably be attributed to photobleaching. The transition between the two decays roughly matches the saturation of the fluorescence in the Golgi (see the right panel), casting strong doubts on our assumption that $c \propto I$. Compensating for this is possible (see 15), but requires significant effort on the experimentalists' part and if the difference between two subsequent frames is small the effect on our fit is neglegible.



.

In the right panel we show the fluorescence in the Golgi ROI for each of the three cells. Specifically, we plot the mean intensity in each ROI, normalized on the maximum intensity and compensated for the loss of fluorescence as shown in the left panel. Interestingly, we observe that all three curves roughly follow a similar pattern. Although one of the cells (blue line) seems to have some sort of delay, the fluorescence seems to increase in a linear way up to frame 100, when the fluorescence saturates. The purple cell shows a significant drop at frame 200, but since the ManII protein remains in the Golgi, this is not caused by any type of intracellular transport and thus not of interest to us. The linear increase and common pattern suggests that the transport properties are not concentration dependent at these concentrations.

To perform our fit in the next section, we have also determined the time derivative of the dataset, $\partial_t I$. We plot this for four different frames in figure fig. 3.3 .

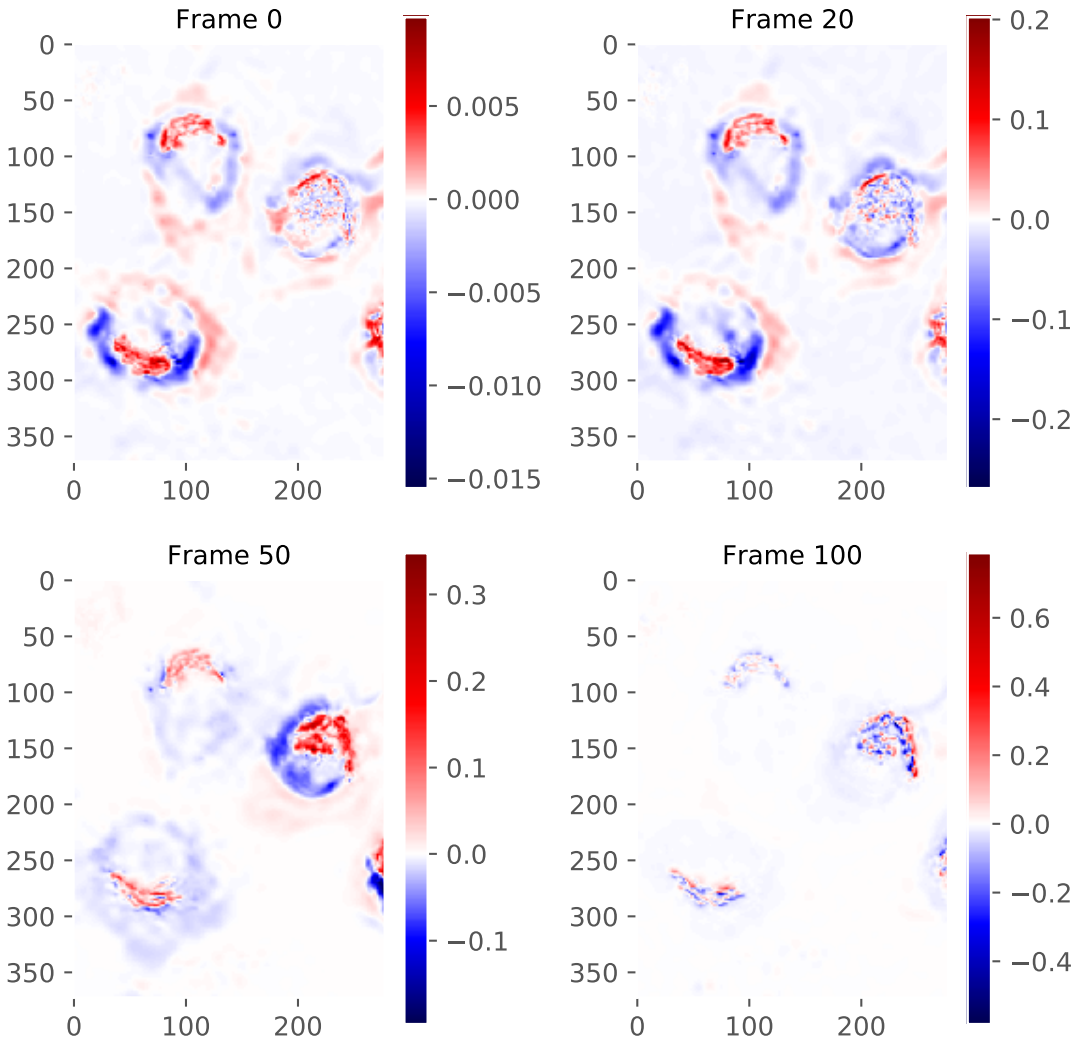


Figure 3.3: The determined time derivative four different frames of the ManII RUSH experiments.

Areas where the derivative is positive (thus were the concentration increases) are coloured red, while areas where the concentration decreases are coloured blue. As expected, the Golgi shows up in each cell as a bright red object. Note however that we also observe red areas towards the edges of the cell. As the concentration close to the Golgi decreases, the red area moves outwards and slowly takes over the blue area. This could be caused by ERES acting as a in that area, but given that ERES are located throughout the cell it seems more likely such a pattern would be caused by diffusion.

3.3 Analysis of LS-fit

In this section we analyse results of the least squares fit. We've used a 7x7 window in the spatial domain to perform the sliding window operation, fitting each frame of the movie independently. We analyse the diffusion, advection and the error of our fit. These fits are movies and we're hence unable to print them - please find our these movies at our GitHub. In figure fig. 3.4 we show two typical inferred diffusion fields in the upper row, just after addition of the biotin (frame 4) and halfway to the complete saturation of the Golgi (frame 40).

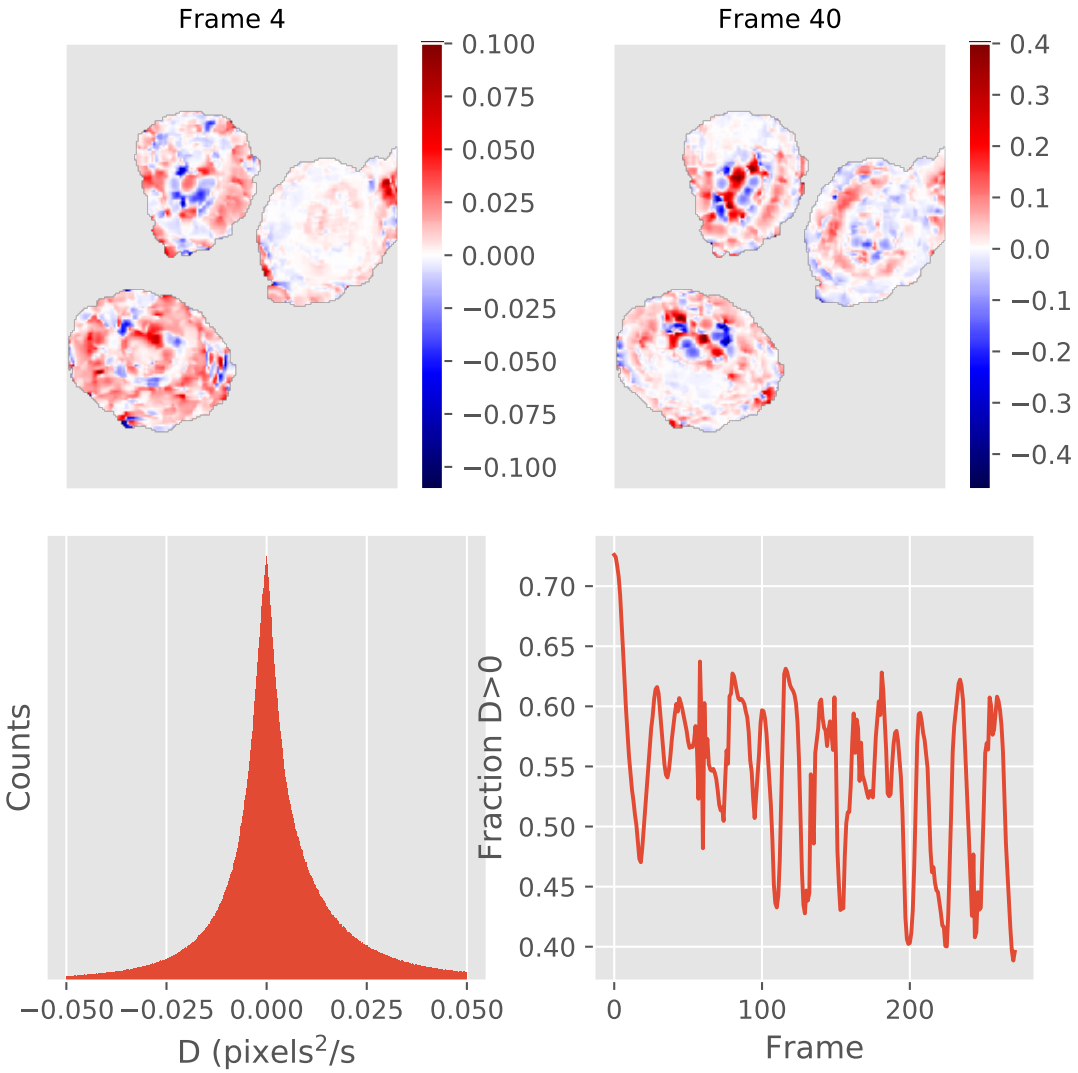


Figure 3.4: Analysis of the inferred diffusion field. The upper row shows the inferred field at two frames, while the lower row shows the distribution of values and the fraction of physical values as a function of time.

Another problem is that we observe many areas with a negative -unphysical- diffusion coefficient. In the lower left panel we plot the distribution of values. Analysis shows that roughly 40% of the inferred field has a negative diffusion coefficient. In the lower right panel we have calculated this fraction as a function of time. It shows that, save for a few initial frames, this fraction is not (strongly) time-dependent. Results are slightly skewed though, since many coefficients are negative but extremely close to zero (e.g -10^{-4}). Nega-

tive diffusion coefficients correspond to clustering, but could also be the result of an incorrect fit. We investigate this in depth after studying the advection profiles, which we show in figure fig. 3.5 . In the four panels we show the inferred velocity in the $\hat{x}$ and $\hat{y}$ direction, the magnitude of the velocity and the angle.

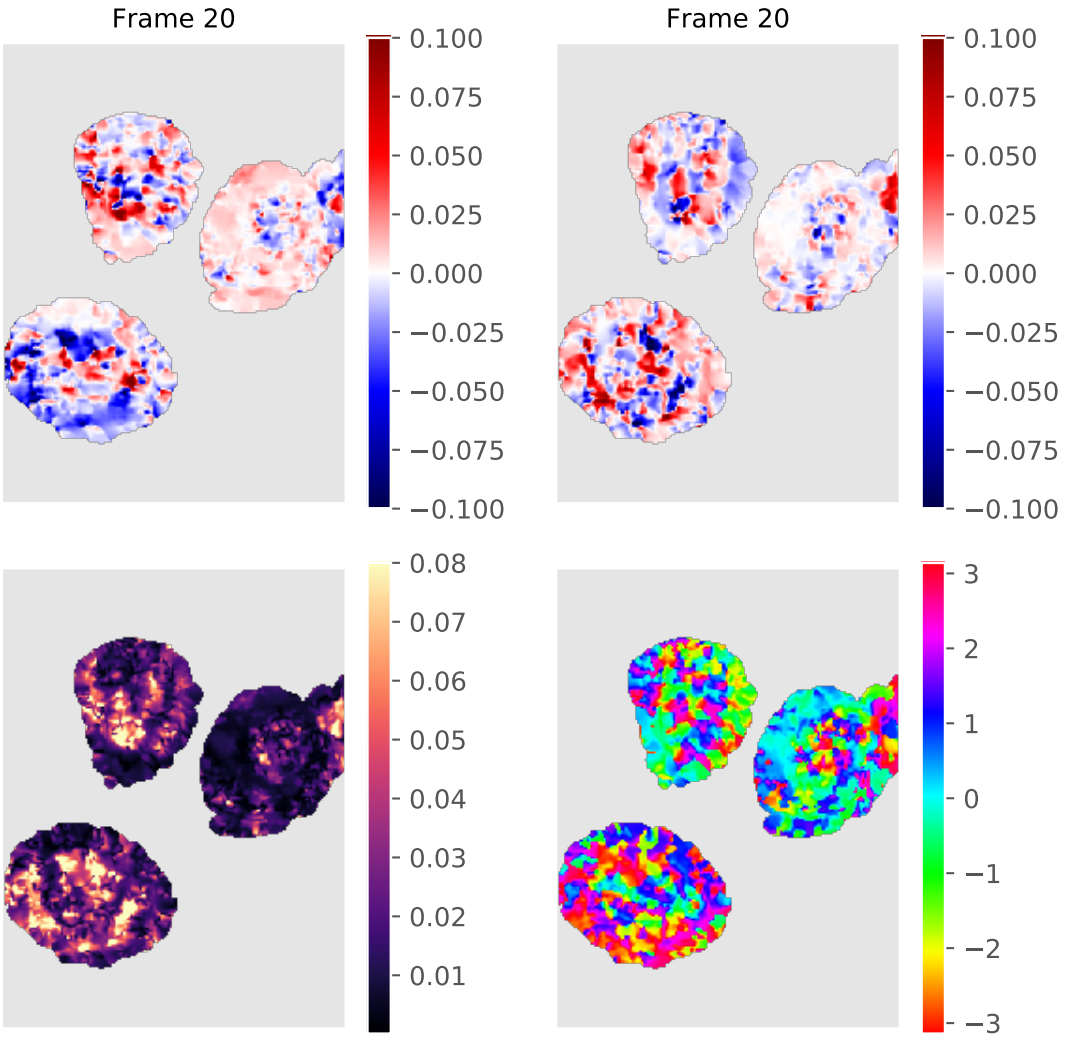


Figure 3.5: Analysis of the velocity fields. The upper rows show respectively v_x and v_y, wwhile the lower row shows the magnitude and angle.

Similar to the diffusion, we observe patterns both in time and space bigger than our fitting window, meaning that the fit isn't completely random. On the other hand, we are not able

to discern any specific flow from the figures in fig. 3.5. To gain more insight into our fit, we analyze a single pixel in time. Figure fig. 3.6 shows the diffusion and advection velocities as a function of time. We've plotted a scaled and translated signal of that pixel in a black dashed line. This pixel is initially constant and then decreases to noise level. Note that initially, while the signal is constant, the diffusion constant is negative. Once the signal starts decreasing, or, in other words, cargo starts flowing from that pixel, we see a physical diffusion constant and non-zero velocities. Once the signal returns to around noise-level at pixel 50, the inferred velocities and diffusion constant seem to become random around 0. In other words, our method seems to work when cargo is flowing, but struggling when the signal is either constant or at noise level. We observe similar behaviour in other pixels, so we contribute (most of) the unphysical diffusion values to constant and noise-level signal. Performing the fit with the diffusion constant constrained to be larger than zero lead to the negative coefficients in the free fit to being set to zero.

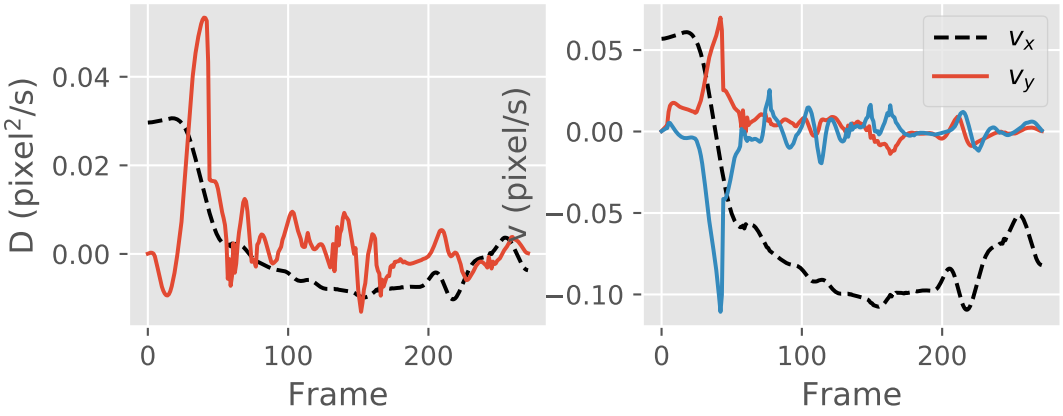


Figure 3.6: Diffusion and advection velocities of a single pixel in time. We've plotted (scaled and translated) signal as a black dashed line to find any correlation.

This doesn't completely explain the magnitude of the coefficients though, which is significantly lower than expected. One possibility is the 'mixing' of the transport fluorescence with the fluorescence of the ER. After the addition of biotin, the fluorescent cargo gets released, but still has a finite residence time in the ER. Since the obtained images are projected over an axis, changes in fluorescence we observe can be both due to intracellular transport as well as processes inside the ER. If these processes have different timescales, this can strongly affect the inferred coefficients.

Another possibility is that we've assumed that a concentration of fluorescent particles leads

to some sort of 'mean-field' fluorescence which we can describe by an advection-diffusion equation. Once the size of the particles becomes on the order of the pixel size, this assumption breaks down. In the case of the ManII trafficking, the pixels are roughly two to three times the size of a vesicle, meaning that we are at the limits of the 'mean-field' assumption.

3.4 Conclusion

We've applied the method developed in the previous chapter to the RUSH trafficking data of the ManII protein. Although our fit shows patterns in the coefficient fields larger than our fitting window, pointing at some underlying pattern, we're unable to make any conclusions about our model. It's been shown that the model seems to perform well when the data is transient, i.e. when cargo is actually flowing. However, for most of the time the data is either constant or at noise level, preventing us from making any conclusions.

Improvements to our method fall roughly into two categories. The first category concerns improvements to the calculation of the derivatives. We've implemented a rather basic 5x5 Sobel filter, but implementing more advanced methods which would result in more accurate derivatives would probably make the biggest improvement. The second category would be improve the fitting procedure. The most obvious candidate is implementing some sort of Bayesian method which would return not just the most probable coefficient, but the entire probability distribution.

4

Physics Informed Neural Networks

In the previous chapters we showed the difficulties in fitting a model in the form of a partial differential equation to spatio-temporal data. The method we developed was a classical numerical approach, separating the problem into several substeps such as denoising, smoothing and numerical differentiation. In this chapter we present an alternative technique, generally referred to as Physics Informed Neural Networks (PINNs), which has already shown impressive performance in fitting models and numerically solving equations[43],[44],[45],[27],[46]. As neural networks are a new technique in phyics, this chapter also served as introduction to neural networks in general. The chapter thus has the following structure:

- Neural Networks - This part will cover the basics of neural networks: their inner workings, training and other general features.
- Physics Informed Neural networks - In this second part we introduce the concept behind PINNs, use it to solve a toy problem and apply it to our RUSH data.
- Conclusion - Finally we summarize the results and observations from the previous sections.

4.1 Neural Networks

Normally when programming a computer to perform some task, we break the problem into smaller pieces and writing down precise instructions. Often, a model of the underly-

ing process is also needed to transform some input into an output. The performance of the algorithm is then only as good as the underlying model and when dealing with complex processes, such models often become intractable or oversimplified. Artificial Neural Networks (ANNs) are a different approach to such a problem. Instead of being *programmed*, they are *trained* and hence 'learn' an underlying model. In a process known as *supervised learning*, the network is given inputs and the desired outputs for each input. Training the network then consists of adjusting its internal parameters until the predictions match the desired outputs. In the next sections we discuss how to adjust these parameters.

4.1.1 Architecture

An excellent introduction is given by Michael Nielsen in his freely available book "Neural networks and deep learning." The following section has been strongly inspired by his presentation.

At the basis of each neural network lies the neuron. It transforms several inputs into an output in a two-step process. In the first step, the inputs x are multiplied with a weight vector w and a bias b is added:

$$z = w\mathrm{x} + b$$

z is called the weighted input and is transformed in the second step by the neuronal *activation function* σ. This gives the output of the neuron a, also known as the activation:

$$a = \sigma(z) = \sigma(w\mathrm{x} + b) \tag{4.1}$$

The activation introduces non-linearity into the network and hence is crucial; without it a neural network would merely be several matrix multiplications. The classical activation is a tanh, i.e $\sigma(z) = \tanh(z)$, but many other forms exist, each having its benefits. Several neurons in parallel constitute a *layer* and several layers can be connected to create a network. The layers in the middle of the network are referred to as *hidden layers*. An example of such a network with two hidden layers is shown in figure fig. 4.1.

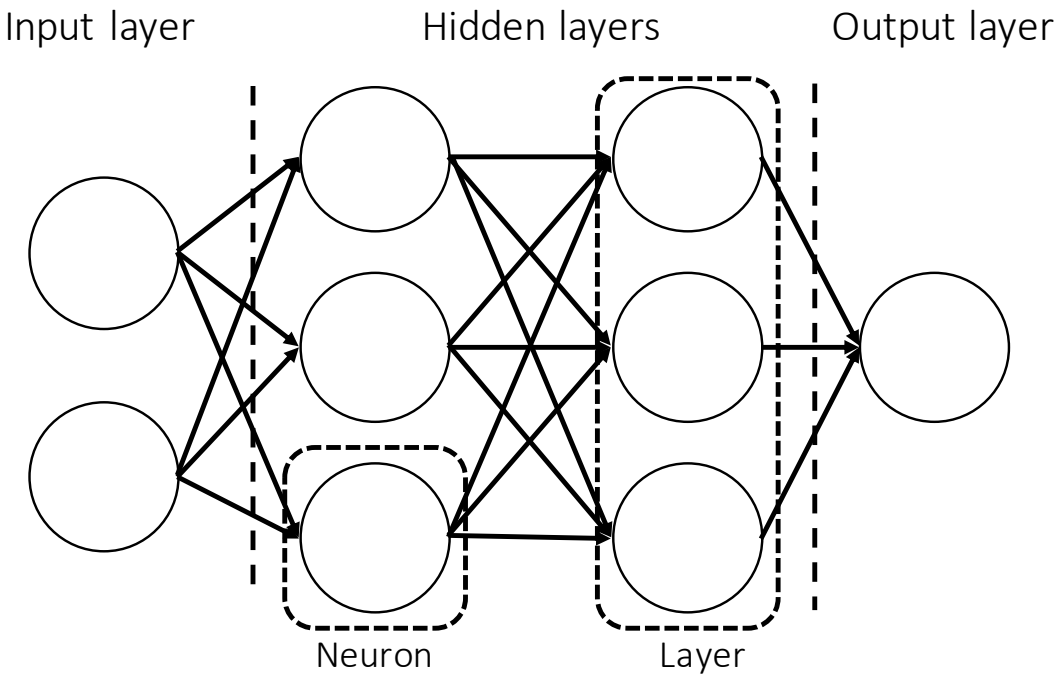


Figure 4.1: Schematic overview of a neural network. The left layer is known as the input layer, the right layer as the output layer and the layers inbetween are referred to as hidden layers.

4.1.2 Training

Consider again equation eq. 4.1. In a network with multiple layers, it is useful to express the activation a^l of layer l in the activation of layer $l-1$, so that eq. 4.1 becomes :

$$a^l = \sigma(z^l) = \sigma(w^l a^{l-1} + b^l)$$

where w^l and b^l are respectively the weight matrix and bias of layer l. As stated, training a neural network means adjusting the weights w^l and biases of each layer b^l until the output of the neural network a^L - the activation of the last layer L - matches the desired output y_i. We thus require a metric to define the difference between the prediction and the desired output. This metric is known as the *cost function* $\mathcal{L}$ and one of the most commonly used cost functions is the mean squared error:

$$\mathcal{L} = \frac{1}{2n} \sum_i |y_i - a_i^L|^2 \tag{4.2}$$

where n is the number of samples, y_i the desired output of sample i and the prediction of the network given the inputs of sample i. As the cost function is a measure of the difference between the prediction and desired outputs, training a neural network comes down to minimizing the cost function. Such minimization problems are solved by gradient descent techniques.

Gradient descent techniques are based on the fact that given some position, the fastest way to reach the minimum from that position is by following the steepest descent. Thus, given a function $f(\mathrm{x})$ to minimize w.r.t to x, we guess an initial position x_n and iteratively update it until it converges:

$$\mathrm{x}_{n+1} = \mathrm{x}_n - \gamma \nabla f(\mathrm{x}_n) \tag{4.3}$$

γ is known as the learning rate and sets the 'stepsize'. Although this is an iterative technique, if the minimization problem is convex (i.e. no local minima), gradient descent is guaranteed to converge to it. Note that gradient descent requires calculation of the derivative w.r.t to the variables of the function to be minimized. In other words, one needs to know the derivative of the cost function w.r.t each of the weights and biases in the network. A naive finite difference scheme would quickly grow computationally untractable, even for networks with just two hidden layers. Alternatives to gradient descent exist, but all require calculation of the derivatives. In the next section we present an algorithm which is able to efficiently calculate these derivatives.

Back propagation and automatic differentiation

In this section we introduce the so-called *backpropagation* algorithm. The backpropagation algorithm allows for the efficient calculation of the cost function derivatives in a neural network. For simplicity, we move away from a vector notation and introduce w^l_{jk}, the weight of the k-th neuron in layer $l-1$ to neuron j in layer l and b^l_j, the bias of the neuron j in the l-th layer. We introduce the error of neuron j in layer l as:

$$\delta^l_j = \frac{\partial C}{\partial z^l_j}$$

We can rewrite this using the chain rule as:

$$\delta^l_j = \sum_k \frac{\partial C}{\partial a^l_k} \frac{\partial a^l_k}{\partial z^l_j}$$

The second term on the right is always zero except when $j = k$, so the summation can be dropped. Given equation eq. 4.1, we note that $\partial a_j^l / \partial z_j^l = \sigma'(z_j^l)$. For the last layer $l = L$, we can directly calculate the derivative, resulting in:

$$\delta_j^L = |a_j^L - y_j| \sigma'(z_j^L) \tag{4.4}$$

Equation eq. 4.4 relates the error in the output layer to its activation and weighted input. Again using the chain rule, we can express the error in a layer l, δ_j^l ,in terms of the error in the next layer, δ_j^{l+1}:

$$\delta_j^l = \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_k \delta_k^{l+1} \frac{\partial z_k^{l+1}}{\partial z_j^l}$$

Using equation eq. ??, we obtain after substitution:

$$\delta_j^l = \sum_k \delta_k^{l+1} w_{kj}^{l+1} \sigma'(z_j^l) \tag{4.5}$$

Equation eq. 4.4 gives us the error in the final layer, while equation eq. 4.5 allows us to propagate the error back through the network - hence the algorithm is named backpropagation. Two more expressions are needed to relate the error in each neuron δ_j^l to the derivatives w.r.t. the weights and biases. Making use yet again of the chain rule gives the last two backpropagation relations:

$$\frac{\partial C}{\partial b_j^l} \frac{\partial b_j^l}{\partial z_j^l} = \frac{\partial C}{\partial b_j^l} = \delta_j^l \tag{4.6}$$

$$\delta_j^l = \sum_k \frac{\partial C}{\partial w_{jk}^l} \frac{\partial w_{jk}^l}{\partial z_j^l} \rightarrow \frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l \tag{4.7}$$

Given the four fundamental backpropagation relations, we state the algorithm. It consists of four steps:

1. Complete a forward pass, i.e., calculate a_j^L.
2. Calculate the error in the final layer using eq. 4.4 and propagate it backwards using eq. 4.5 to obtain the error in each neuron. Using eq. 4.6 and eq. 4.7, calculate the derivatives required for the minimizer.

3. Perform a minimization step (e.g. equation eq. 4.3) and update the weights and biases.
4. Return to step one until the minimization algorithm in step three converges.

Mathematically, back propagation is a version of a more general technique known as automatic differentiation. Suppose we want to calculate the derivative of some data $f(x)$. Symbolic differentiation would give the most precise answer, but often the function f is not known. Furthermore, even if f would be known, it quickly becomes too hard to calculate a symbolic derivative of a complex function f. One could then turn to numerical differentiation using some finite difference scheme or locally fitting a polynomial whose derivative is then calculated. All these methods require relative closely spaced data and are very sensitive to noise. Automatic differentiation allows for the high precision calculation of numerical derivatives. At its fundamental level, any computational operation, no matter how complex, is a long string of elementary operations whose derivative is easily determined. Using the chain rule, we can then calculate the derivative of any computation in terms of these smaller elementary operations. To see this, consider a function $f(x) = a + bx$. Writing this in terms of elementary operations gives:

$$f(x) = a + bx = w_1 + w_2 w_3 = w_1 + w_4 = w_5$$

The derivative of each subexpression w_i is easily calculated:

$$w_1' = 0, w_2' = 0, w_3' = 1, w_4' = w_2' w_3 + w_2 w_3', w_5' = w_4' + w_1'$$

The derivative of f is then:

$$f' = w_5' = w_4' + w_1' = (w_2' w_3 + w_2 w_3') + w_1' \tag{4.8}$$

We have thus expressed the derivative of f in quantities we know and indeed, after filling in the remaining derivatives we obtain $f' = w_2 = b$. Note the similarity to backpropagation; in automatic differentiation we are only interested in the final expression on the right of equation eq. 4.8, whereas in backpropagation we wish to know the intermediate derivatives (i.e. w_5', w_4') too. Back propagation is thus a version of automatic differentiation in which the intermediate values are calculated too. In the next section we show that automatic differentiation enables easy encoding of physics into a neural network, leading to a so-called Physics Informed Neural Network (PINN).

4.2 Physics Informed Neural Networks

In this section we introduce Physics Informed Neural Networks (PINNs), a recently developed technique[46,27] which merges physical models and neural networks. We first intro-

duce how PINNs encode physical laws and models in neural networks and discuss why this yields such a powerful technique. This is illustrated by applying it to a simple diffusive process and show that even in the presence of noise, PINNs can accurately infer a (spatially-varying) diffusion constant. We then apply a PINN to the RUSH data and end the chapter with our conclusions.

4.2.1 The concept

Consider a set of spatiotemporal experimental data, $u(x,t)$ and a model which describes the temporal evolution of this dataset:

$$\partial_t u = \lambda_1 + \lambda_2 u + \lambda_3 \nabla u + \lambda_4 \nabla^2 u = f(1, u, u_x, ...) \tag{4.9}$$

We now wish to know which value for the parameters λ_i best describes the dataset $u(x,t)$. Naively, one could train a neural network on a training set created by numerically solving eq. 4.9 for different values λ_i and then feed this network the experimental data $u(x,t)$. Although theoretically this yields the correct result, for complex processes such as a Navier-Stokes flow or the Schrodinger equation this quickly grows intractable due to the massive amount of training data required for an accurate prediction.

PINNs circumvent this issue by directly encoding physical laws and models such as eq. 4.9 into the neural network. We can write any PDE as:

$$g = -\partial_t u + f(1, u, u_x, u_x x, u^2, ...) \tag{4.10}$$

This function g can be added to the costfunction, because to satisfy the PDE, $g \to 0$:

$$\mathcal{L} = \frac{1}{2n} \sum_i |u_i - a_i^L|^2 + \frac{l}{n} \sum_i |g_i|^2$$

where l sets the effective strength of the two terms. By adding g to the cost function, it acts as 'physics-regularizer' and unphysical solutions are penalized; we have thus encoded the physics directly into the neural network. Note that while we know the form of g, its coefficients λ_i are unknown. However, we can treat the coefficients as variables of the cost function, i.e. $\mathcal{L}(w^l, b^l, \lambda)$ and thus by training the network on the dataset $u(x,t)$, we automatically infer the coefficients. Consequently, we don't need a vast set of training data, as we solve the problem *by* training the network.

Theoretically, PINNs should not only be able to infer constant coefficients, but also coefficient *fields*. Instead of treating the coefficients as a variable to be optimized, we add another

output to the network. Such a network is known as a multi-output PINN and the difference between a single and multi output network is shown in figure fig. 4.2. PINNs can also be used to numerically solve PDEs. By removing the mean squared error term from the cost function but adding initial values and boundary conditions, training the network will now result in the network learning the solution to the PDE g, whilst respecting the given boundary and initial conditions. This alternative means of numerically solving a model doesn't need advanced meshing of the problem domain required in computational fluid dynamics or carefully constructed (yet often unstable) discretization schemes, as it requires the physics to be fullfilled at every point in the spatiotemporal domain.

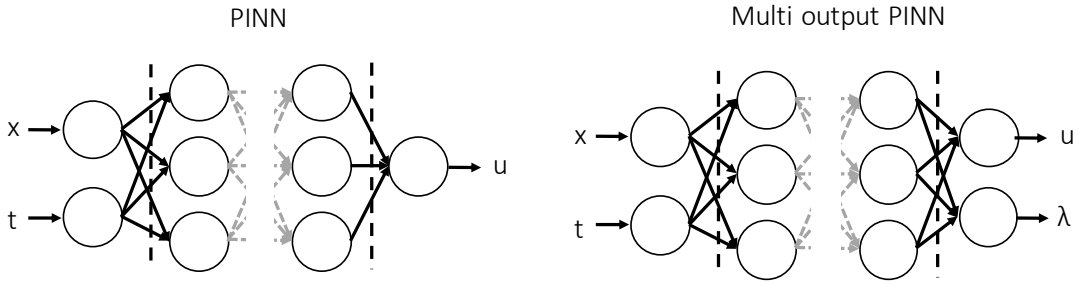


Figure 4.2: Left panel: a single output PINN. **Right panel:** A multi-output PINN. The network now also predicts the coefficients values at each data point.

4.2.2 PINNs in practice

Before applying a PINN to the RUSH data, we study a toy problem to gain more insight into its behaviour. We also prove that a PINN is able to correctly infer a coefficient field from noisy data. Our toy problem of an initial gaussian 1D concentration profile:

$$c(x, 0) = e^{-\frac{(x-x_0)^2}{2}}$$

with $x_=0.5$ and $\sigma = 0.01$ diffusing in a box of length L according to:

$$\frac{\partial c(x,t)}{\partial t} = \nabla \cdot [D(x)\nabla c(x,t)] \tag{4.11}$$

on the spatial domain $[0, 1]$ with perfectly absorbing boundaries at the edges of the domain:

$$c(0, t) = c(1, t) = 0$$

If $D(x) = D$, this problem can be solved using a Greens function. Although being a simple problem, it contains all the essential features of a PINN. For the application of a PINN to more complex systems such as the Burgers, Schrodinger or Navier-Stokes equations, we refer the reader to the papers of M. Raissi et al (46, 27).

4.2.2.1 Constant diffusion coefficient

We first numerically solve equation eq. 4.11 with a diffusion coefficient of $D(x) = D_0 = 0.1$ between $t = 0$ and $t = 0.5$. Using a spatial and temporal resolution of 0.01, our total dataset consists of 5151 samples, while we have configured the neural network with 6 hidden layers of 20 neurons each and have set $\lambda = 1$. The left panel of figure fig. 4.3 shows the ground truth (i.e. the numerical solution of equation eq. 4.11) and the absolute error w.r.t to the groundtruth of the neural network output.

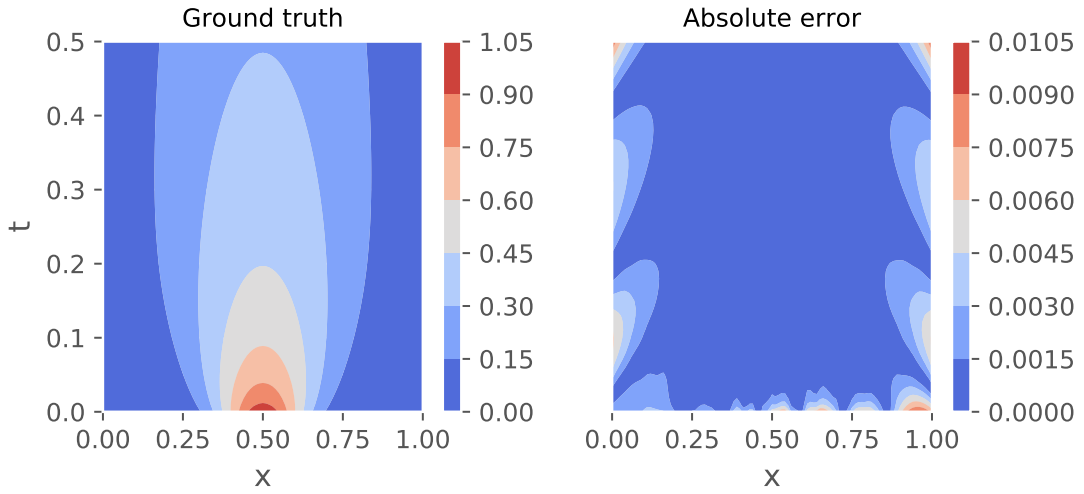


Figure 4.3: Left panel: Simulated ground truth of the problem. **Right panel**: The absolute error of neural network. Note that most of the error is located at areas with low concentration, i.e. signal.

The inferred diffusion coefficient is $D_{pred} = 0.100026$: an error of 0.026%. From the absolute error we observe that the error seems to localize in areas with low concentration. This is a feature we've consistently observed: in areas with low 'signal', the neural network struggles. Considering that in these areas there is simply not much data to learn from, this is not unexpected.

The input data of the previous problem is noiseless and thus of limited practical interest. We add 5% white noise to the data of the previous problem and train the network on this

noisy dataset. Note that the network is now doing two tasks in parallel: it's both denoising the data and performing a fit. In the left panel of figure fig. 4.4 we show the concentration profile at times $t = 0, 0.1$ and 0.5, with the prediction of the PINN superimposed in black dashed lines at each time. On the right panel we show the absolute error with respect to the ground truth. Observe that the error again localizes in areas with low concentration. The inferred diffusion constant is $D_\circ = 0.10052$: an error of 0.52%. Although the error is an order of magnitude higher compared to the noiseless data, an error of less than 1% is extremely impressive.

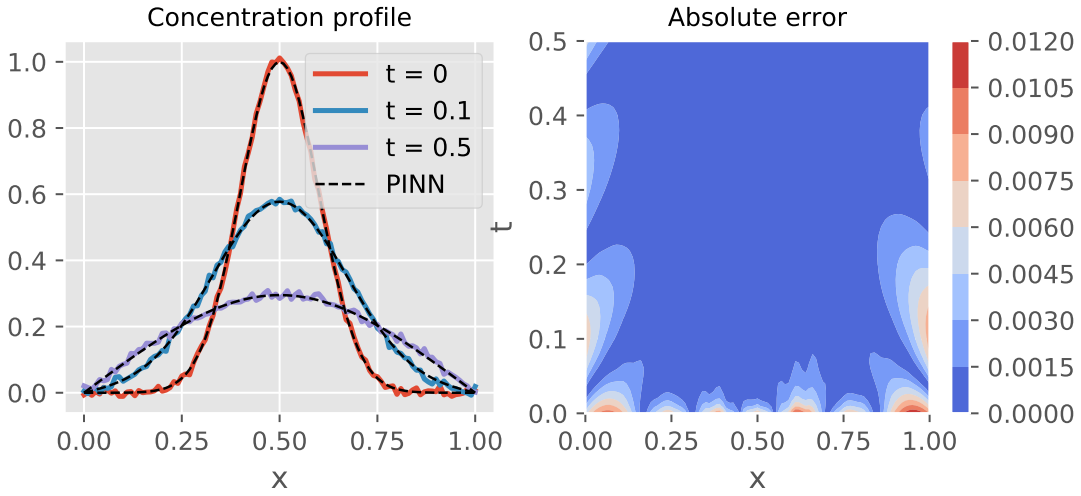


Figure 4.4: Left panel: The original noisy concentration profile at several times with the neural network inferred denoised version superimposed. **Right panel**: The absolute error of neural network with respect to the ground truth. Note that most of the error is located at areas with low concentration.

4.2.2.2 Varying coefficients

As stated, it should be possible to infer coefficient fields by using a two output neural network. We first test this on the noisy constant diffusion ($D_\circ = 0.1$) dataset of the previous problem. In this case, while the neural network is allowed to assign a different diffusion constant to each point in the spatiotemporal domain, it should return $D = 0.1$ for each. Figure fig. 4.5 shows a summary of the results in four panels. In the upper left we show the data on which the network is trained, while the upper right panel shows the predicted concentration profile. Note the excellent match between the two. In the lower right panel we show the inferred diffusion field. We observe a good match in the middle of the plot, but the neural network again struggles in areas with low concentration, such

as close to the edges of the system. A more quantitative analysis of the predicted diffusion and concentration is presented in the lower left corner. Here we plot the Cumulative Distribution Function (CDF) of the absolute relative error of both the concentration and the diffusion constant. Note that the PINN predicts the concentration very well, with roughly 80% of the points having less than 5% error, but struggles more with the diffusion coefficient. Given that the diffusion coefficient is inferred self-consistently thorugh its role in the physics-informed part of the cost function, this is not unexpected.

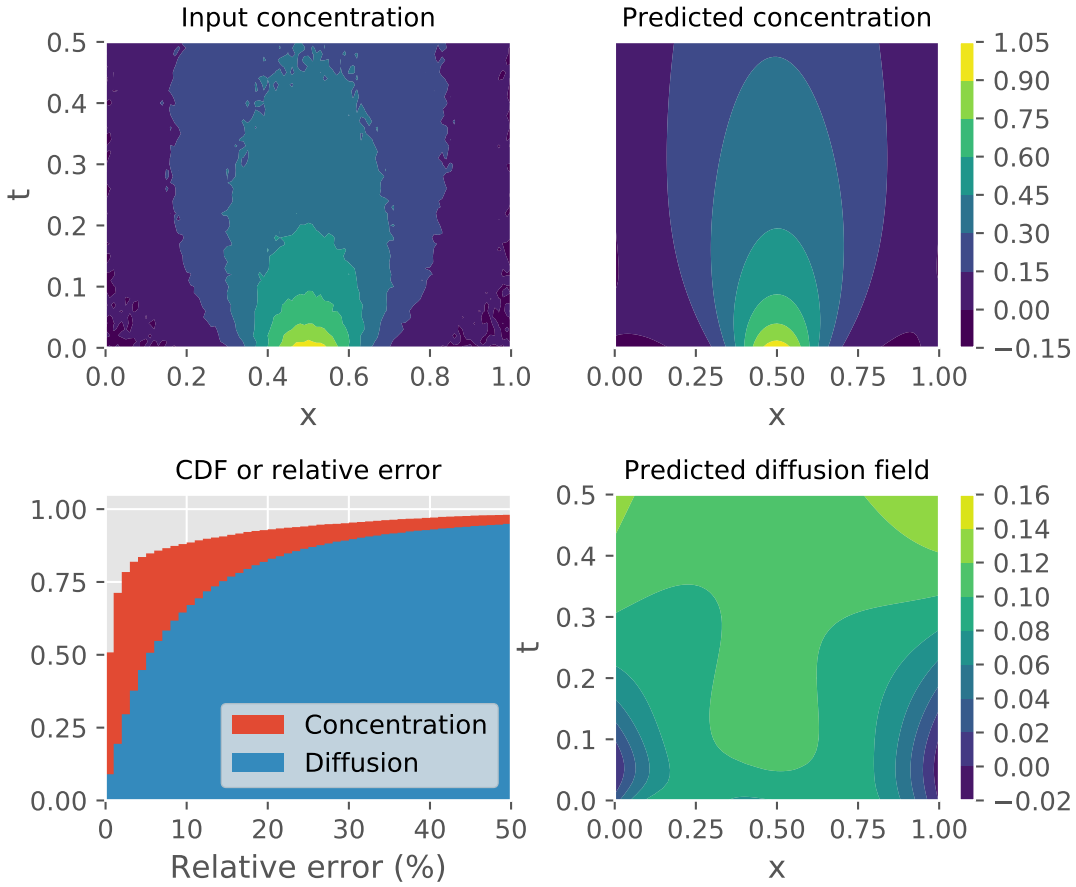


Figure 4.5: We show the training data and predicted concentration profile in the upper left and right panels. The lower right panel shows the inferred diffusion field while the lower left panel shows the CDF of the relative error of the diffusion and concentration.

In figure fig. 4.6 we show a similar analysis for data with a non-constant diffusion field. Equation eq. ?? has been numerically solved on a grid consisting of 50000 points and diffu-

sion constant profile $D(x) = 0.2 + 0.1\tanh(x)$. Remarkably, the neural network is able to accurately infer the network with 85% of the diffusion field having an error of less than 10%. In figure fig. 4.7 we show the inferred diffusion profiles in more detail by projecting them along the time axis. Observe that, yet again, the error is largest where the signal is lowest. Nonetheless, we've proven that a neural network is able to accurately infer a coefficient field from noisy data.

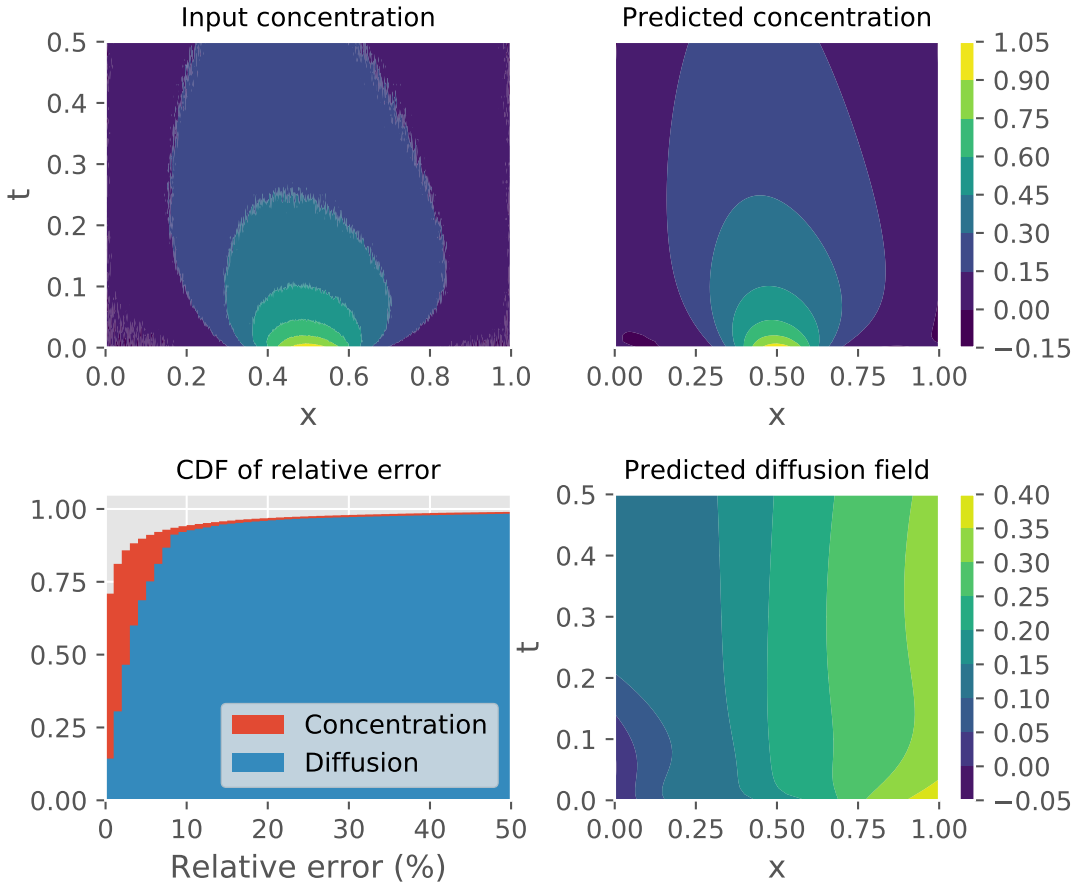


Figure 4.6: We show the training data and predicted concentration profile in the upper left and right panels. The lower right panel shows the inferred diffusion field while the lower left panel shows the CDF of the relative error of the diffusion and concentration.

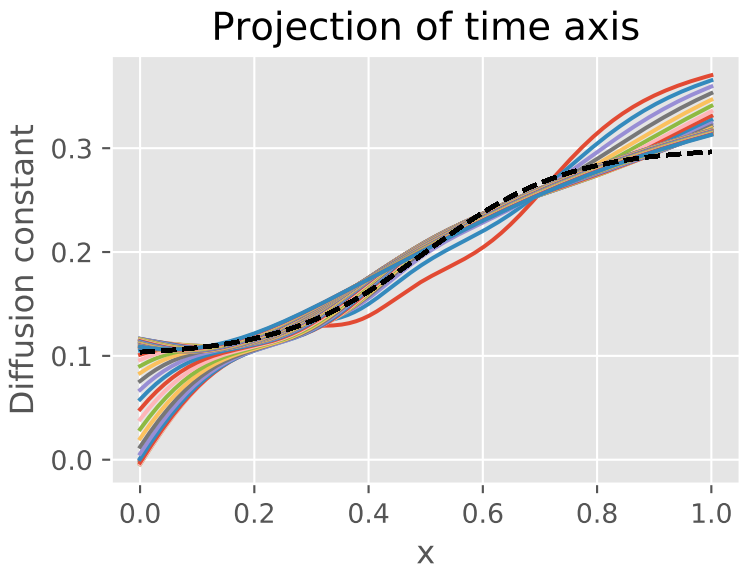


Figure 4.7: Projection of the inferred diffusion profile along the time axis.

4.2.2.3 Real cell

4.3 Conclusion

4.3.1 Weak points and how to improve

5

Golgi as a phase separated droplet

In this second part of the thesis we develop a model linking the Golgi function and size to the properties of the intracellular transport. We start with a general section on phase separation, followed by a section where we introduce an approximation. This approximation, known as the effective-droplet approximation, makes phase separation analytically tractable. We then introduce our model and its biological justification. The results of our model will be presented in the next section.

5.1 Phase separation

Consider a mixture of two molecules, type A and B, with underling interaction strengths χ_{ij}. Depending on the strength and sign of these interactions, the system is either in a mixed state with a constant concentration of A and B, or in a phase-separated state. Landau showed that instead of a complete statistical description, phase separation could be modeled by a system with a double well free energy function[47]. We can define an *order parameter* $c = N_A/N_B$ which describes the state of the system and define a free energy density function $f(c)$ with minima at c_o^- and c_o^+:

$$f(c) = \frac{b}{2(\Delta c)^2}(c - c_o^-)^2(c - c_o^+)^2$$

where b characterizes the strength of molecular interactions and $\Delta c = |c_o^- - c_o^+|$. Once the system phase separates, the system will have two areas of concentration c_o^+ and c_o^- with

a boundary inbetween. Associated with this boundary is a surface tension, so that the full free energy of the system becomes

$$F(c) = \int dV(f(c) + \frac{1}{2}\kappa(\nabla c)^2$$

To find the equilibrium concentration profile, we minimize this free energy:

$$\frac{\delta F}{\delta c} = f'(c) - k\nabla^2 c = \mu(x) = 0 \tag{5.1}$$

where $\delta F/\delta c$ is a functional derivative, as we minimize with respect to the concentration *profile*. Solving such an equation is generally not possible due to the third order terms of the free energy density function, but in 1D equation eq. 5.1 has been solved to yield:

$$c(x) = \frac{c_o^- + c_o^+}{2} + \frac{c_o^+ - c_o^-}{2} \tanh\left(\frac{x}{w}\right) \tag{5.2}$$

where $w = \sqrt{\kappa/b}$ is the width of the boundary. When we quench a system from a mixed stated into a phase separated state (a process known as spinodal decomposition), the actual concentration profile is a far cry from equation eq. 5.2. Inside the system, maze-like domains form and keep growing untill a single dense and dilute phase are left. This is shown in figure fig. 5.1.

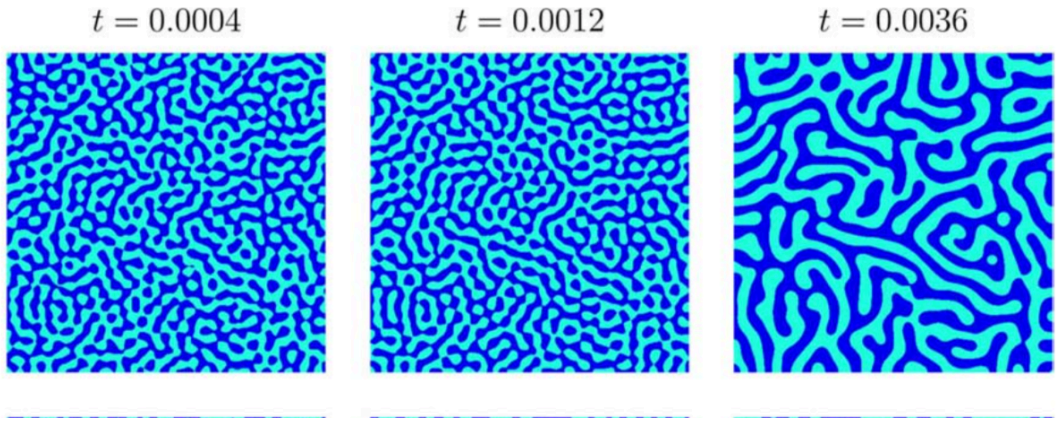


Figure 5.1: Cahn hilliard domains

In this process, the dynamics need to be taken explicitly into account. In the case of liquid-liquid phase separation, the order parameter c is conserved, as a molecule of type A cannot

change into type B. This means that the order parameter can only exchange locally, so that:

$$\partial_t c = -\nabla \cdot \mathrm{j}$$

where j is a flux. We can relate the flux to the chemical potential:

$$j = -m\nabla\mu$$

where m is a coefficient characterizing the mobility. Equation eq. 5.1 also gives us an expression for the chemical potential, so that we finally obtain the *Cahn-Hilliard equation*:

$$\frac{\partial c}{\partial t} = m\nabla^2[f'(c) - k\nabla^2 c]$$

It is this equation which governs the behaviour observed in figure fig. 5.1. Due to its nonlinearity and fourth order derivatives simply solving the Cahn-Hilliard is usually forsaken in favour of deriving a scaling relation, which relates the domain growth speed dR/dt to the domain size R or some other system parameters. Another option is to study the system in the so-called effective droplet approximation, as we do in the next section.

5.2 Effective droplet

Consider again equation eq. 5.2. If $w \ll 1$, we can approximate the system by describing it as two bulk phases, separated by an interface. If we apply correct boundary conditions at the interface, we can essentially split the system into two separate problems for the dense and dilute phase and match them at the interface. We thus model the phase separated system as a droplet which exchanges material with its environment through its interface, as shown in figure fig. 5.2

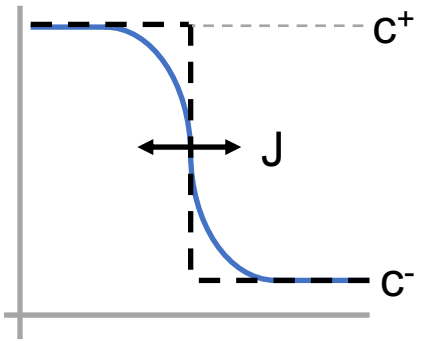


Figure 5.2: Model of an effective droplet. Blue line is full Cahn-Hilliard, black dashed line effective droplet.

By assuming the interface to be at thermodynamic equilibrium, the growth of the droplet is described in terms of the fluxes across the interface. Consider again equation eq. 5.1. As we can neglect the interfacial term, we have $\mu = f'(c)$. In each bulk phase, we linearize the chemical potential around it's equilibrium density (c_o^- or c_o^+) to yield:

$$\frac{\partial c}{\partial t} = D\nabla^2 c \tag{5.3}$$

where we've absorbed all the coefficients into a single coefficient D. Observe that we have obtained a diffusion equation. Since we now have a linear equation in c, we can analytically solve this. Moreover, it is possible to add additional effects such as decay, production or advection and still keep an analytically solvable model, as we show in the next section and chapter.

By introducing the interface and breaking the problem into two separate parts, we need two new boundary conditions to solve equations such as eq. 5.3. By assuming the interface to be at thermodynamic equilibrium, we will derive a set of boundary conditions independent of the position, size or kinetic parameters of the droplet. Consider such a phase-separated system with an infinitely thin interface. The total free energy of the system can then be written as:

$$F = V_1 f(\varphi_1) + V_2 f(\varphi_2)$$

where V_i and φ_i are respectively the volume and concentration of phase i and $f(\varphi_i)$ is the free energy density. Assuming incompressibility ($V_1 + V_2 = V$) and conservation of particles ($V_1\varphi_1 + V_2\varphi_2 = V\varphi$) constrains the system to two free variables, so that minimizing the free energy with respect to φ_1 and V_1 gives us two conditions:

$$f'(\varphi_1) = f'(\varphi_2)$$

$$0 = f(\varphi_1) + f(\varphi_2) + (\varphi_2 - \varphi_1)f'(\varphi_2)$$

Since $f'(\varphi) = \mu(\varphi)$, the first condition states that both phases must have the same chemical potential, while the second one states that both phases must have equal pressure. The obvious solution to these equations is a completely mixed state with $\varphi_1 = \varphi_2$. A non-trivial phase-separated solution exists as well, where φ_1 and φ_2 are the two minima of the free energy density function $f(\varphi)$. Note that this valid for droplets in 1D. In higher dimensions, the curvature of the droplet will affect the boundary conditions due to the Laplace pressure, but one can show that this leads to an extra term which scales with $1/R$[48].

Having defined boundary conditions, equations such as eq. 5.3 can be solved. Although one could solve these equations fully time-dependent using Green's functions, we assume a quasi-steady state so that $dc/dt = 0$.This will give us the concentration profiles in and outside the droplet, droplet growth however is determined by the fluxes across the interface. We show this in the next section.

5.2.1 Fluxes, activity and interfaces

Given a concentration profile $c(x)$, a diffusive flux can be calculated by applying Ficks' second law:

$$J(x) = -D\frac{\partial c}{\partial x}$$

Using this expression, the flux at the interface on the inside and outside of the droplet, J_{in} and J_{out}, can be calculated. Note that in and out respectively refer to inside and outside of the droplet rather than the direction of the flux; our boundary conditions fix the concentration at the interface but not the (direction of the) fluxes. If these fluxes are not balanced, there exists a net flux across the interface which leads to either growth or decay of the droplet. As the droplet changes size, the interface moves with a speed v_n. We derive an expression for v_n in terms of the fluxes across it. To move the interface a distance Δx, a net material gain of $\Delta x \Delta c$ is needed. This net gain is given by the net flux in a time Δt, so that:

$$\Delta x \Delta c = (J_{in} - J_{out})\Delta t$$

which can be rewritten as:

$$\frac{\Delta x}{\Delta t} = v_n = \frac{J_{in} - J_{out}}{\Delta c} \tag{5.4}$$

In the passive case (and assuming quasi-steady state), the concentration both inside and outside the droplet would be described by a solution to laplace's equation (i.e. $\nabla^2 c = 0$), leading to a flat concentration profile $c_{out}(x) = c_\circ^-$, $c_{in}(x) = c_\circ^+$ and hence $J = 0$ everywhere; the system is at thermodynamical equilibrium with $\mu = 0$ and the droplet doesn't change size as $v_n = 0$. Placing such a droplet in a supersaturated environment where $c_{out}(x) > c_\circ^-$ would lead to a lead to a non-zero J_{out}, resulting in the droplet growing to infinity by a process known as Ostwald ripening. As we show in the next section, active droplet supress the Ostwald ripening[49], leading to a droplet with a finite radius.

We now make the droplet *active* by adding a chemical reaction in the droplet which decays the droplet material A into some other other material B. Assuming material B diffuses very fast and is thus always in equilibrium, we ignore material B and describe solely $\mathcal{A}$. Adding a decay term to equation eq. 5.3 gives:

$$D\nabla^2 c - kc = 0 \tag{5.5}$$

where k is a decay constant. Solving equation eq. 5.5 will always give a convex solution and hence a finite J_{in}. A typical concentration profile for an active droplet is shown in figure fig. 5.3.

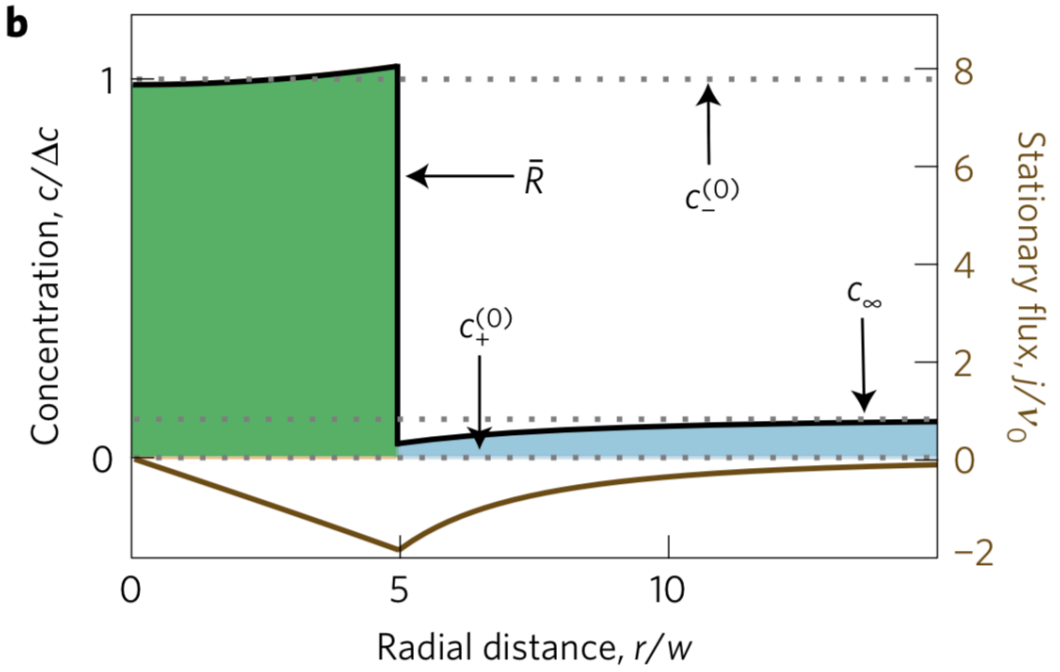


Figure 5.3: Typical concentration profile of active droplet. Taken from 17

The 'decay flux' J_{in} will need to balanced by a flux into the droplet J_{out} for a stable droplet to exist. This is a key property of active systems: while the system is at steady state (i.e. $v_n = 0$), it is not at thermodynamical equilibrium, as $\mu \neq 0$ and $J \neq 0$. Another fascinating property of active droplets is that they can propel themselves. To see this, consider a droplet of radius R at position x_0 with two interfaces moving respectively at v_l and v_r. In a time dt, the droplet moves to a new postion $x_0 + dx$ and will have a new radius $R + dR$:

$$x_0 - R + v_l dt = x_0 + dx - (R + dR)$$

$$x_0 + R + v_r dt = x_0 + dx + (R + dR)$$

Solving this set of equations for dx and dR gives:

$$\frac{dR}{dt} = \frac{1}{2\Delta c}(v_r - v_l) \tag{5.6}$$

$$\frac{dx_0}{dt} = \frac{1}{2\Delta c}(v_l + v_r) \tag{5.7}$$

Combining these equations with equation eq. 5.4 finally relates the growth and movement of the droplet to the fluxes across the interface:

$$\frac{dR}{dt} = \frac{1}{2\Delta c}\left[(J_{in}^{x=R} - J_{in}^{x=-R}) + (J_{out}^{x=-R} - J_{out}^{x=R})\right]$$

$$\frac{dx_0}{dt} = \frac{1}{2\Delta c}\left[(J_{in}^{x=-R} + J_{in}^{x=R}) - (J_{out}^{x=-R} + J_{out}^{x=R})\right]$$

Studying equations eq. 5.6 and eq. 5.7 shows that movement only happens if the fluxes are asymmetrical; the droplet center is displaced because one side of the droplet grows faster than the other. This imbalance is caused by a concentration gradient and thus *droplets will move up the gradient*, as the flux on the high concentration side will be higher than on the low side.

These equations allow us to find steady states both with respect to the size and the position of the droplet. In the next section we adapt the equations to model the Golgi and couple it to the intracellular transport through the calculations of the fluxes J_{out}. We do so in the next section.

5.3 Golgi as an active droplet

In the introduction we justified using a phase-separation approach to describe the Golgi. In this section we develop our model for the Golgi from biological considerations, but having established the mathematical background of phase separation, we parallely present the mathematical description.

We can recognize four different populations in our system: immature cargo -heading to the Golgi-, mature cargo, -originating from the Golgi and which is produced from immature cargo in the golgi-, the Golgi itself and the cytoplasm, which acts as the solute. We start by reducing this set of populations to a system described by a single concentration c. Assuming maturation from the Golgi as a oneway process, i.e. immature cargo turns into mature cargo but not otherwise, and no interaction between the mature and immature cargo during intracellular transport, we can ignore the mature cargo. Modeling the solvent implicitly, the immature cargo in the cytoplasm is then represented by a dilute phase in some concentration c, while the Golgi is described by a dense phase in the same concentration.

Upon adding the drug nocadazole to mammalian cells, the microtubules are depolymerized and the Golgi ribbon breaks up into separate stacks[50] . These stacks are fully functional[7] and move away from their perinuclear location to colocate with an ERES. If we model not

the complete Golgi but a single stack, we can reduce our problem to 1D, where a droplet can move from one side of the system, representing the Golgi ribbon, to the other side, representing the ERES. As each stack is fully functional, we make no simplifications with respect to the function of the Golgi. Although many complex models of the maturation exist, we model it as a simple decay-like term:

$$\frac{\partial c}{\partial t} = -\nabla J - kc$$
$$J = -D\nabla c$$

where k is a maturation constant. We now turn our attention to the intracellular transport. In our model fitting chapter we presented an argument that we could model the intracellular transport as an advection-diffusion equation. Evidence exists of vesicles refusing with the ER[8], so we add an additional decay term, so that the concentration outside the droplet is described is described by:

$$D\partial_x^2 c(x) - v\partial_x c(x) - ac(x) = 0 \tag{5.8}$$

with v an advection velocity and a some decay constant. Solving this equation will lead to a concentrion profile with a gradient and we thus model our golgi as an active droplet growing in a concentration gradient, inspired by 51. A study of the biogenesis of the golgi[8] shows that stacks are transported to the ribbon over the microtubules, so we thus add the advection also to the dense phase:

$$D\partial_x^2 c(x) - v\partial_x c(x) - kc(x) = 0 \tag{5.9}$$

Note that the dense and dilute phase description are thus almost similar, save for a different decay (maturation) constant. One could pick different diffusion constants and advection speeds, but for simplicity and without loss of generality we pick the same. As our free energy function has minima at c_0^+ and c_0^-, our boundary conditions at the interface are:

$$c(x_0 \pm R) = \begin{cases} c_0^+, & \text{inside} \\ c_0^-, & \text{outside} \end{cases}$$

As stated, we model our system in 1D, with one boundary representing the ERES and the other boundary as the location of Golgi Ribbon. We place the ERES on the left side of the system and thus model this boundary as source:

$$(-D\partial_x c + vc)|_{x=0} = J_{in}$$

whereas the right boundary is merely the edge of the system and we thus model it as a zero-flux boundary:

$$(-D\partial_x c + vc)|_{x=L} = 0$$

We solve this set of equations in the next chapter.

6

Results model

The previous chapter introduced phase separation and our model for the Golgi as a phase-separated droplet. In this chapter we solve the equations and present the results and biological implications. The first section solves the model and presents the calculated fluxes through the interface. It also introduces several lengthscales and dimensionless parameters of the problem. As adding advection to an active droplet is a novel approach, we study the affect of advection in the second section. We present the full active-droplet phase diagram in the last section and discuss the biological implications.

6.0.1 Solving the model

The first step is to find a general solution to equations eq. 5.8 and eq. 5.9. As they are of the same form, the general solution is given by:

$$c(x) = C_1 e^{-\frac{x}{l^-}} + C_2 e^{\frac{x}{l^+}} \tag{6.1}$$

where we have defined $l^{\pm}$ as:

$$l^{\pm} = \frac{2D}{\sqrt{4kD + v^2} \pm v} \tag{6.2}$$

where k should be replaced by a for the concentration in the dilute phase. Note that $l^{\pm}$ defines the lengthscales of the problem and that due to addition of advection there are two;

without advection it simplifies to $\sqrt{D/k}$. It is here we see that the advection leads to some sort of 'symmetry-breaking' in the droplet and that the effect of advection is more than just moving the droplet. A typical concentration profile for a droplet is shown in figure FIGURE. We can associate l^+ with the right side of the well and l^- with the left side. We study the effect of advection in the next section. Applying the boundary conditions and calculating the flux at position $x_\circ + R$ and $x_\circ - R$ gives the following fluxes outside:

$$J_{out}^{x=-R} = J_{in}\frac{(1+\frac{l_-}{l_+})e^{\frac{-(x_1)}{l_-}}}{Pe_- + Pe_+\frac{l_-}{l_+}e^{\frac{-x_1}{l}}} + \frac{c_\circ^{out}D}{l_+}\frac{Pe_+(1-e^{\frac{-x_1}{l}})}{1+\frac{l_-}{l_+}\frac{Pe_+}{Pe_-}e^{\frac{-x_1}{l}}} \tag{6.3}$$

$$J_{out}^{x=R} = -c_\circ^{out}D\frac{Pe_-Pe_+(1-e^{\frac{-x_2+L}{l}})}{l_+Pe_- + e^{\frac{-x_2+L}{l}}l_-Pe_+} \tag{6.4}$$

where we have introduced the coordinates x_1 and x_2, which are defined respectively as $x_\circ \pm R$ and correspond to the position of the left and right interface. We have also defined the Peclet-like numbers $Pe^\pm$:

$$Pe^\pm = 1 \mp \frac{vl^\pm}{D}$$

and a new combined lengthscale $l = \frac{l^+l^-}{l^++l^-} = 1/l^+ + 1/l^-$. The flux on the left of the droplet consists of two terms, with the first one accounting for the influx and the second one for the interface with the droplet. The flux on the right is clearly similar, but lacks a second term since we've set a zero-flux boundary at $x = L$.

We now turn to the fluxes on the inside of the droplet. These fluxes separately are not particularly insightful, but, considering equations eq. 5.6 and eq. 5.7, we can study their sum and difference. Introducing $J_{rad} = J_{in}^{x=R} - J_{in}^{x=-R}$ and $J_{pos} = J_{in}^{x=R} + J_{in}^{x=-R}$, we obain:

$$J_{rad} = \frac{-2c_\circ^+D}{l}\frac{\sinh\frac{R}{l^-}\sinh\frac{R}{l^+}}{\sinh\frac{R}{l}}$$

$$J_{pos} = 2c_\circ^{in}D\left[\frac{Pe_-}{l_-}\frac{\sinh\frac{R}{l_+}\cosh\frac{R}{l_-}}{\sinh\frac{R}{l}} - \frac{Pe_+}{l_+}\frac{\sinh\frac{R}{l_-}\cosh\frac{R}{l_+}}{\sinh\frac{R}{l}}\right]$$

The flux J_{rad} is the total flux lost due to maturation, while the flux J_{pos} is the difference in fluxes due to maturation on both sides of the droplet, which is clearly visible in its symmetric construction. J_{rad} contains an important point about the validity of effective droplet

theory. Consider it in the limit of $R \to \infty$:

$$\lim_{R\to\infty} = -2c_o^+\sqrt{kD} \tag{6.5}$$

where for simplicity we have neglected advection. As we have included a maturation term, we would expect the flux due to maturation to scale with R, which indeed for small R it does. However, equation eq. 6.5 tells us that the maturation flux saturates. Consider again equation eq. ?? and note that inside the droplet the concentration is convex and decays on a lengthscale l. If $R \gg l$, the concentration in the centre of the droplet will go to zero and hence the maturation flux will saturate. Thus, effective droplet theory is only valid in the region for $R < l$. When we include advection the point is slighly more subtle as advection changes the internal concentration profile of the droplet. We investigate this in the next section.

6.1 Free droplet

6.1.1 No decay - the effect of advection

In this section we investigate the effect of advection on an active droplet. Naively, one would expect advection to move the droplet with a velocity v, and indeed, for a passive droplet this happens. As the concentration profile is flat inside a passive droplet, the only non-zero flux inside the droplet is due to advection: $J_{adv} = c_o^+ v$, while outside we have $J_{adv} = c_o^- v$, thus giving $dR/dt = 0, dx_o/dt = v$. In an active droplet the situation is slight more complex due to the convexity of the concentration profile. The diffusive fluxes point inwards from the interfaces; they're equal in magnitude, but aligned antiparallel. The advective fluxes at the interfaces however are aligned parallelly. The net flux at both interfaces is thus different, as shown in figure FIGURE. We now quantify this effect.

Figure about fluxes in droplet

For simplicity, but without loss of generality, we neglect decay outside the droplet, so that $a = 0$. The flux then becomes location independent:

$$J_{out}^{x=-R} = J_{in}$$

$$J_{out}^{x=R} = 0$$

The flux on the right interface of the droplet is zero as there is no source nor decay on the right side of the droplet and in a quasi-steady state approximation the flux then must become zero.The equations for the flux in the droplet remain unchanged as they are independent of the transport parameters. Developing the internal droplet fluxes for $R \ll l^{\pm}$ gives:

$$J_{rad} \approx -2c_{\circ}^{+} kR \left(1 - \frac{kR^2}{3D}\right) \tag{6.6}$$

$$J_{pos} \approx 2c_{\circ}^{+} v \left(1 - \frac{kR^2}{3D}\right) \tag{6.7}$$

Observe that the positional flux is modified by the same factor as the maturation flux and that due to the activity, the positional flux is a factor $kR^2/3D$ smaller. To see why activity reduces the positional flux, first consider the concentration profile inside the a non-convected active droplet. In this case, the lengthscales l_- and l_+ are equal and the droplet is 'symmetric' around the middle of the droplet. Turning on the advection, this symmetry is broken and the lengthscales l_- and l_+ change. Specifically, as can be seen from equation eq. 6.1, we can associate l_- with the left of the droplet and l_+ with the right side of the droplet. Plotting these lengthscales in figure fig. ??, we observe that l_+ decreases with v, while l_- increases; the location of the minimum concentration is thus shifted in the directio of the advection. This means that the concentration profile on that side is steeper and hence has a bigger diffusive flux, as shown in FIGURE, leading to the decrease in the positional flux.

Ignoring the third order term in eq. 6.6, we have the following equations for the radius and the position of the droplet:

$$\frac{dR}{dt} = (-2c_{\circ}^{+} kR + J_{in})/2\Delta c$$

$$\frac{dx_{\circ}}{dt} = (2c_{\circ}^{+} v - J_{in})/2\Delta c$$

As we are interested in stable droplets, we have $dR/dt = 0$, so that $R_{stable} = J_{in}/2c_{\circ}^{+}k$. For the positional change, we have $v = J_{in}/2c_{\circ}^{+}$. This means that a finite velocity is required for the droplet to change direction. To see why, we note that $dx_{\circ}(v = 0)/dt = -J_{in}/2\Delta c$ and since J_{in} sets the gradient when no advection is present, we see eq. 5.7 in action: the droplet moves up the gradient. Thus, for the droplet to change direction of movement, the advection needs to cancel out the movement due to the gradient and hence a finite advection is required.

We now study this system numerically. We're most interested in the behaviour of the droplet as a function of v and J_{in} and thus plot in figure eq. 6.1 the stable radius in the left panel and the minimum concentration in the droplet in the right panel as a function of J_{in} and v.

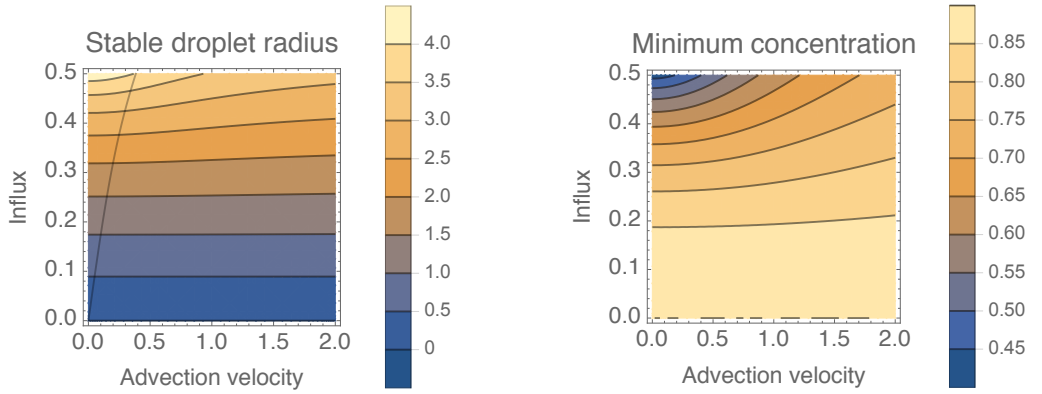


Figure 6.1: Plotting in mathematica is terrible.

We see that for low v the radius is independent of the velocity, but that for higher v the radius decreases. Concurrently, in the right panel we observe that the minimum concentration in the droplet decreases with increasing J, but decreases with v. The reason is that for increasing J, the droplet grows and the concentration inside the droplet thus decreases. We study this in detail in figure fig. 6.2, where we plot the radius and minimum concentration of the line $J = 0.25$.

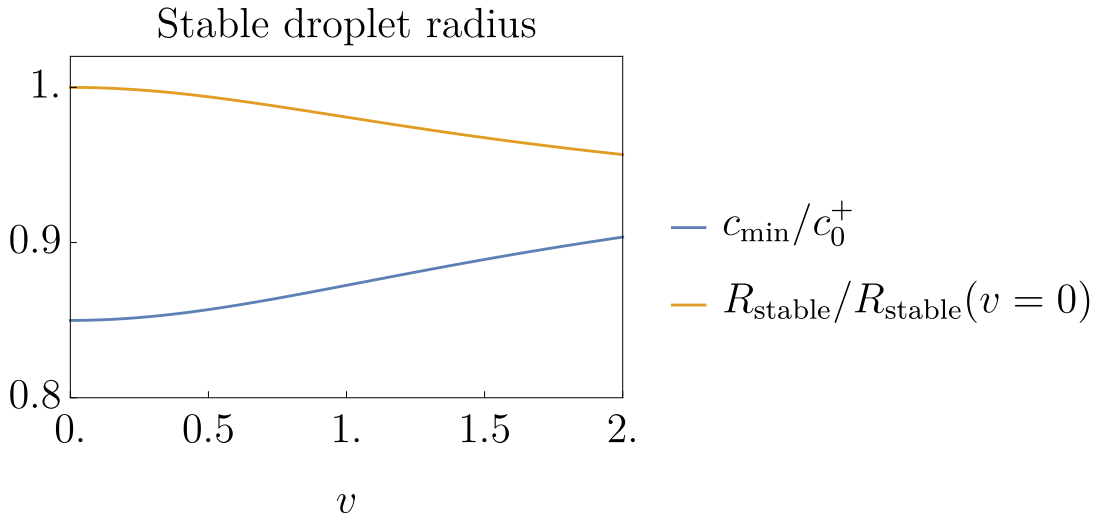


Figure 6.2: Effect of advection.

Observe that advection decreases the droplet radius by almost 5%, while simultaneously also increasing the minimum concentration. The advection is thus 'compacting' the droplet. This compacting is also the cause of the increased minimum concentration. The compaction is again a consequence of the fluxes; since the fluxes are aligned on the side facing the advection, this side experiences a higher flux than the interface on the other side, hence compacting the droplet.

How stable is the equilibrium droplet? Perturbing dR/dt around the equilibrium R gives $d\delta r/dt = -2c_\circ^+ k\delta r$. Since both k and $c_\circ^+$ are bigger than zero, any fluctutations cancel so that the equilibrium is stable. The blue line is $dx_\circ/dt = 0$. As expected, increasing the J_{in} increases the gradient and a larger velocity is required to counteract the movement of the droplet up this flow. These positions where both the droplet radius and droplet position are stable are not specific points inside the system; rather they are a specific set of parameters for which the droplet speed becomes zero.

6.1.2

We now wish to find a model in which the position has a stable position. So not just a set of parameters where $dx_\circ/dt = 0$, but a position $x_\circ$ where the droplet is stable. The flux on the inside is independent of the position of the droplet, so to get a stable position we need a position dependent flux on the outside. To this end, we introduce a decay term in the

dilute phase. We then end up with the full equations ... . We first numerically solve these equations before attempting an approximation. The result is shown in figure ... and shows a 'phase-diagram' of our active droplet model. In the left panel we show the minimum concentration in the droplet, while the middle and right panel shows the numerical value of $dx_\circ/dt$ and d_R/dt, all as a function of the CoM position $x_\circ$ and radius R. From the right two plots we've extracted the lines $dx_\circ/dt = 0$ and $dR/dt = 0$, which we've superimposed on all three panels. Where these lines cross a stable state exist. We first discuss each line independently and then discuss the stable points.

First consider the line $dR/dt = 0$, a droplet with stable radius. We observe that left of the line $dRdt > 0$, with the opposite on the right. This means that each radius is stable: a perturbation w.r.t to the radius will not propagate. The $dx_\circ dt = 0$ is more interesting. It has the apperance of a 'finger' and has a stable and unstable branch. The arm below the point is stable, where the arm above the point is not. This has important implications for the crossings: the lower intersection is stable w.r.t. to all perturbations, while the upper intersection is unstable. Pertubations above the dxodt line will propagat, while perturbations below will move the droplet to the lower intersection.

This is all fine and dandy, but there's one problem: the lower intersection, the stable point, is unphysical. We characterize the droplet in terms of its center of mass $x_\circ$ and radius R, meaning that the left interface is at a position $x_\circ - R$. The red line in the plots shows the line $x_\circ = R$ and everything below this line is unphysical, as it means the left interface is at a position $x_\circ - R <$, past the system edge. Why does this happen? First, in a dynamical description this wouldn't happen, as the left interface boundary condition prevents moving past. In a static description this can however since we specify the edge of the system as a flux boundary condition. Inspection of the concentration (see figure) learns that the droplet moves on top of the source: the flux inside the droplet at a point is similar to the boundary condition. In 2D this wouldnt be a problem but in our 1D description it is as the BC is specified as a flux and not as a source. That means we're left with just an unstable point. Why is this point unstable? Investigating the fluxes shows that despite the decay the left interface is an order of magnitude bigger than the right interface, thus behaviour is mainly determined by the left interface. Once the left interface starts moving due to a perturbation, the rest follows.

6.2 Droplet stuck to walls

6.3 Conclusion

6.3.1 Biological implications

We now make some very speculative biological connection to the observation of Golgi properties. A strong marker is that once the microtubules are depolymerized, the stacks move away from the perinuclear area and colocate with the ERES. This is what we see when we turn of the advection, the droplet moves up the gradient to the source. A similar thing is seen with biogenesis, the stacks are made around the ERES but need microtubules to be transported to the center.

7

Conclusion

So what can we say about our project? We tried a lot but nothing worked ha-ha.

Appendix 1: Some extra stuff

Add appendix 1 here. Vivamus hendrerit rhoncus interdum. Sed ullamcorper et augue at porta. Suspendisse facilisis imperdiet urna, eu pellentesque purus suscipit in. Integer dignissim mattis ex aliquam blandit. Curabitur lobortis quam varius turpis ultrices egestas.

8

References

9

Phase diagram

Gert-Jan Both - 12/11/2018

9.1 Free energy

Consider the double well free energy with minima at c_o^- and c_o^+:

$$f(c) = \frac{b}{2\Delta c^2}(c - c_o^-)^2(c - c_o^+)^2$$

A system is a thermal equilibrium when $\mu = df/dc = 0$ and a state is stable if $d^2f/dc^2 > 0$. If $d^2f/dc^2 < 0$, the state is unstable as fluctuations keep growing; the system phase separates. The points at which $d^2f/dc^2 = 0$ is called the spinodal/inflection points and the area between the spinodal phase separates. Plotting the free energy density, we see that the middle hill between the points $c = (c_o^+ + c_o^-)/2 \pm \sqrt{3}(c_o^+ - c_o^-)$ is unstable.

What does this mean for our system? The effect of the advection is to change the concentration; the dilute phase becomes denser while the dense phase becomes more dilute. Once the concentration reaches the inflection points, the dilute phase will phase separate, hence the occurence of a droplet, while once the dense phase goes past the inflection point, it will introduce a dilute phase; we get another interface. In the next section we investigate when this happens.

9.2 Advection-Diffusion: when does phase separation happen?

In the previous section we've shown that once the concentration reaches one of the inflection points, the system will phase separate. We now study when exactly this happens. We thus study an advection-diffusion-decay system with a source on one side and a no-flux boundary on the left. The highest concentrations will

then always be at the edges of the system, so we determine the concentrations at each side of the system. We're most interested in the influx J_{in} and advection speed v, so we determine for v at which J_{in} the concentration on the left and right reaches the inflection point.

$$\text{Left: } J_{in} = -\frac{2al((1 - \frac{1}{\sqrt{3}}c_o^+) + (1 + \frac{1}{\sqrt{3}}c_o^-))}{\frac{vl}{D} - coth(\frac{L}{2l})}$$

$$\text{Right: } J_{in} = al((1 - \frac{1}{\sqrt{3}}c_o^+) + (1 + \frac{1}{\sqrt{3}}c_o^-))\left(e^{-\frac{L}{2l}(\frac{vl}{D}-1)} - e^{-\frac{L}{2l}(\frac{vl}{D}+1)}\right)$$

Where l is some lengthscale $l = D/\sqrt{4aD + v^2}$. We plot these two curves in the figure below:

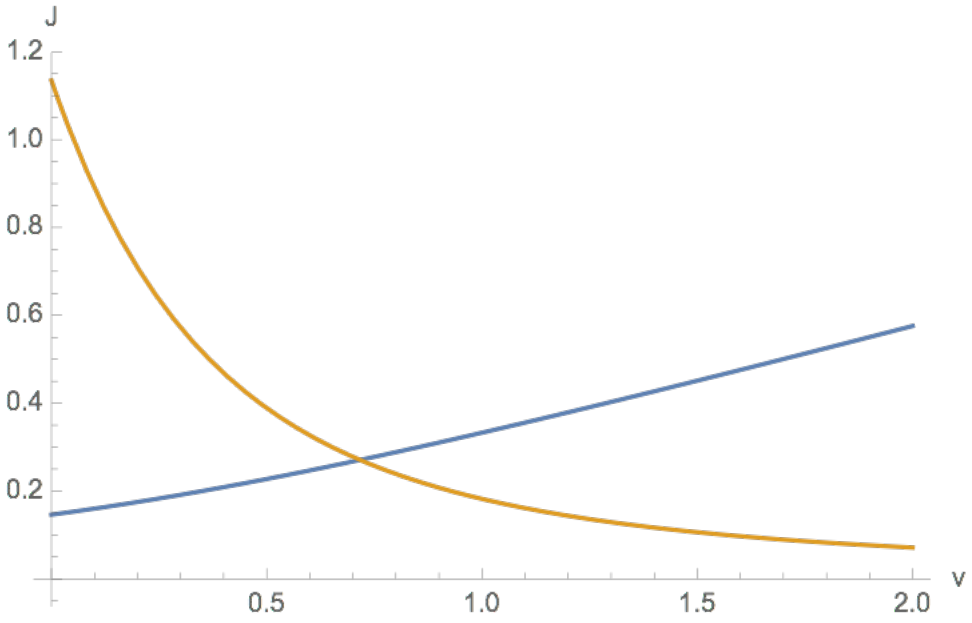


Figure 9.1: Blue line: Minimum J left side. Orange line: minimum J right side.

We can recognize four areas in this plot. Below both the blue and the orange line, the concentration doesn't become high enough anywhere and no phase separation will happen. Below the orange line and above the blue line, on the left, only a droplet on the left is stable, while on the right exactly the reverse; only the right is stable. Above both lines the concentration is high enough on both sides. This plot onyl tells us when phase-separation should happen; it doesn't tell if it does. We investigate this in the next section.

Where do the two lines cross? We expand the minimal J's to second order in v around 0, as the plot above shows that doing it in first order will severely underestimate the crosspoint. The obtained expressions are then again linearized to first order in a, yielding:

$$v^* = \frac{aL}{2}$$

which, compared to the plot, is remarkably close.

9.2.1 Effective droplet

We now need to prove that indeed a stable droplet exists if phase separation happens according to the plot above.We thus investigate at which parameters a droplet with $R = 0$ and $dR/dt|_{R=0} = 0$ exists. For the left and right droplet, we find the following relations:

$$\text{Left: } J_{in} = -\frac{2ac_0^- l}{\frac{vl}{D} - coth(\frac{L}{2l})}$$

$$\text{Right: } J_{in} = ac_0^- l\left(e^{-\frac{L}{2l}(\frac{vl}{D}-1)} - e^{-\frac{L}{2l}(\frac{vl}{D}+1)}\right)$$

where l is some lengthscale $l = D/\sqrt{4aD + v^2}$. We now compare this to the 'phase-diagram' we have obtained from studying the flow equation. By dividing the minimum flux from the flow model by the minimum flux obtained from the effective droplet model, we obtain:

$$\frac{J^{AD}}{J^{ED}} = \frac{(3-\sqrt{3})c_0^+ + (3+\sqrt{3})c_0^-}{6c_0^-}$$

For both the droplet on the left and right. We see that $J^{AD}/J^{ED} > 1$ if $c_0^+ > c_0^-$, which it is by definition. In other words, if dilute concentration reaches the inflection point in the AD model, a stable effective droplet is guaranteed to exist. It also means that once phase separation occurs to the AD model, the effective droplet predicts a non-zero radius. Although this is dynamics and we do statics so there's nothing more we can really say about it. We also have:

$$\frac{J^{AD}_{left}}{J^{AD}_{right}} = \frac{J^{ED}_{left}}{J^{ED}_{right}}$$

meaning that the velocity at which the two left and right minima cross is the same for the both models. The implication of this is that the difference between when the phase-separation happens according the AD model and the ED model is due to the difference in concentration. As we're looking for a steady state description, we continue with the AD model. According to our calculations, there is an area where both droplets can exist at the same time. We investigate this using a model with two droplets. Performing a similar analysis (i.e. dR1/dt=dR2dt=0, R1=R2=0) yields:

$$J = \frac{c_0^- D}{2l}\left(\frac{vl}{D} + \frac{(1 + e^{\frac{L}{l}} - 2e^{\frac{-L}{2l}(\frac{vl}{D}-1)})}{(1-e^{\frac{L}{l}})}\right)$$

and a critical speed above which droplets appear:

$$v^* = \frac{aL}{2}$$

This critical speed is similar to the crossing of the left and right droplet! This means there's some sort of critical point? As the numerics in the next section show, this a robust point and not just a first order effect.

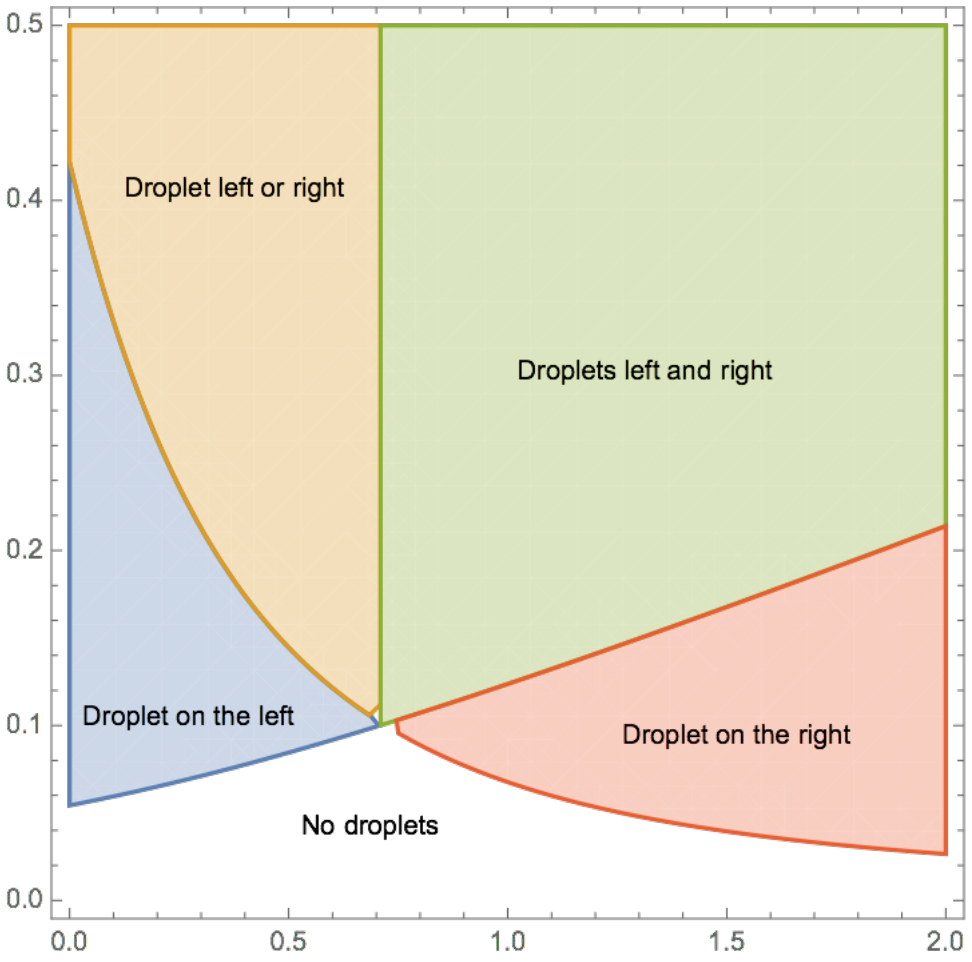


Figure 9.2: image-20181112121152157

There is a tiny sliver of left or right between the droplets on the left and right and droplets on the right, but we ignore this as its really small.

9.2.2 Mass conservation

We now study mass conservation by comparing the mass in a non-separated system with a phase separated. For simplicity we set $k \to a$. Given any concentration profile (either above or below critical concentration) given by an advection-diffusion-decay equation, the mass in the system is:

$$\int_{\circ}^{L} c(x)dx = \frac{J_{in}}{a}$$

which makes sense as J_{in} is what comes into the system and a what goes out. Now, given the same parameters J_{in}, a there exists some phase-separated configuration with stable droplet radius R^*. Without loss of generality, we assume J_{in} and a are such the droplet appears on the left side of the system. The total mass inside such a system is:

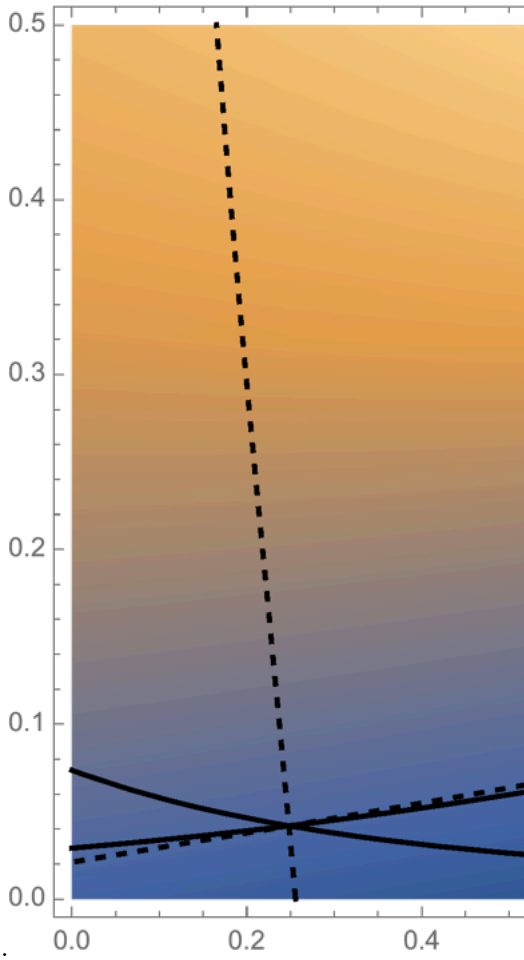
$$\int_0^{R^*} c_{in}(x)dx + \int_{R^*}^{L} c_{out}(x)dx$$

Where the stable droplet radius R^* corresponds to $dR/dt|_{R=R^*} = 0$. Assuming the droplet radius remains small, we expand both the total mass as well as the stable droplet radius in a first order Taylor series around $R = 0$, which when combined gives:

$$\int_0^{R^*} c_{in}(x)dx + \int_{R^*}^{L} c_{out}(x)dx = \frac{J_{in}}{a}$$

We thus see that mass is conserved when phase separation occurs.

9.3 Numerics



We numerically solve equations xxx for the interface speed to obtain a phase diagram:

We note that this pretty much matches our analytical work, save for the two-droplet line. The reason for this is that we've calculated where both droplets appear, which is not always true. We can also have a second droplet appear, while the first one is already there. Thus the speed v we have found is actually the v of the crossing, but analytically we're not gonna do much better...

9.3.1 Taking into account concentration

Although we have the droplet stability program, at some speeds and influxes the concentration inside the droplet becomes too low or outside too high. We plot this in the figure below. Everything to the right and above the dashed lines is unstable.

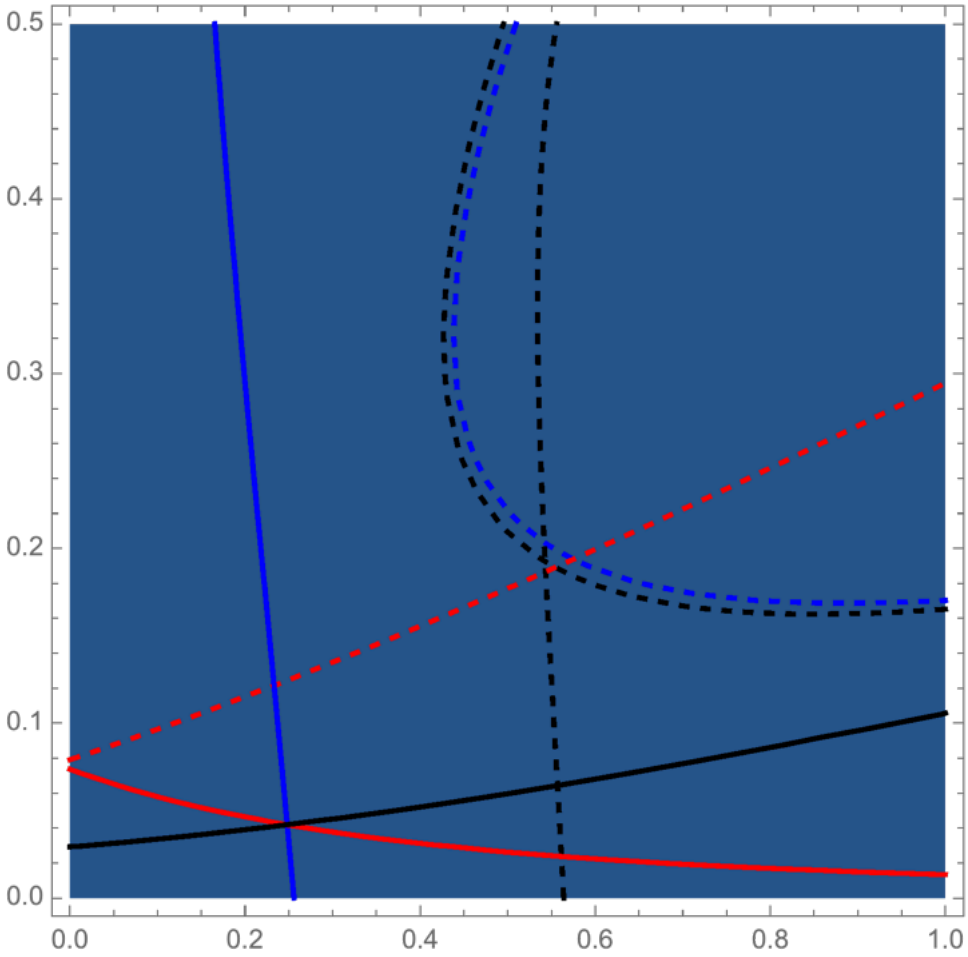


Figure 9.3: image-20181112152459201

The solid lines correspond to the interfaces from the phase diagram, while the dashed lines are the lines at which the concentration becomes too high or too low. We see that the single droplets are always stable in their domain, but not the two droplet model. Inside the blue dashes curve, the concentration in the left droplet drops below the inflection point and the droplet thus becomes 'free'. As we've seen, the free droplet will move to the right untill the other side of the system, so inside the blue dotted curve only the droplet on the right is stable.

Unfortunately, I don't see a way to get analytically nice stuff out as the inflection point is 'just' a concentration for the advection diffusion model....

1. Emr, S. *et al.* Journeys through the Golgi—taking stock in a new era. *The Journal of Cell Biology* 187, 449–453 (2009).

2. Tang, D. & Wang, Y. Cell cycle regulation of Golgi membrane dynamics. *Trends in Cell Biology* 23, 296–304 (2013).

3. Rothman, J. E. The Future of Golgi Research. *Molecular Biology of the Cell* 21, 3776–3780 (2010).

4. Gosavi, P. & Gleeson, P. A. The Function of the Golgi Ribbon Structure - An Enduring Mystery Unfolds! *BioEssays* 39, 1700063 (2017).

5. Budnik, A. & Stephens, D. J. ER exit sites - Localization and control of COPII vesicle formation. *FEBS Letters* 583, 3796–3803 (2009).

6. Bressloff, P. C. & Newby, J. M. Stochastic models of intracellular transport. *Reviews of Modern Physics* 85, 135–196 (2013).

7. Wei, J.-H. & Seemann, J. Unraveling the Golgi Ribbon. *Traffic* 11, 1391–1400 (2010).

8. Ronchi, P., Tischer, C., Acehan, D. & Pepperkok, R. Positive feedback between Golgi membranes, microtubules and ER exit sites directs de novo biogenesis of the Golgi. *Journal of Cell Science* 127, 4620–4633 (2014).

9. Newby, J. M. & Bressloff, P. C. Quasi-steady State Reduction of Molecular Motor-Based Models of Directed Intermittent Search. *Bulletin of Mathematical Biology* 72, 1840–1866 (2010).

10. Newby, J. & Bressloff, P. C. Random intermittent search and the tug-of-war model of motor-driven transport. *Journal of Statistical Mechanics: Theory and Experiment* 2010, P04014 (2010).

11. Alberts. *Molecular biology of the Cell*.

12. Staehelin, L. A. & Kang, B.-H. Nanoscale Architecture of Endoplasmic Reticulum Export Sites and of Golgi Membranes as Determined by Electron Tomography. *PLANT PHYSIOLOGY* 147, 1454–1468 (2008).

13. Griffiths, H. Rubisco is said to be both the most important enzyme on Earth and surprisingly inefficient. Yet an understanding of the reaction by which it fixes CO_2 suggests that evolution has made the best of a bad job. 2

14. Glick, B. S. & Luini, A. Models for Golgi Traffic: A Critical Assessment. *Cold Spring Harbor Perspectives in Biology* 3, a005215–a005215 (2011).

15. Hirschberg, K. *et al.* Kinetic Analysis of Secretory Protein Traffic and Characterization of Golgi to Plasma Membrane Transport Intermediates in Living Cells. *The Journal of Cell Biology* 143, 1485–1503 (1998).

16. Boncompain, G. *et al.* Synchronization of secretory protein traffic in populations of cells. *Nature Methods* 9, 493–498 (2012).

17. Zwicker, D., Seyboldt, R., Weber, C. A., Hyman, A. A. & Jülicher, F. Growth and division of active droplets provides a model for protocells. *Nature Physics* 13, 408–413 (2017).

18. Hyman, A. A., Weber, C. A. & Jülicher, F. Liquid-Liquid Phase Separation in Biology. *Annual Review of Cell and Developmental Biology* 30, 39–58 (2014).

19. Zwicker, D., Decker, M., Jaensch, S., Hyman, A. A. & Jülicher, F. Centrosomes are autocatalytic droplets of pericentriolar material organized by centrioles. *Proceedings of the National Academy of Sciences* 111, E2636–E2645 (2014).

20. Sbalzarini, I. F. Seeing Is Believing: Quantifying Is Convincing: Computational Image Analysis in Biology. in *Focus on Bio-Image Informatics* (eds. De Vos, W. H., Munck, S. & Timmermans, J.-P.) 219, 1–39 (Springer International Publishing, 2016).

21. Chen, K.-C., Qiu, M., Kovacevic, J. & Yang, G. Computational Image Modeling for Characterization and Analysis of Intracellular Cargo Transport. in *Computational Modeling of Objects Presented in Images. Fundamentals, Methods, and Applications* (eds. Hutchison, D. et al.) 8641, 292–303 (Springer International Publishing, 2014).

22. Lee, H.-C. & Yang, G. An image-based computational method for characterizing whole-cell scale spatiotemporal dynamics of intracellular transport. in *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)* 699–702 (IEEE, 2015). doi:10.1109/ISBI.2015.7163969

23. Yang, G. Bioimage informatics for understanding spatiotemporal dynamics of cellular processes: Spatiotemporal dynamics of cellular processes. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine* 5, 367–380 (2013).

24. Hebert, B., Costantino, S. & Wiseman, P. W. Spatiotemporal Image Correlation Spectroscopy (STICS) Theory, Verification, and Application to Protein Velocity Mapping in Living CHO Cells. *Biophysical Journal* 88, 3601–3614 (2005).

25. Kisley, L. *et al.* Characterization of Porous Materials by Fluorescence Correlation Spectroscopy Superresolution Optical Fluctuation Imaging. *ACS Nano* 9, 9158–9166 (2015).

26. Semrau, S. & Schmidt, T. Particle Image Correlation Spectroscopy (PICS): Retrieving Nanometer-Scale Correlations from High-Density Single-Molecule Position Data. *Biophysical Journal* 92, 613–621 (2007).

27. Raissi, M., Perdikaris, P. & Karniadakis, G. E. Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations. *arXiv:1711.10566 [cs, math, stat]* (2017).

28. Barron, J. L., Fleet, D. J. & Beauchemin, S. S. Performance of Optical Flow Techniques. 60 (1994).

29. Dong, G., Baskin, T. I. & Palaniappan, K. Motion Flow Estimation from Image Sequences with Applications to Biological Growth and Motility. in *2006 International Conference on Image Processing* 1245–1248 (IEEE, 2006). doi:10.1109/ICIP.2006.312551

30. Vig, D. K., Hamby, A. E. & Wolgemuth, C. W. On the Quantification of Cellular Velocity Fields. *Biophysical Journal* 110, 1469–1475 (2016).

31. Garcia, D. Robust smoothing of gridded data in one and higher dimensions with missing values. *Computational Statistics & Data Analysis* 54, 1167–1178 (2010).

32. Zimoń, M., Reese, J. & Emerson, D. A novel coupling of noise reduction algorithms for particle flow simulations. *Journal of Computational Physics* 321, 169–190 (2016).

33. Zimoń, M. *et al.* An evaluation of noise reduction algorithms for particle-based fluid simulations in multiscale applications. *Journal of Computational Physics* 325, 380–394 (2016).

34. Grinberg, L., Yakhot, A. & Karniadakis, G. E. Analyzing Transient Turbulence in a Stenosed Carotid Artery by Proper Orthogonal Decomposition. *Annals of Biomedical Engineering* 37, 2200–2217 (2009).

35. Grinberg, L. Proper orthogonal decomposition of atomistic flow simulations. *Journal of Computational Physics* 231, 5542–5556 (2012).

36. Bruno, O. & Hoch, D. Numerical Differentiation of Approximated Functions with Limited Order-of-Accuracy Deterioration. *SIAM Journal on Numerical Analysis* 50, 1581–1603 (2012).

37. Knowles, I. & Renka, R. J. METHODS FOR NUMERICAL DIFFERENTIATION OF NOISY DATA.

38. Rizk, A. *et al.* Segmentation and quantification of subcellular structures in fluorescence microscopy images using Squassh. *Nature Protocols* 9, 586–596 (2014).

39. Holcman, D. Modeling DNA and Virus Trafficking in the Cell Cytoplasm. *Journal of Statistical Physics* 127, 471–494 (2007).

40. Lagache, T. & Holcman, D. Effective Motion of a Virus Trafficking Inside a Biological Cell. *SIAM Journal on Applied Mathematics* 68, 1146–1167 (2008).

41. Dinh, A.-T., Theofanous, T. & Mitragotri, S. A Model for Intracellular Trafficking of Adenoviral Vectors. *Biophysical Journal* 89, 1574–1588 (2005).

42. Brandenburg, B. & Zhuang, X. Virus trafficking – learning from single-virus tracking. *Nature Reviews Microbiology* 5, 197–208 (2007).

43. Karpatne, A., Watkins, W., Read, J. & Kumar, V. Physics-guided Neural Networks (PGNN): An Application in Lake Temperature Modeling. *arXiv:1710.11431 [physics, stat]* (2017).

44. Sharma, R., Farimani, A. B., Gomes, J., Eastman, P. & Pande, V. Weakly-Supervised Deep Learning of Heat Transport via Physics Informed Loss. *arXiv:1807.11374 [cs, stat]* (2018).

45. Pun, G. P. P., Batra, R., Ramprasad, R. & Mishin, Y. Physically-informed artificial neural networks for atomistic modeling of materials. *arXiv:1808.01696 [cond-mat]* (2018).

46. Raissi, M., Perdikaris, P. & Karniadakis, G. E. Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations. *arXiv:1711.10561 [cs, math, stat]* (2017).

47. Bray, A. J. Theory of phase-ordering kinetics. *Advances in Physics* 51, 481–587 (2002).

48. Weber, C. A., Zwicker, D., Jülicher, F. & Lee, C. F. Physics of Active Emulsions. *arXiv:1806.09552 [cond-mat]* (2018).

49. Zwicker, D., Hyman, A. A. & Jülicher, F. Suppression of Ostwald ripening in active emulsions. *Physical Review E* 92, (2015).

50. Sengupta, D. & Linstedt, A. D. Control of Organelle Size: The Golgi Complex. *Annual Review of Cell and Developmental Biology* 27, 57–77 (2011).

51. Weber, C. A., Lee, C. F. & Jülicher, F. Droplet ripening in concentration gradients. *New Journal of Physics* 19, 053021 (2017).