Geometrical Models and Stochastic Geometry of Subcellular Structures

Hans Jacob Teglbjærg Stephensen

April 10, 2021

Abstract

The analysis of the geometry of objects is a fundamental property with important interpretations in most fields of natural science. As the fields of bioimaging, biology and pathology evolve, so should the computational and statistical methods which we use for the correction of imaging artifacts and for measuring the stochastic variations in the shape of geometric objects of interest.

When imaging biological structures at nanometer scales, we are at the frontier of what is currently possible. This is clearly shown by a variety of physical limitations on scale, resolution, field of view, signal-to-noise ratio which, among other things, limits the sensitivity to the presence of specific objects. Similarly, the likelihood of imaging artifacts increases. At these scales, the images are often noisy and prone to misalignment or deformations, and hence, they often require some form of correction before accurately characterising and measuring the geometry of the objects of interest. In Chapter 2, we present a realignment method using biological structures which has a stochastically known shape showing superior correction on specific types of data.

When we analyze the geometry of objects, we often want to determine if there is a geometric interdependence between objects. This problem need careful definition because a shape is commonly viewed as an intrinsic property independent of other objects. In order to address questions which arise practically, a relational shape measure should be affected both by the spatial distance as well as by an object to object shape dependence. In Chapter 3, we handle all of these problems by presenting a method which uses a distance map from one or a set of reference objects to a number of objects of interest. We measure multiple interpretable geometric measures on these and present summary statistics, edge corrections terms and assess the equivalence class under the measures in a simplified example.

In cell migration studies, you want to measure the distribution of cells within some distance from a point of origin such as an injections site. Commonly, you only have point annotations limited to planar sections of your volume giving rise to a geometric problem since the measurement is represented as spherical shells but the annotation are planar. In Chapter 4, we present a theoretical foundation for estimating this spherical statistic from planar point annotations. We also present ideas on how to address the problem of estimating a full volume density by interpolating the density between sections using both a machine learning and an image registration approach.

Further, we present contributions to three advances across point statistics and pathology. Firstly, we present a K-function summary statistic on curves using a combination of a spatial distance and a directional distance measure which originates from mapping the curves into a Reproducing Kernel Hilbert Space in which a meaningful measure can be defined easily. We then present analysis and imaging pipeline descriptions of two contributions in pathology which have similar methodology and both provide use cases for the method presented in Chapter 3.

Resumé

Analysen af geometrien af objekter er en fundamental egenskab med vigtige fortolkninger i de fleste naturvidenskablige felter. Når felter som bioafbildning, biologi og patologi udvikles, ligeså må vi udvikle de beregningsmæssige og statistiske metode, som vi bruger i korrigering af billedartifakter samt til at måle den stokastiske variation i formen af geometriske objekter som har vores interesse.

Når man optager billeder af biologiske strukture på nano-skala, befinder vi os på den forreste grænse af hvad på nuværende tidspunkt er muligt. Dette kan tydeligt ses i de fysiske begrænsinger vi møder i forhold til skala, opløsning, synsfelt, samt signal/støj-forhold samt i sensitiviteten til at detektere specifikke objekter. Ligeledes stiger sandsynligheden for at der opstår afbildningsartifakter. På disse skalaer viser billederne sig ofte som støjfyldte og er sårbare overfor fejljusteringer og derformering hvilket kræver korrektion før man nøjagtigt kan karakterisere og måle geometries af de objekter som har vores interesse. I kapitel 2 præsenterer vi en justeringsmetode som bruger biologiske strukturer, som hvis middelform er stochastisk kendt, til at foretage god korrektion på specifikke typer data.

Når vi analyserer geometrien af objekter har vi ofte brug for at slutte om der er en geometrisk afhængighed imellem objekterne. Dette problem har brug for præcis definition fordi *form* ofte refererer til en selvstændig egenskab for hvert objekt. For at addressere dette spørgsmål, som opstår i praksis, må et relationelt form-mål nødvendigvis afhænge både af den rumlige afstand samt en relation imellem deres individuelle former. I kapitel 3 håndtere vi alle disse problemer ved at præsentere en metode som bruger en afstandfunktion fra et sæt af referenceojekter til et sæt af objekter vi er interesserede i at måle. Vi måler et udvalg af geometriske mål som direkte kan fortolkes. Vi præsentere opsummerende statistikker, kant-korrektion og kigger ligeledes på ækvivalensklassen givet under målet for et simpelt undersæt af former.

I et celle-migrationsstudie måle man fordelingen af celler som funktion af afstanden til et fælles oprindelsespunkt, f.eks. et injektionssted. I mange tilfælde har man kun punktmarkeringer i et begrænset sæt af planare vævssnit i det totale volume hvilket gør at der opstår nogle geometriske problemer fordi vi på den ene side måler i kugleskaller imens punktmarkeringerne er planare. I kapitel 4 præsenterer vi det teoretiske fundament for at estimere denne sfæriske statistik ud fra planare punktmarkeringer. We præsentere også idéer på hvordan man kan addressere problemet med at estimere punktdensiteten for det fulde volumen både ved at bruge machine learning samt billedregistrering.

Til slut præsenterer vi bidrag til tre fremskridt henover punktstatistik og patologi. Først præsenterer vi en K-funktions opsummerende statistik på kurver som bruger en kombination af den rumlige afstand samt et retningsafhængigt afstandsmål som opstår når vi afbilder kurverne ind i et Reproducing Kernel Hilbertrum hvorom et meningsfuldt mål let kan defineres. Derefter præsenterer vi analyse samt afbildnings-pipeline af to bidrag indenfor patologi hvorom metodikken overlap og begge drog brug af metoden

præsenteret i kapitel 3.

Dedication

To my mum and dad, whome always have been firm believers in the possibility of any human being, even in their sometimes troubled children. Who has listened with interest, wonder, and occasional confusion at my technical mutterings. To my supervisor whome I've had a countable infinite number of discussion in an uncountable infinite spectrum of topics, and who have been helpful to support a healthy work-life balance through trying times. To the handful of mentor figures in previous educations who fostered development by being open-minded role models and embodying the scientific method through their teachings. To my girlfriend and two children who has been my anchor to reality and given me a meaning to continue. An finally, to my brother-in-law, who sent me on this journey.

Acknowledgements

I want to thank my colleagues at the Computer Science Department at the University of Copenhagen (DIKU), who have provided an atmosphere of scientific growth, hospitable warmth as well as the structure for which to make the time spent there incredibly pleasant and prosperous. I want to thank Stine Hasselholt for the long time collaboration. A more thorough and heartful person will be hard to find. I also want to thank Carlos Benitez Villanueva and the team at the Center for Translational Neuromedicine for being welcoming and open during my stay at their lab.

Contents

Chapter 1

Introduction

Data science describes the shift to a more data-driven approach in industry and science alike. A change perpetuated by the availability of data, steady increases in computing power, advances in parallelism, and the advent of algorithms for large-scale processing and analysis. Biology, similarly to other fields, has also leveraged this approach with the advent of single sequence analysis, mass-scale chemical analysis and directly relevant to my work, in the acquisition, processing, and analysis of biological samples as three-dimensional images.

The advances in biology and data science are deeply coupled since on one hand, advances in biology often present us with new questions to be answered of which many require new image acquisition methods, larger amounts of data, new challenges to post-process the data, and require novel methods of analysis. Conversely, new methods in data science perpetuate advances in biology by allowing us to ask questions which were previously out of reach. For an example of this interdependence, we need to look no further back than to December 2020 where Google DeepMind's Machine Learning program AlphaFold [45] reached historic results in the CASP protein folding competition.

From the view of a Computer Scientist, answering biological questions requires that we formulate the question in a mathematical setting which parallels or *models* the real world. In the field of image analysis, we often construct an image processing pipeline that takes as input some unprocessed digital data and, through a series of steps, presents us with an output that can either be further interpreted or directly contains an answer to a hypothesis with some level of certainty.

1.1 Summary of the Founding Theory

Biological Imaging, or Bioimaging for short, covers the process of reproducing a biological sample visually as an image. Historically done by hand, drawing what was seen through a microscope, today large parts of the process are automated with the resulting image stored digitally. Image capturing methods include light microscopes detecting photons, electron microscopes detecting electrons, usually from within a vacuum chamber, and magnetic-resonance-imaging (MRI) where the magnetic response is used to reconstruct a volume image of the scanned object, to name a few.

Digital imaging presents an eternal struggle to find the best trade-off between field-of-view, resolution, and the resources to process and store the image. Ideally, you want to both see enough details of the biological objects of interest and have a sufficiently large field-of-view to observe the context of the objects for significant statistical analysis.

However, both requirements mean that the number of pixels in the volume will typically exceed the memory capacity of most current-day computers. Currently, the increase in the resolution of volume images is outpacing the increase in memory capacity by a significant margin. Additionally, for electron microscopes, a trade-off between noise level and sample integrity has to be found. Increasing the acquisition time increases the number of electrons fired at the sample, result often in sample degradation due to an increase in the sample's temperature.

Specifically, when producing volume images in electron microscopy, images are formed by combining a sequence of two-dimensional images. The process includes alternating between imaging one section of the material and then removing a layer, either by cutting as done in Serial Block-face Scanning Electron Microscopy (SBEM) or by milling using ions as done in Focused Ion Beam Scanning Electron Microscopy (FIB-SEM), or a similar alternative. Both processes result in a series of images that together form the volume. The proper recombination of an image series presents new challenges because the sections are often misaligned for a variety of reasons such as a buildup of charge in the sample which can deflect the electron beam, manual adjustment of the sample position during imaging, or changes to instrument parameters as well as physical deformations resulting from the cutting of the material.

The above circumstances present unique challenges and constraints when working with these types of images often handled in post-processing as a part of an imaging processing pipeline. An imaging processing pipeline refers to a series of actions performed on an image or set of images. A simple manual linear pipeline could be carried out in a standard image manipulation software, for example by cropping each image, then adjusting their brightness level, and finally threshold the result to yield a binary mask. In a biological context, we use image processing pipelines to achieve specific results, such as counting or measuring cells in an image. A pipeline need not be a linear sequence of steps but can be a sizeable graph of actions with multiple recombinations of temporary results and with multiple outputs such as both a filtered image and a mask of detected objects in the image.

Finally, when assessing objects in images, we are often interested in characterizing the shape of an object, either to get clues about the function of the object or to assess statistical differences between groups of objects. The shape of an object usually refers to the remaining characteristics of an object when you ignore the position and rotation of the object. How large is the object? Is the object bent, flat or rounded, elongated, jagged? We are as humans deceptively good at assessing such features at a glance, but we struggle to accurately assess the differences when the objects become complicated, when the variation is subtle, or when the number of objects is large. In science, we, therefore, lean towards turning the shape characteristics of the object into quantitative measurable parameters we can assess numerically and compare statistically. Examples of simple measures are the volume, the diameter of circumscribing sphere, and the surface area of the object. These fundamental measures, such as surface area and volume, can be combined to yield derived measures, such as the sphericity measure which describes how close the object is in shape to a perfect sphere. Fitting ellipsoids to the object yields three radii characterizing the object as a spheroid covering the shape space of spheres, oblate or prolate shapes. Another notable shape analysis methods are Principal Component Analysis (PCA) which is used to reduce the dimensionality of the variation of a group of objects into one where only the meaningful shape variation of the group is preserved. Similarly, spectral shape analysis represents the object by spectral components which are found by solving the Helmholtz Equations on a triangulated mesh. These are merely a

few popular examples of a plethora of methods developed in the last century.

In parallel to the above methods, spatial statistics, which is usually focused on analyzing the spatial characteristics of point-processes, has over a long period grown into a fully-fledged framework for analysis on families of objects under the term Stochastic Geometry. Here a family of objects is often viewed as the combination of a set of points and a corresponding set of marks, where each mark is a set detailing the interior of each object. The shape is here often measured in a summarized manner to assess features such as the fractional volume of objects in a volume and the spatial interdependence of the objects. A classic approach originated from spatial statistics in form of the K-function[11] which can effectively determine the interdependence of a realization of a point process, for instance comparing to the *standard candle*, the Poisson Point process, representing complete spatial randomness. The Poisson Process is therefore used to model many phenomena such as bus arrival times, uniform random placement of points which in turn is used to model natural phenomena such as raindrop locations on a piece of paper. If the K-function differs from a Poisson Process, it can determine if the point process exhibits attractive or repulsive behavior and show clearly if the points cannot be within a certain distance of each other, that the point process is *hard-core*.

1.2 Overview of Contributions

In this work, we present three main contributions which are based squarely on the experience and knowledge of both fields. In the first main contribution, I was inspired deeply by my good colleague Stine Hasselholt on how she manually aligned her biological volume images. We assessed the strict requirements to the accuracy of such realignment and found the classical approaches often gave unsatisfactory realignments in this specific task. From this, we present a new image correction method in Chapter 2 which uses biological shape characteristics to perform corrections at sub-pixel accuracy outperforming existing approaches. This paper is published in Communications Biology [50]. The second main contribution is a new measuring technique described in Chapter 3. The method is inspired by specific limitations of current approaches when wanting to examine variations in shape between groups of biological structures and builds upon the established theory in stochastic geometry. In Chapter 4, we present a method for assessing sparse point patterns radially from a source. We present a statistic on the points accounting for the missing data and whether the source point is known or inaccurate. Further, we present ideas on estimating a full volume density using a machine learning and registration approach.

Further, we will present contributions to several articles. In Chapter 5, we present a summary statistic for curves using a reproducing kernel Hilbert space. In Chapter 6, we present two analyses in pathology on ultrastructural changes in ALS model mice and Huntington's model mice respectively. In both, we use the methods previously described in Chapter 3.

The articles described in Chapters 2, 3, and 5 are presented here as published or submitted except for references and formatting.

Chapter 2

Restoring drifted electron microscope volumes using synaptic vesicles at sub-pixel accuracy

2.1 Abstract

Imaging ultrastructures in cells using Focused Ion Beam Scanning Electron Microscope (FIB-SEM) yields section-by-section images at nano-resolution. Unfortunately, we observe that FIB-SEM often introduces sub-pixel drifts between sections, in the order of 2.5 nm. The accumulation of these drifts significantly skews distance measures and geometric structures, which standard image registration techniques fail to correct. We demonstrate that registration techniques based on mutual information and sum-of-squared-distances significantly underestimate the drift since they are agnostic to image content. For neuronal data at nano-resolution, we discovered that vesicles serve as a statistically simple geometric structure, making them well-suited for estimating the drift with sub-pixel accuracy. Here, we develop a statistical model of vesicle shapes for drift correction, demonstrate its superiority, and provide a self-contained freely available application for estimating and correcting drifted datasets with vesicles.

2.2 Introduction

In three-dimensional (3D) scanning methods, such as focused ion beam scanning electron microscope (FIB-SEM), serial block-face imaging (SBF-SEM) and serial section transmission electron microscopy (SS-TEM), the scanning method alternates between forming a two-dimensional (2D) image and removing a layer of material. For 3D geometrical analysis, the sequence of 2D images must be recombined into a single, 3D image, which, unfortunately, is non-trivial. Sectioning of the tissue, as well as the subsequent imaging by electrons, often introduces a sideways sub-pixel translation between sections known as drift. In FIB-SEM, the drift may arise from a variety of practically uncontrollable factors, such as bending of the electron beam due to a charge gradient in the material and physical movement of the entire sample. Uncorrected drifts skew 3D distances, thus introducing errors in subsequent statistical and geometric 3D analyses, and in turn, on possible biological conclusions. Figure 2.1 shows an example of an ultrastructure brain region from a healthy adult rodent with easily noticeable drift when the dataset is viewed across multiple image planes. Datasets such as this have been and

Drift introduces angles to structure in the sectioning direction, z, and is invisible in a single section.

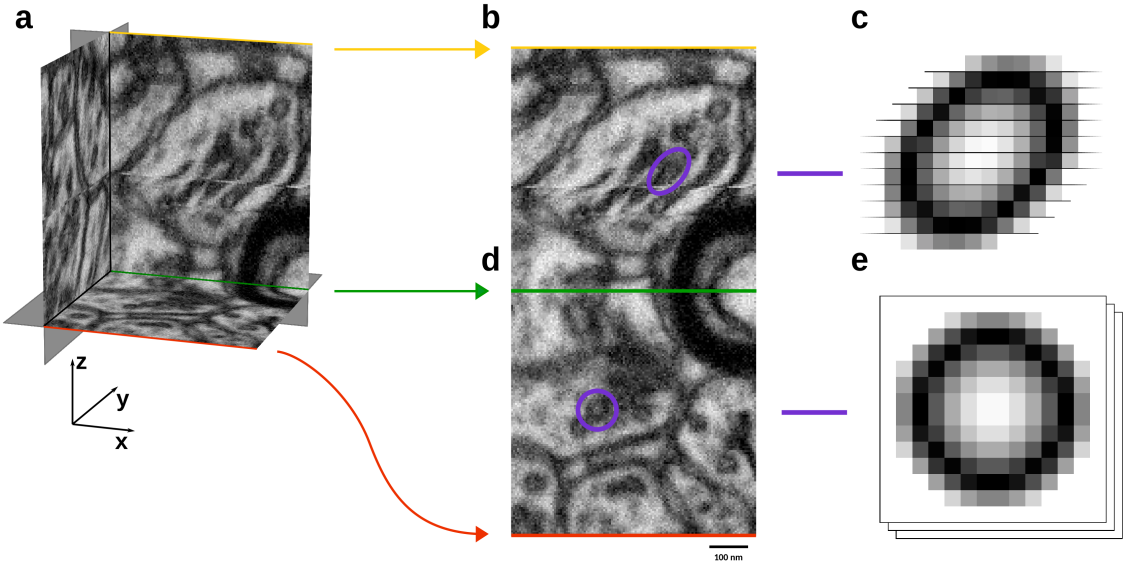


Figure 2.1: **a** Small block from the publicly available FIB-SEM dataset of the CA1 Hippocampus region of a healthy adult rodent (https://cvlab.epfl.ch/data/data-em/) showing a pre-synaptic region with vesicles displaying significant drift. **b** The xz-plane in a highlighting the apparent effect of drift. **c** Artists depiction of a drifted vesicle as they appear in **b**. **d** The x-y plane in a highlighting the lack of drift effects in this plane. **e** Artists depiction of a drifted vesicle as they appear in **d**.

are still used actively [61, 4, 5] with no apparent mention of the correction for potential drift. In these works, it is unclear what effect such misalignment may have had on the presented results. In other works [35, 32, 14, 28, 22], the correction has been performed using the ImageJ plug-in TurboReg (www.epf.ch/thevenaz/turboreg/), StackReg (http://bigwww.epfl.ch/thevenaz/stackreg/), Matlab, or a similar. These tools support both manual registration, where the user specifies corresponding landmark points, and automatic registration methods, e.g., pyramidal least-squares minimization of the image intensities [55], maximizing mutual information [9], and normalized mutual information [53, 10].

In FIB-SEM, large, pseudo-linear structures at an angle to the sectioning direction will appear to move spatially perpendicular to the sectioning direction when viewing the sections in sequence. This will misdirect typical automatic registration methods since they are unable to distinguish translations caused by drift and apparent translation caused by structures at an angle. Manual landmark annotations risk similar defects. An example of the problem can be seen in Figure. 2.2**a**, **b**, where a simple synthetic 3D FIB-SEM image has been generated with two spherical vesicles and a single membrane-like structure at an angle to the sectioning direction. Even though no drift is imposed here, a standard registration approach translates each image to force the membrane to be perpendicular to the image section direction, stretching the vesicles in the process. (Figure. 2.2**c**, **d**), shows a similar effect on real data. The reason is that the membrane-like structure dominates the dissimilarity measure since its volume is much larger than that of the vesicles.

A problem with many registration methods is that they are agnostic to the image content. For example, real neuronal tissue contains small and large structures, and stan-

Standard registration methods confuse drift and naturally occurring trends in images.

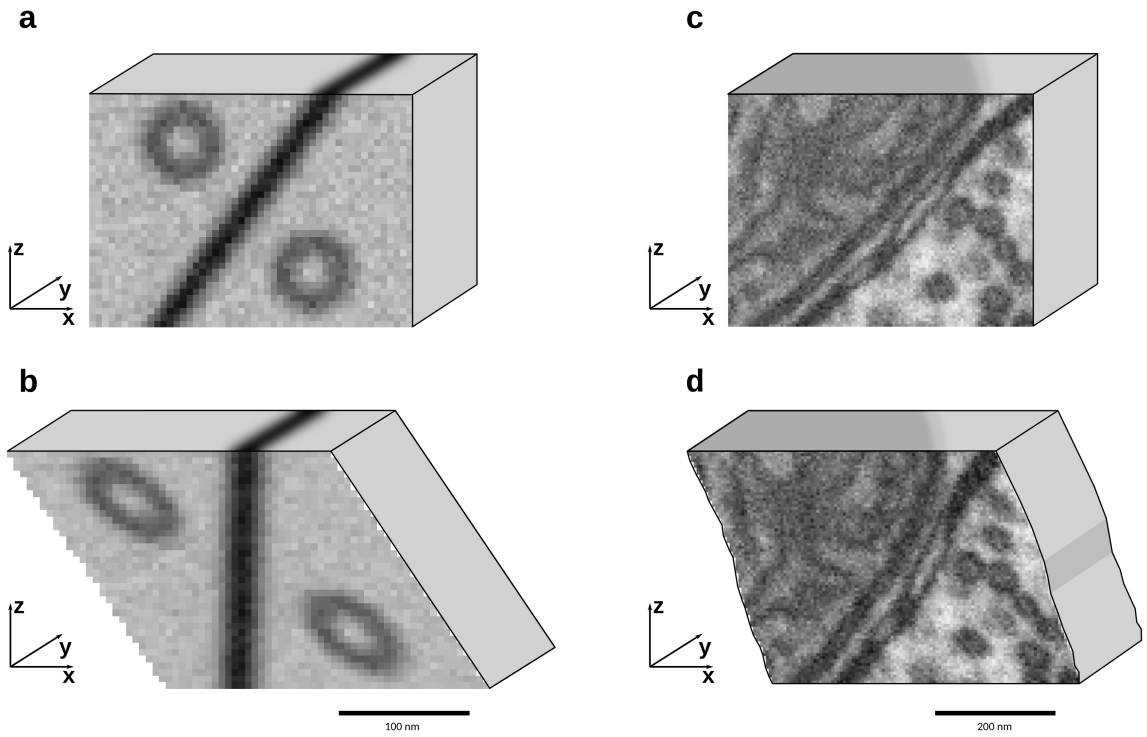


Figure 2.2: **a** A synthetic image with two model perfectly spherical vesicles and a model membrane at an angle to the sectioning direction. **b** The result of using standard section-by-section registration with Mutual Information for dissimilarity measure and the implementation found in Matlab. **c** A FIB-SEM sub-volume with features which influence standard image measures adversely. **d** The result of using standard section-by-section registration displays signs of deformation by the vesicles being less spherical. This sub-volume was registered using ImageJ with StackReg and TurboReg plugins.

dard registration methods perform well when the angles of these structures are evenly distributed with respect to the sectioning direction. However, we have observed that this is not the case for the FIB-SEM images, we have analyzed. Firstly, for small regions of interests dominating pseudo-linear structures often appear, and at larger regions of interest, neuron processes have a tendency to be similarly oriented, resulting in inaccurate drift estimates. Thus, registration methods relying on global measures or landmark points will be less than optimal for such sections.

For improved registration, we must include models of the imaged data, such that the registration method can distinguish between drift and apparent drift caused by dominating structures at an angle to the sectioning direction. In images of neuronal tissue at nano-resolution as, e.g., the publicly available FIB-SEM dataset of the CA1 Hippocampus region of a healthy adult rodent (https://cvlab.epfl.ch/data/data-em/, $5\,\mathrm{nm}^3$ voxel size, $1065 \times 1536 \times 2048$ voxels) we observe that vesicles are abundant, small, and on average spherical. A vesicle has a lipid bilayer shell, which physically can be modeled as an elastic material with a bending energy density functional [6, 20] minimizing the curvature of the vesicle's surface. Hence, in equilibrium and without external forces, a vesicle's shape is spherical. Vesicles can take a variety of exotic shapes under special conditions [44, 33],

Our drift estimation uses the following four steps.

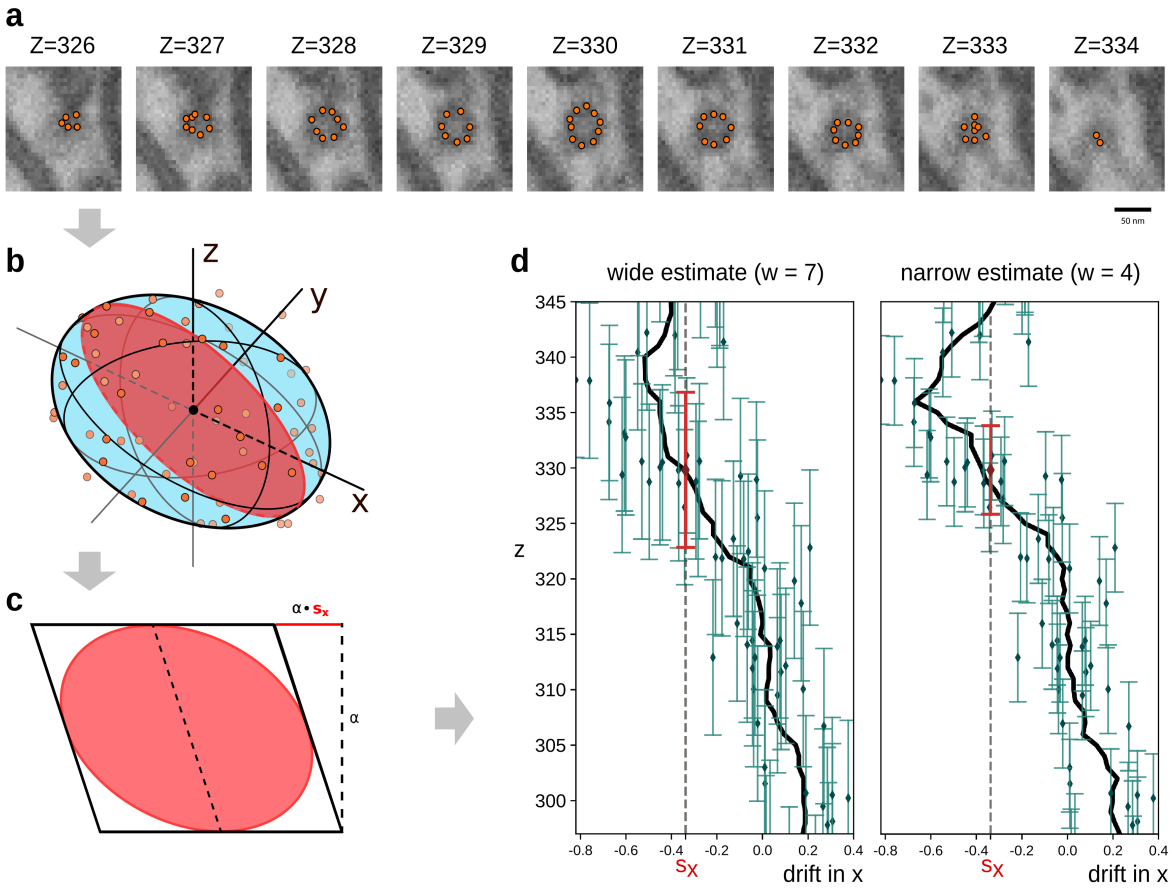


Figure 2.3: **a** The boundary of vesicles is manually annotated. **b** An ellipsoid (blue) is fitted to the annotated points (orange). The intersection of the fitted ellipsoid with the z-x plane is shown in **c**. **c** The skews in the zx- and yz-planes is calculated, here illustrated in the z-x plane only. d For each image section, the drift is estimated by the average of the drift components in x, y, and z. Diamond marker denotes a z-location of a vesicle and the bars illustrates their influence in the local average. **d** left/right shows the effect of varying the user-specified length w on the drift estimate.

though their most probable and most common shapes are spheroids, prolates, and oblates. In this work, we propose to model the variability of the vesicle shape as ellipsoids. Since vesicles are numerous near synapses, since synapses are numerous in our images, and since vesicles on average are expected to be spherical, we can estimate the drift as the average, per section deviation from the spherical shape. We claim that this method is independent of large-scale structures such as the orientation of neuronal processes.

Our method is summarized as follows: firstly, we annotate the boundary of the vesicles by manually placing points in the images (Figure. 2.3a). Secondly, we fit an ellipsoid to the annotated points of each vesicle using a least-squares approach (Figure. 2.3b). Thirdly, we assume first that the skew of each estimated ellipsoid alone is due to drift and calculate a drift component for each of the ellipsoids (Figure. 2.3c). Fourthly, we computing the average of the drift-components by regarding each drift-component estimate as a point observation at the center of the ellipsoid, and then for each section i, average all point observations within sections $i \pm w$ where w defines a width of the estimate (Figure. 2.3d).

2.3 Results

To assess the quality of drift estimates, we generated three synthetic datasets. The first two have constant drift across all sections: $(\delta x, \delta y) = (0.3\,\text{pixels}, 0.0\,\text{pixels})$ and $(0.1\,\text{pixels}, 1.0\,\text{pixels})$, and the third dataset has a drift that varies across sections (Supplementary Information and Supplementary Fig. 1). Our synthetic datasets were generated by placing random ellipsoids with uniformly random axis lengths between 3 and 6 voxels, uniformly random orientations, and uniformly random positions in a non-overlapping manner, after which the drift was added. We annotated a variety of different numbers of vesicles across image sections to assess the influence on the drift estimate. A total of 71, 97, and 283 vesicles were manually annotated in the synthetic datasets described above. Annotating the synthetic datasets took on the order of 7 h in total, which is approximately 1 vesicle per minute.

Our experiments on the synthetic datasets demonstrate that using ellipsoids to estimate the drift is significantly more accurate than standard approaches on the synthetic datasets with constant drift. Our method estimates the drift with an absolute average error of $0.22 \cdot 10^{-1} \pm 0.08 \cdot 10^{-1}$ apparently independent on the drift magnitude. In contrast, the standard registration methods estimate the drift with the error $1.12 \cdot 10^{-1} \pm 1.48 \cdot 10^{-1}$ apparently proportional to the drift magnitude. (Histograms of drift estimates are shown in Supplementary Figs. 2 and 3). The standard approaches are biased even though no larger structures like cell membrane or mitochondria are present, and we suspect that the bias is caused by subtle imbalances in the statistical distribution of image content angles in the sectioning direction.

Our synthetic image with varying drift is non-smooth. In the presence of large drift changes in the sectioning direction, the estimated drift from ellipsoids estimates are smoothed out across the change (Figure. 2.4). This smoothing is due to two factors: firstly, the vesicles are fitted across multiple sections, which adds some error. Secondly, the estimate is based on vesicles with the set distance, w from the section. Reducing w will reduce the smoothing effect but also increase the effects of noise. Aside from the regions with large drift changes, we see a significant improvement over standard approaches (Supplementary Figure. 4). We note that the standard approaches have some variation from section to section, whereas our estimates based on ellipsoids are very stable (Supplementary Figure. 5).

Our drift estimate depends on the number of vesicles annotated, and to assess this dependence, we employ a bootstrapping approach: We use the total set of fitted ellipsoid drifts and 50000 times sample a subset of 1-200 drift-point estimates. The resulting average absolute error shows a perfect reciprocal dependence (Supplementary Fig. 6). Fitting to this bootstrapped data gives a predicted error function of $y = 0.1375x^{-0.4915}$, thus for one vesicle, an estimated error of 0.1375 is obtained, and to halve the error, your roughly need to annotate four times the vesicles. For the real FIB-SEM dataset, we manually annotated 961 vesicles. This dataset has 1065 pixels in the sectioning direction distanced 5 nm apart. Assuming that the average height of a vesicle is 45 nm this implies that there a vesicle on average is seen in nine sections. Thus, on average we have annotated $961 \cdot 9/1065 \cong 8.12$ vesicles per section, and the estimated error is $0.1375 \cdot 8.12^{-0.4915} \cong 0.049\,\text{pixels} \cong 0.25\,\text{nm}$ for each section.

The drift estimation accuracy also depends on the variation in radii of the vesicles. Specifically, in the extreme case that the vesicles are exactly spherical in the without drift, we only need a single vesicle to estimate a constant drift exactly since the drift-component will be equal to the drift. We therefore also assessed the correspondence between both

On synthetic images, our local drift estimates are precise for when the drift is constant and show smooth transition when drift varies.

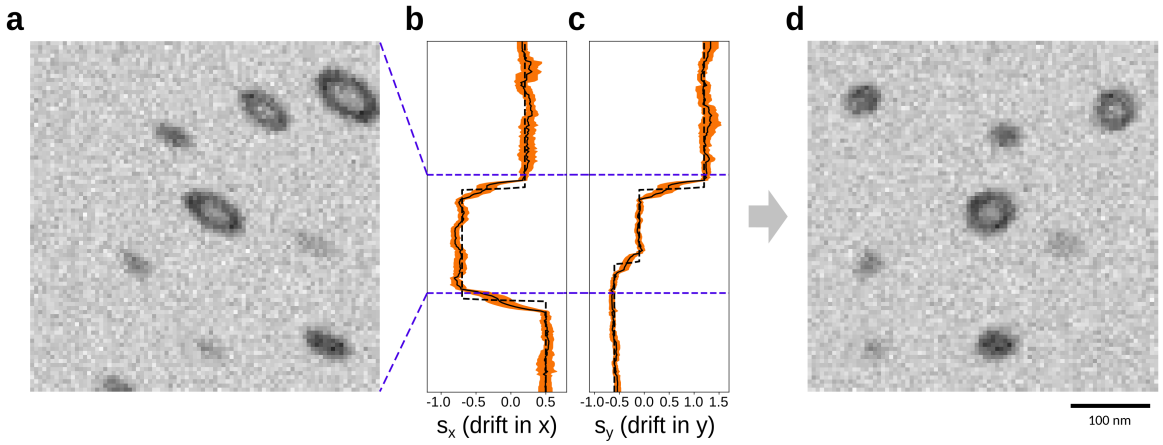


Figure 2.4: **a** A small subsection of a drifted synthetic image shown in a side view (zx-plane). **b**, **c** The estimated ellipsoid drift in x and y, respectively, as a function of z. Stippled curve is the ground truth, black curve is the estimate, and orange shows the variation in the per-vesicle estimates. **d** The same synthetic image corrected in z-x plane.

radii variation and drift magnitude on the estimation error. We generate a fourth synthetic image with 50000 vesicles for a variety of drift magnitudes and vesicle radii and measure the mean absolute error using ten vesicles for each estimate (Supplementary Fig. 7). Firstly, the figure shows a clear dependence on the variation of the radius. If radius does not vary, and if each vesicle is a sphere, then there is no error, but the error increases as the variation in radius increases. Secondly, the figure shows that the estimate is unaffected by the drift magnitude. Thus, estimating a sub-pixel drift and large drift will give an error of equal size.

For this work, we compare our method with the registration methods using the sum of squared difference (SSD) and mutual information (MI), normalized mutual information (NMI), and normalized cross-correlation (NCC) as dissimilarity measures. For completeness, we also experimented with correction using phase correlation17 and compared with built-in registration implementations in Matlab. Not reported here, we also experimented with optical flow estimation as implemented in OpenCV across the pairwise image section using only subregions of the images with vesicles. Our method is clearly superior to the state-of-the-art global registration methods on the synthetic images when compared to the ground-truth drift.

For real FIB-SEM images, the ground-truth drift values usually do not exist. Hence, the following conclusions are based on our experience with synthetic data described above. In the publicly available FIB-SEM dataset of the CA1 Hippocampus region of a healthy adult rodent (https://cvlab.epfl.ch/data/data-em/), we observe a significant non-zero drift signal (Fig. 5). After drift-correction using our drift-estimates, the vesicles look visually less ellipsoidal. Qualitatively comparing the drift estimate on the real with the synthetic images, we find that the estimate on the real dataset is similarly distinctly different from standard approaches (Fig. 6 and Supplementary Fig. 8). The estimates by the individual standard approaches can be seen to be in close agreement with each other with regards to both the magnitude and direction of the translation. However, for subregion in the real dataset, we also see that standard methods are biased with respect to image content (Fig. 6). Asserting the section-wise drift, we observe rapid changes in

Our drift estimates show similar behavior on real data as on the synthetic data.

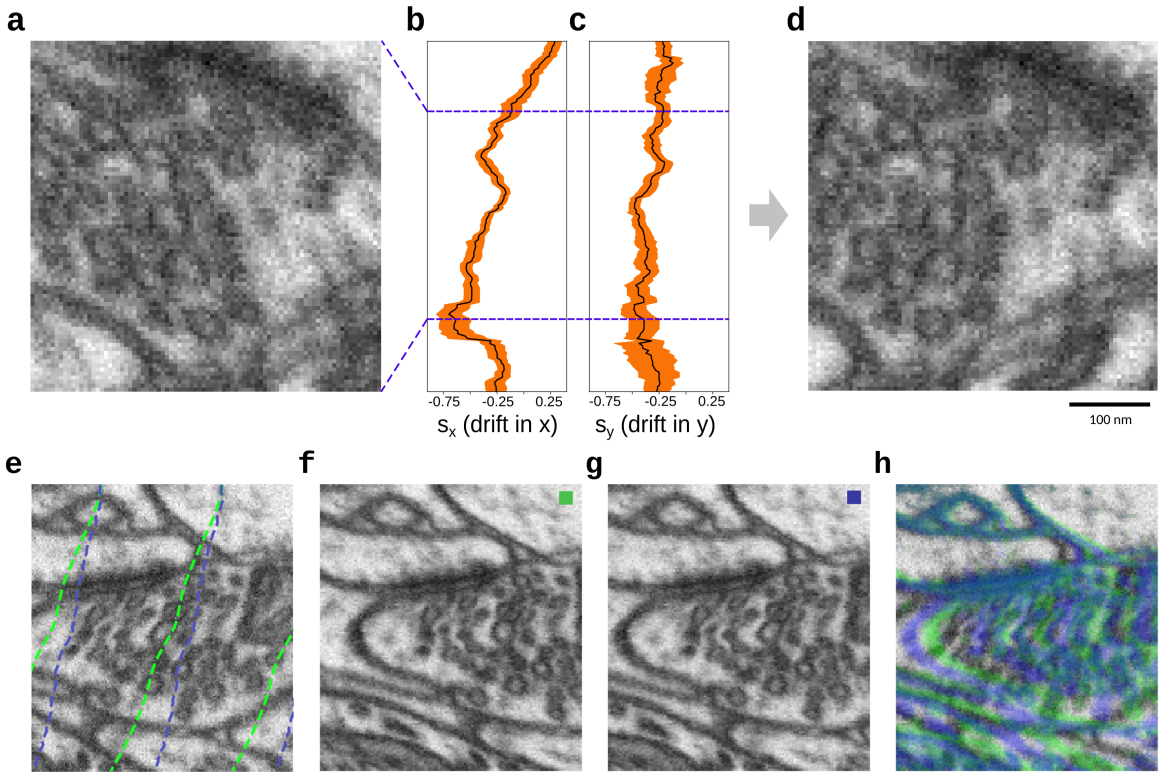


Figure 2.5: **a** A small subsection of a drifted FIB-SEM image shown in a side view (z-x plane). **b** The estimated ellipsoid drift in x as a function of z. **b**, **c** The estimated ellipsoid drift in x and y, respectively, as a function of z. Black curve is the estimate, and orange shows the variation in the per-vesicle estimates. **d** The same FIB-SEM image corrected in z-x plane. **e** Comparison of drift estimated by our proposed method (green) and using normalized mutual information image measure (blue). **f** The resulting registration using our proposed method. **g** The resulting registration using normalized mutual information. **h** Pseudo-color overlay of registration results obtained by our proposed method compared to registration using normalized mutual information.

drift estimates similarly to the synthetic image with varying drift (Supplementary Fig. 9). Hence, we expect some estimation error and smoothing effects to be present for the real data as well.

2.4 Discussion

To conclude, FIB-SEM images often suffer from sub-pixel drift often in the order of 0.5 pixels, and this drift accumulates across several slices resulting in distortion of distance and shape measures in 3D. Standard registration methods fail to correct this drift, as these methods cannot distinguish between drift and naturally occurring slopes in the sectioning direction. We have discovered that due to the abundance of vesicles in neuronal tissue and their biomechanical properties, they function well as statistical markers for drift, and we have presented a simple method for identifying and correcting the drift. We have compared our method with state-of-the-art registration methods based on global measures, and our

Drift estimates for standard registration method are consistent and most often outside the 95% confidence interval of our method.

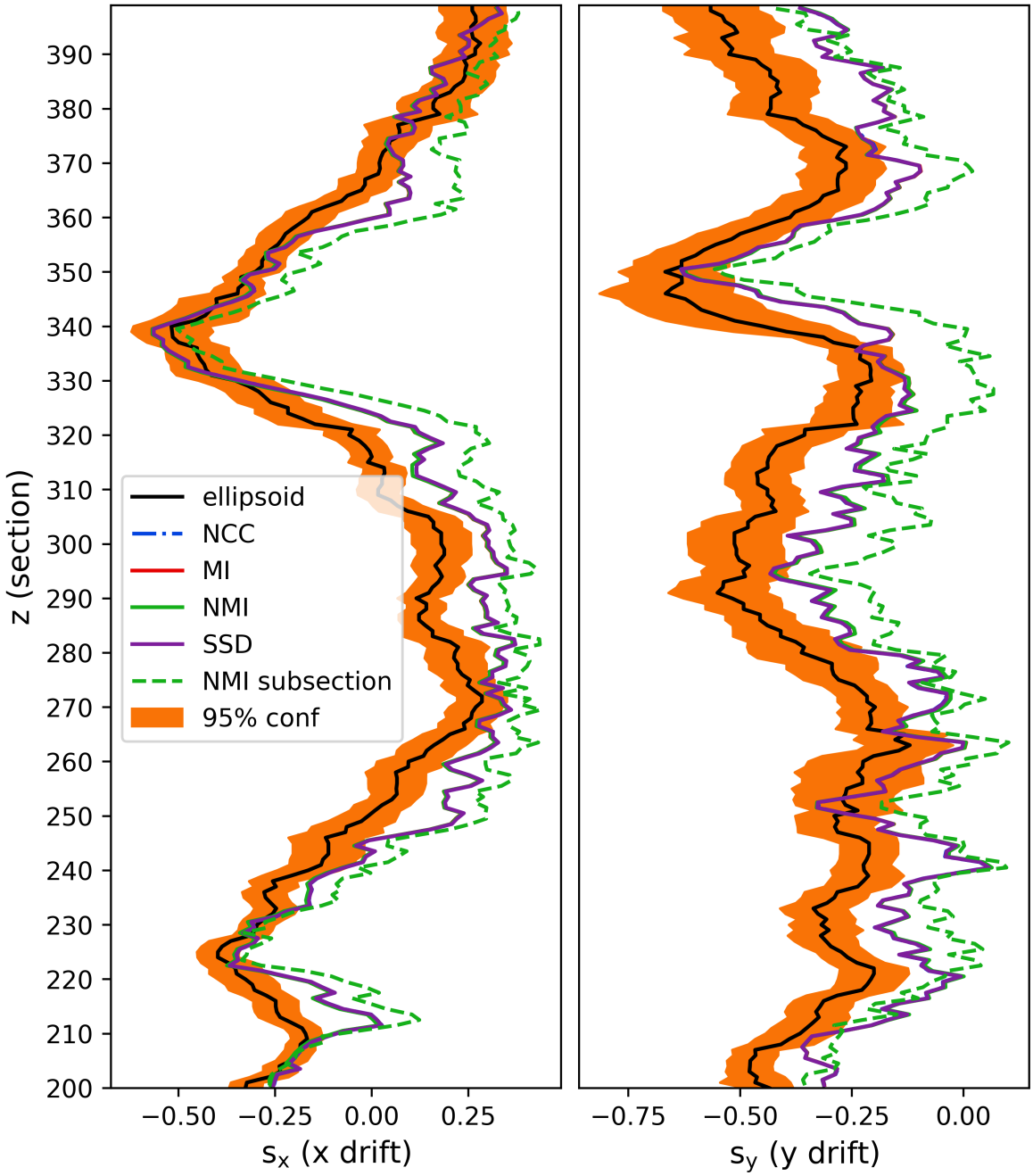


Figure 2.6: (left) and (right) show the estimates in the x- and y-direction. The stippled curve shows the result of applying the standard registration method on a subsection of the image, which is dominated by a large process.

method has proven to be more accurate. We encourage correction of drift in neuronal tissue whenever the analysis is of or relies on the geometry of the structures. Further work should be carried out to assess the effect of the drift on biological images and to develop less labor-intensive methods for estimating and correcting this drift.

2.5 Methods

Our method consists of the following sequence of steps: (1) Annotating vesicle boundary using points. (2) obtaining ellipsoids by least squares estimation on the boundary points. (3) calculate drift parameters for each ellipsoid. (4) Estimate the average drift locally to each section using the ellipsoids drift parameters in conjunction with their position. Our method is implemented and available online[49].

Choice of ellipsoid representation

Let x, y, z be the axes of an image with x, y the plane of each image section and z the image sectioning axis. An ellipsoid centered at the origin can be described implicitly by the quadratic surface equation $\mathbf{u}^T H \mathbf{u} = 1$, where $\mathbf{u} = [x, y, z]^T$, and H is a 3×3 symmetric positive definite matrix,

$$H = \begin{bmatrix} A & D & E \\ D & B & F \\ E & F & C \end{bmatrix} \quad . \tag{2.1}$$

We will refer to the elements of H as the parameters of the ellipsoid.

Obtaining ellipsoids from boundary points

We fit an ellipsoid to the vesicle membrane points of each vesicle in 3D. At least 9 non-degenerate points are required for a unique fit of an ellipsoid with an arbitrary center, radii, and rotation. We choose to fit the ellipsoids given as $\mathbf{u}^T H \mathbf{u} = 1$ using a least-squares approach [57] which we implemented in Python. We briefly compared this method to a numeric gradient decent approach minimizing the squared distance in the Euclidean norm as it has a slight difference in result compared to the algebraic norm defined by the ellipsoid equation. Our results showed that the algebraic norm minimization method produced slightly more accurate drift estimates while being many orders of magnitude faster than the numeric decent method.

Estimating drift from ellipsoid parameters

We define the drift in the image as a sideways translation of each image section with respect to the previous section. Let $\delta x, \delta y$ be the amount of translation by which some section is translated with respect to the previous section and let Δz denote the distance between subsequent sections. We represent the translation as a shear map with shear coefficients $s_x = \delta x / \Delta z, s_y = \delta y / \Delta z$. If we assume the drift is constant as a function of z, we can represent the drift as one single mapping S given by

$$S\mathbf{x} = \begin{bmatrix} 1 & 0 & s_x \\ 0 & 1 & s_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x + s_x z \\ y + s_y z \\ z \end{bmatrix} = \mathbf{u} \quad . \tag{2.2}$$

The shear mapping is a non-singular linear transformation, and since $S^{-1}S$ is the identity transformation, the quadratic equation is still solved when

$$1 = \mathbf{x}^T H \mathbf{x} = \mathbf{x}^T (S^{-1}S)^T H S^{-1} S \mathbf{x} = \mathbf{u}^T S^{-T} H S^{-1} \mathbf{u} \ . \tag{2.3}$$

Thus, transforming each point on the ellipsoid by S corresponds to a new quadratic surface defined by the matrix representation $\hat{H} = S^{-T} H S^{-1}$, or equivalently $H = S^T \hat{H} S$. Since an ellipsoid is a quadratic surface with a closed surface, and since non-singular linear transformations on closed surfaces cannot produce open surfaces, we conclude that the result is still a closed surface defined by a quadratic surface, i.e., an ellipsoid, spheroid or sphere.

Let $\hat{H}$ be the shear-transformed ellipsoid estimated from data with elements $\hat{A}$, $\hat{B}$, $\hat{C}$, $\hat{D}$, $\hat{E}$, and $\hat{F}$. The values of $\hat{E}$ and $\hat{F}$ gives the shape of the ellipsoid in the z-x and the y-z plane, i.e., the tilt of the ellipsoid as a function of z. An untilted ellipsoid is symmetric across the plane z and has $E = F = 0$. Defining the untilted ellipsoid in terms of a shear-transformed tilted ellipsoid $H = S^T \hat{H} S$, we set $E = F = 0$ and solve for s_x and s_y. We get

$$s_x = \frac{\hat{D}\hat{F} - \hat{B}\hat{E}}{\hat{A}\hat{B} - \hat{D}^2} \quad , \quad s_y = \frac{\hat{D}\hat{E} - \hat{A}\hat{F}}{\hat{A}\hat{B} - \hat{D}^2} \tag{2.4}$$

Let $\mathbf{s} = [s_x, s_y]^T$ represent the shear of some ellipsoid. By assumption, each ellipsoid is rotated uniformly at random. Thus, it follows that given no drift in the data, we should have $\mathbb{E}[\mathbf{s}] = 0$, since by an argument of symmetry, a tilt in any direction should be equally likely. Assume now we add some drift $\mathbf{k}$, giving rise to new shear parameters $\mathbf{s}'$. Since the composition of shear transformations simply amounts to adding the shear parameters, and by the linearity of expectation, we get

$$\mathbb{E}[\mathbf{s}'] = \mathbb{E}[\mathbf{s} + \mathbf{k}] = \mathbb{E}[\mathbf{s}] + \mathbb{E}[\mathbf{k}] = \mathbf{k} \ . \tag{2.5}$$

Thus, given N fitted ellipsoids with s_i the vector of shear constants for ellipsoid E_i, $1 \leq i \leq N$, we estimate the drift in the images $\mathbf{k}$ by the average drift,

$$\mathbf{k} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{s}'^{(i)} \ . \tag{2.6}$$

Enumerating the image sections by I_j, $1 \leq j \leq M$ such that $I_1, \ldots, I_N$ are ordered with increasing z choosing I_1 as the reference image, drift correction can be obtained by transforming I_j by $S^{-(j-1)}$. Drift correction assuming varying drift

Since drift in images may vary, e.g., due to manual correction during the scanning operation, movement of the sample, or charge equalization, it is likely that the amount of drift varies in the sectioning direction. Given a large enough population of ellipsoids, it is possible to estimate the drift per image section. To accomplish this, we view the individual estimated ellipsoid-drifts as a point-estimate at the center of the ellipsoid. Specifying a width w of the point estimate, we compose a drift estimate for each section by the average of the ellipsoids with center closer than w to the section. Denoting $d(E_i, j)$ the perpendicular distance in the section direction from ellipsoid E_i to section j, we can write the drift estimate $\mathbf{k}_j$ of section j as

$$\mathbf{k}_j = \sum_{i=1}^{N} \frac{1_{d(E_i,j)<w}}{\sum_{n=1}^{N} 1_{d(E_n,j)<w}} s'^{(i)} \tag{2.7}$$

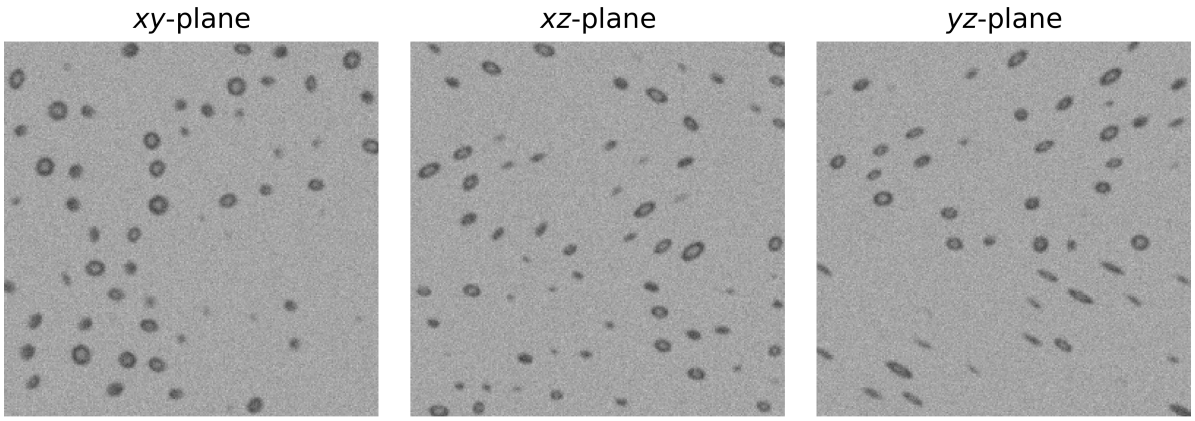


Figure 2.7: Example views of the synthetic dataset with varying drift. Synthetic vesicles can be seen drifted in different directions in the xz and yz-planes, an effect not noticeable in the classic xy-plane.

where 1_P takes the value 1 when P is true and 0 otherwise. In the absence of visible vesicles in one or multiple sections, we suggest either interpolating the drift parameters from nearby known values or assume the drift is zero, depending on the dataset. Reporting summary

Further information on research design is available in the Nature Research Reporting Summary linked to this article.

2.6 Data availability

Our synthetic, derived data, and data for producing our figures can be provided upon request. In this paper, we have discussed the publicly available fib-sem data from https://cvlab.epfl.ch/dat em/.

2.7 Code availability

The code for producing our figures can be provided upon request. The drift correction application is available at (https://doi.org/10.17894/ucph.b61d5ca9-53df-4909-92ee-f8ee026e39bb). The accompanying source code is available upon request.

2.8 Supplementary Figures

2.9 Supplementary Methods

To validate our method, we generated multiple synthetic FIB-SEM datasets where the added drift could be systematically controlled. While imitating all aspects of FIB-SEM images is vastly outside the scope of this work, we decided on a subset of requirements the images would have to fulfill. Thus, the dataset was designed with a real FIB-SEM dataset as reference focusing primarily on generating vesicles as they appear there (https://cvlab.epfl.ch/data/data-em/). We modeled vesicles as ellipsoids with varying radii and rotated uniformly at random. Vesicle membrane thickness was chosen to

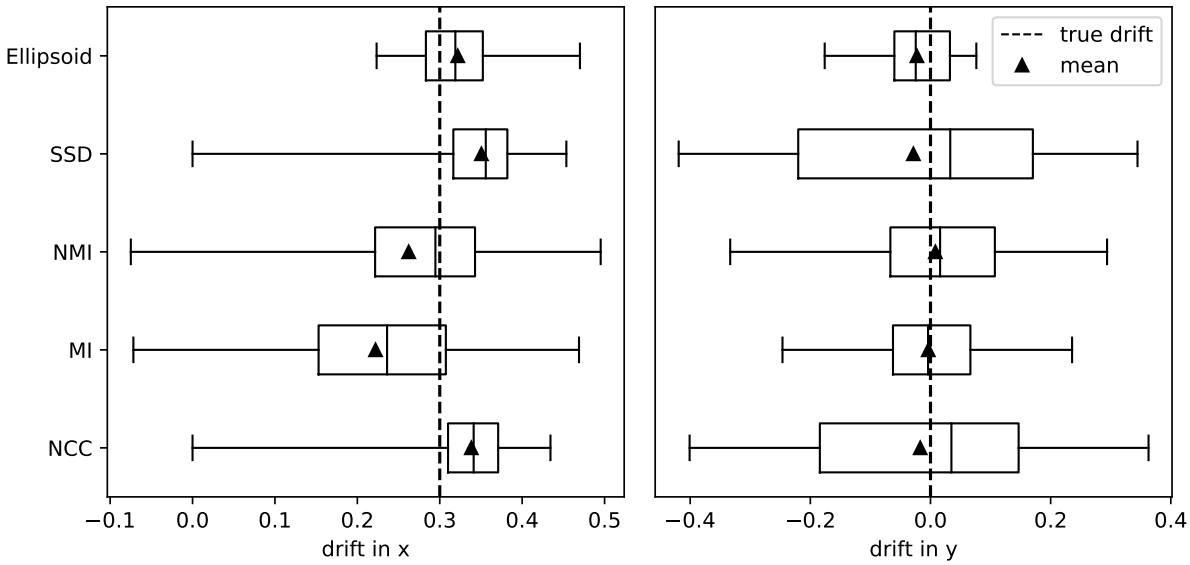


Figure 2.8: Comparison of the distribution of section drift estimates by the ellipsoid method and standard approaches on the synthetic dataset with a constant drift of $(x, y) = (0.3, 0.0)$. Left: drift-estimate in the x-direction. The mean estimated drift for our ellipsoid method has both high accuracy and better precision than the comparison methods.

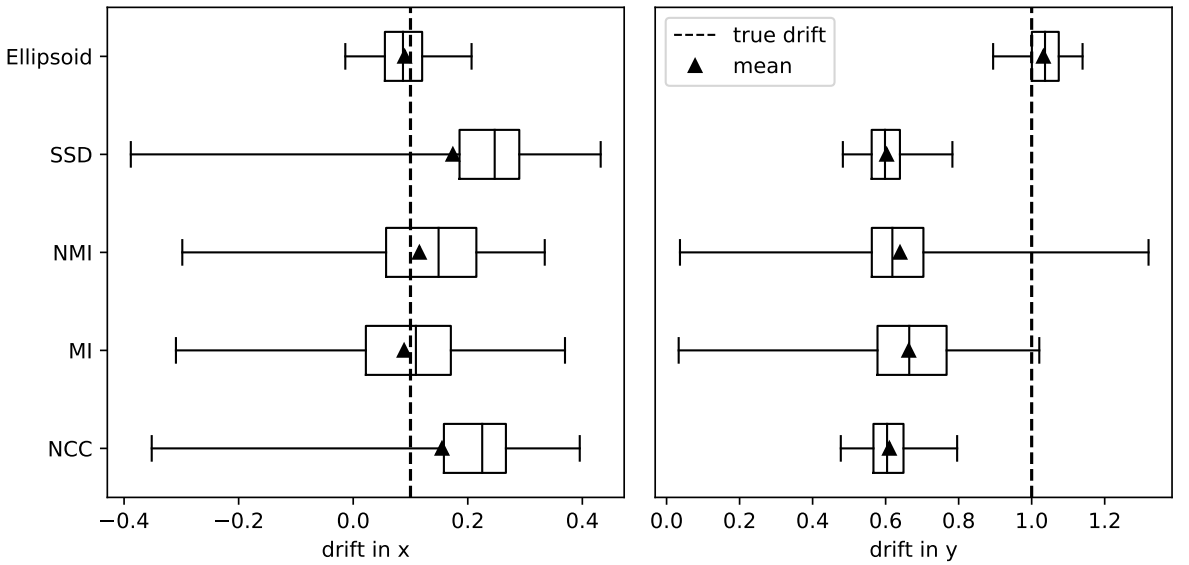


Figure 2.9: Comparison of the distribution of section drift estimates by the ellipsoid method and standard approaches on the synthetic dataset with a constant drift of $(x, y) = (0.1, 1.0)$. Left: drift-estimate in the x-direction. For our method, the mean estimated drift is in close agreement with the true drift and display better precision than the comparison methods. The error of the standard, global methods appear to increase with increasing drift.

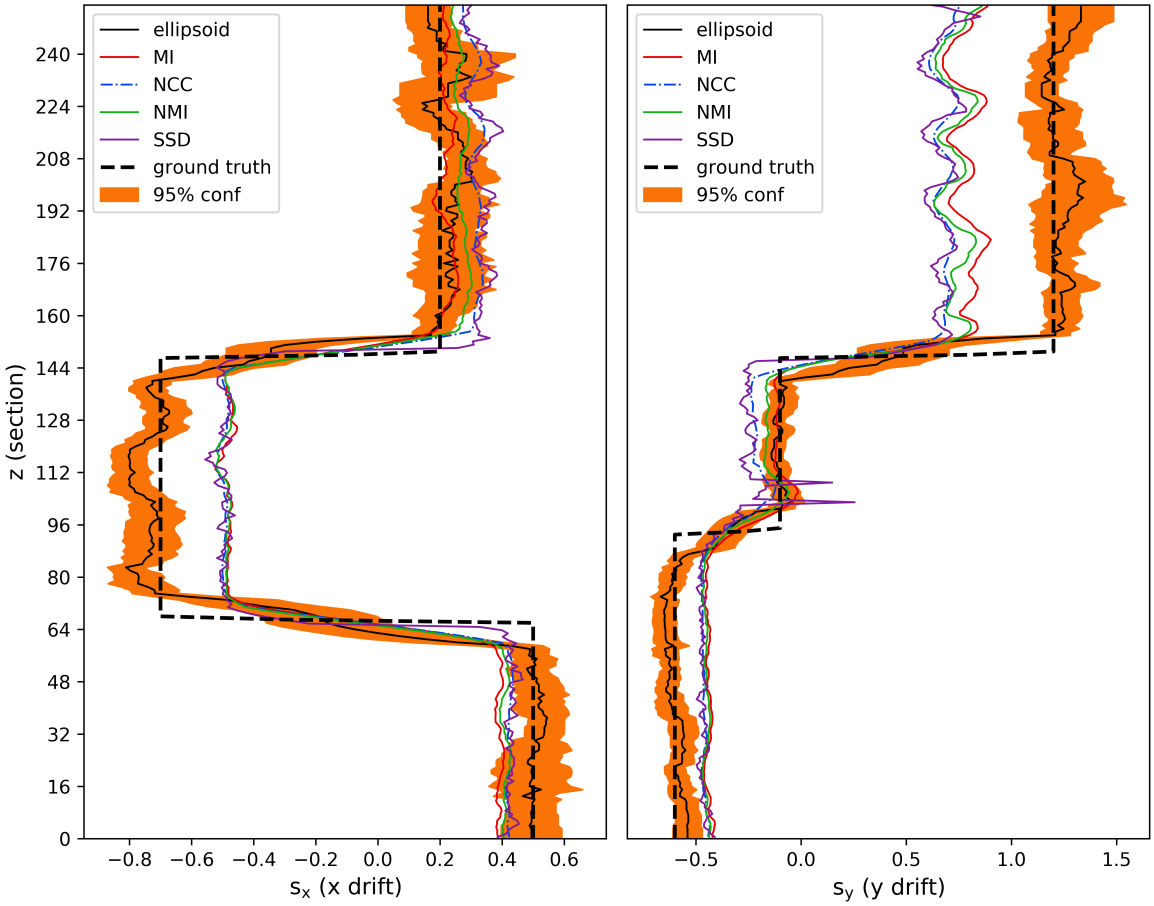


Figure 2.10: The complete drift estimate comparison in all 256 sections of the synthetic dataset with varying drift. All sequences except the one denoted ellipsoid were convolved with a box kernel of size 11 to counter the noise in those approaches to make visual inspection possible. The standard registration methods produce biased drift along both the x and y-axis.

visually match the thickness of the membranes as they appear in the reference dataset. The grayscale values and the contrast were estimated by measuring mean grayscale values in regions in our comparison FIB-SEM image with visually near uniformity for both cytosol and membrane. The radii of the vesicles were generated uniformly at random in the interval from 3 to 6 voxels. Following experiments shows the distribution in real FIB-SEM images are most likely log-normally distributed (Supplementary Figure 2.16). Experiments showed this difference did not have a significant effect on the estimation error.

FIB-SEM images appear blurry because of limitations when imaging at the nano-scale. Modeling this as Gaussian smoothing, we fitted a Gaussian smoothed, box function to membranes of the image. We resampled the image linearly along a line crossing the membrane approximately perpendicular to the membrane. Since the membrane is a 3D structure while we only resampled along a 2D plane, the standard deviation of the Gaussian will be larger when the membrane is not perpendicular to the section direction. Thus, given the resulting distribution (Supplementary Figure 2.17), we conservatively chose to set $\sigma = 1$. FIB-SEM images are also inherently noisy and by our experience appears to be sufficiently well modeled as Gaussian noise. The standard deviation of the noise was

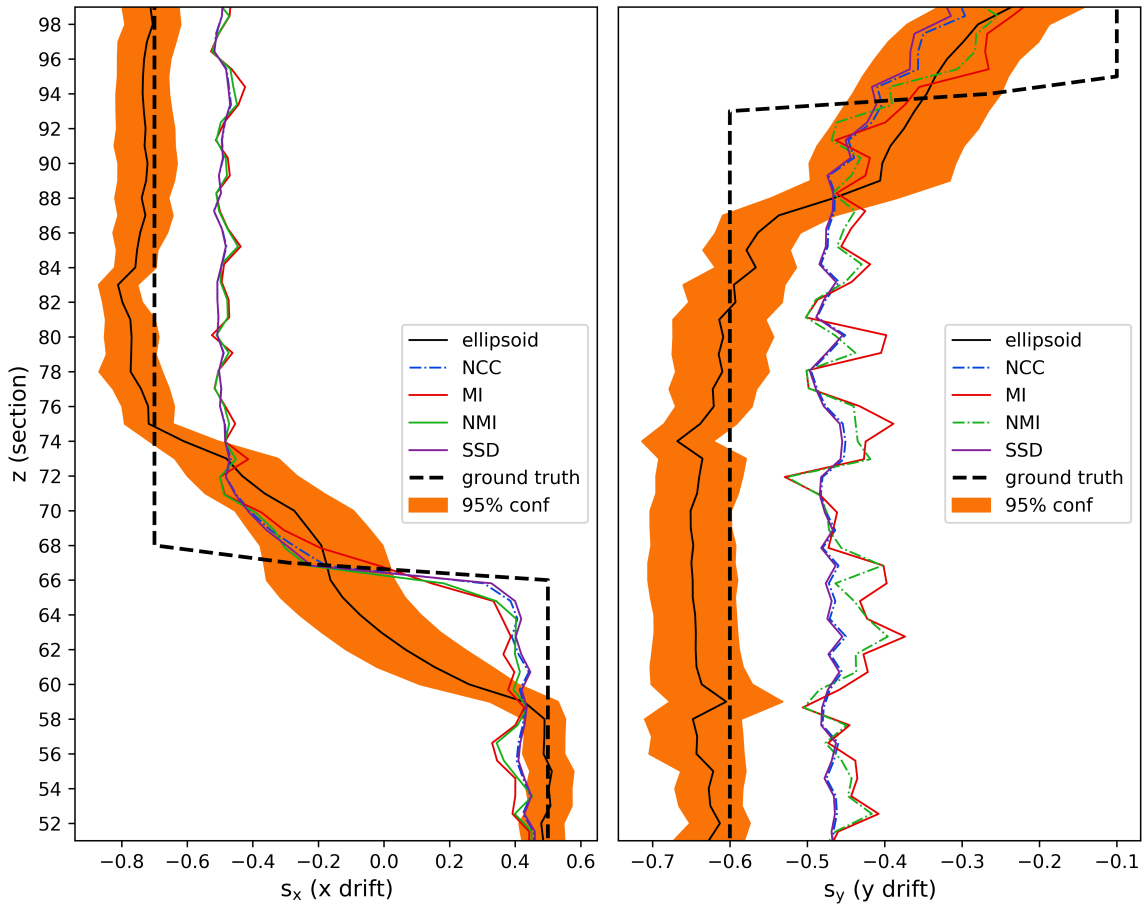


Figure 2.11: A small subsection of the drift estimate in the synthetic dataset with varying drift. Here the results of the standard approaches are presented without any present smoothing meaning the rapidly varying estimate of the standard registration methods is visible.

estimated by measuring the standard deviation of grayscale values in selected regions in our comparison FIB-SEM image with a near uniform appearance, usually cytosol regions without organelles or visible cytoskeleton.

Synthetic datasets were generated by initializing a volume array of a desired size with the estimated cytosol grayscale value. Vesicle centers were then chosen randomly in the volume. Placement of the vesicle was then attempted several times disregarding those which did overlap with previously placed vesicles. For this purpose, we represent vesicles by the ellipsoid matrix H (2.1). Drift was added by multiplying the drift matrix S (2.2) to each point $\mathbf{x}$. For every point x in the volume, vesicle membrane grayscale value was set in the volume if $|\mathbf{x}TSTHS\mathbf{x} - 1| < 0.25$. After placing a desired number of vesicles, the volume was convolved with a Gaussian kernel to apply the smoothing, whereafter Gaussian noise was added to the images. Values outside the intensity range were clipped.

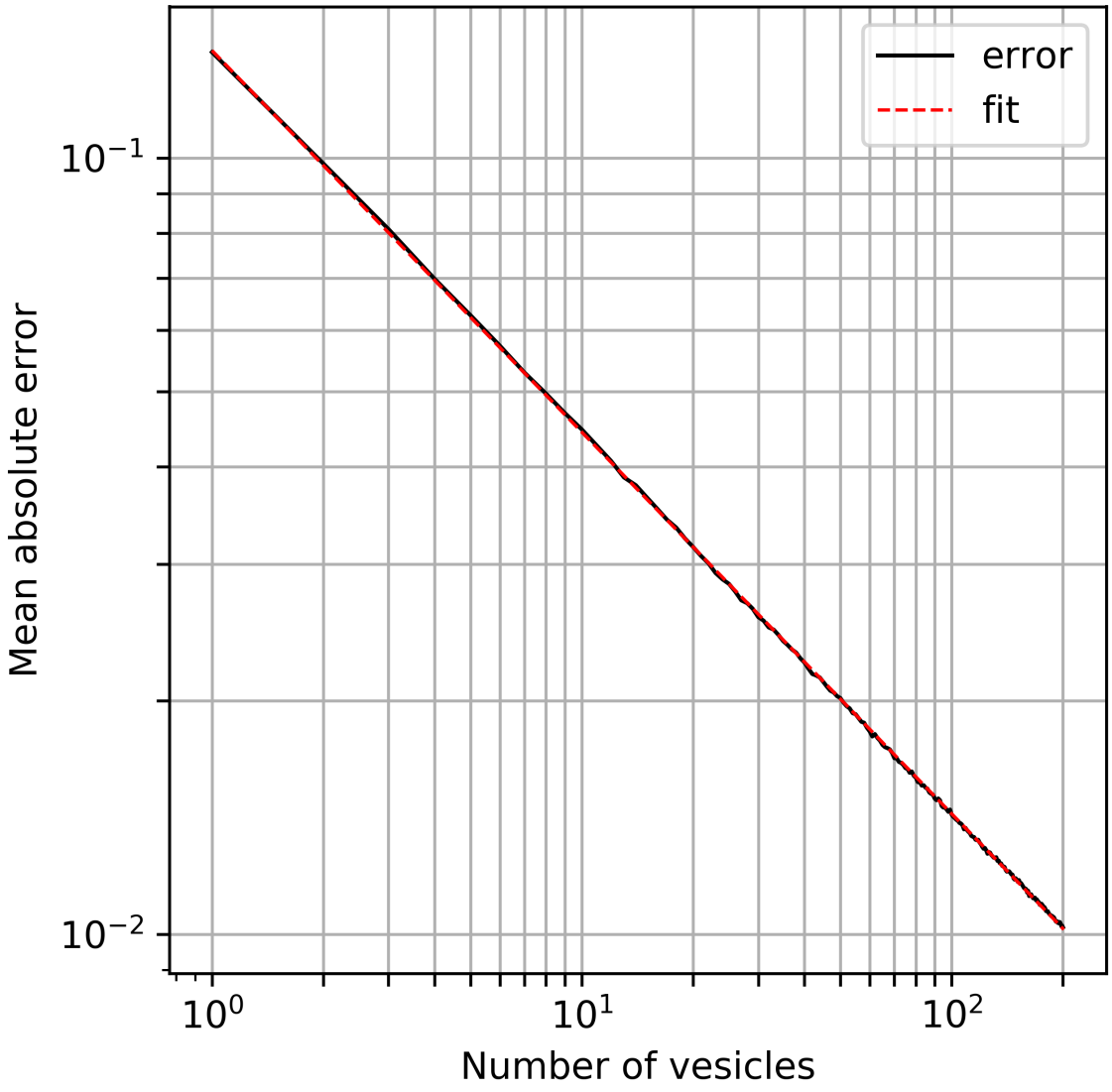


Figure 2.12: Error plot showing in black how the mean absolute error of the drift estimate is connected to the number of vesicles used in the estimate for a section. This plot was simulated using a bootstrapping approach sam-pling from the estimated drift components 50000 times for each point on the graph. For each sample, the drift was estimated, and the absolute error measured. This indicates an reciprocal convergence in the error as a function of the number of vesicles. In red is shown the fit ($y = 0.1375x^{-0.4915}$) of the bootstrapping data with

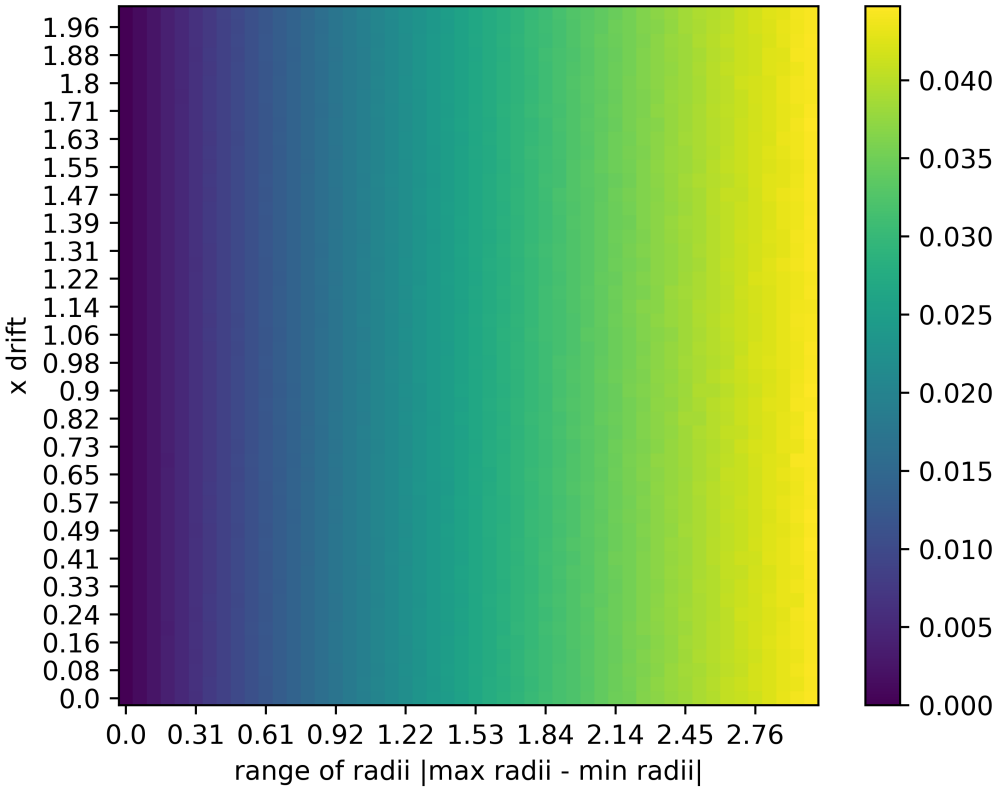


Figure 2.13: Error plot assessing the interdependence of drift magnitude and radii variation to the mean absolute estimation error. Drift was varied only along the x-axis for simplicity since the method is fundamentally independent of the direction of the drift. The radii were varied from 3 to 6 voxels, which is similar to estimates from the FIB-SEM dataset used in this work. The error seems to be independent on the magnitude of the drift.

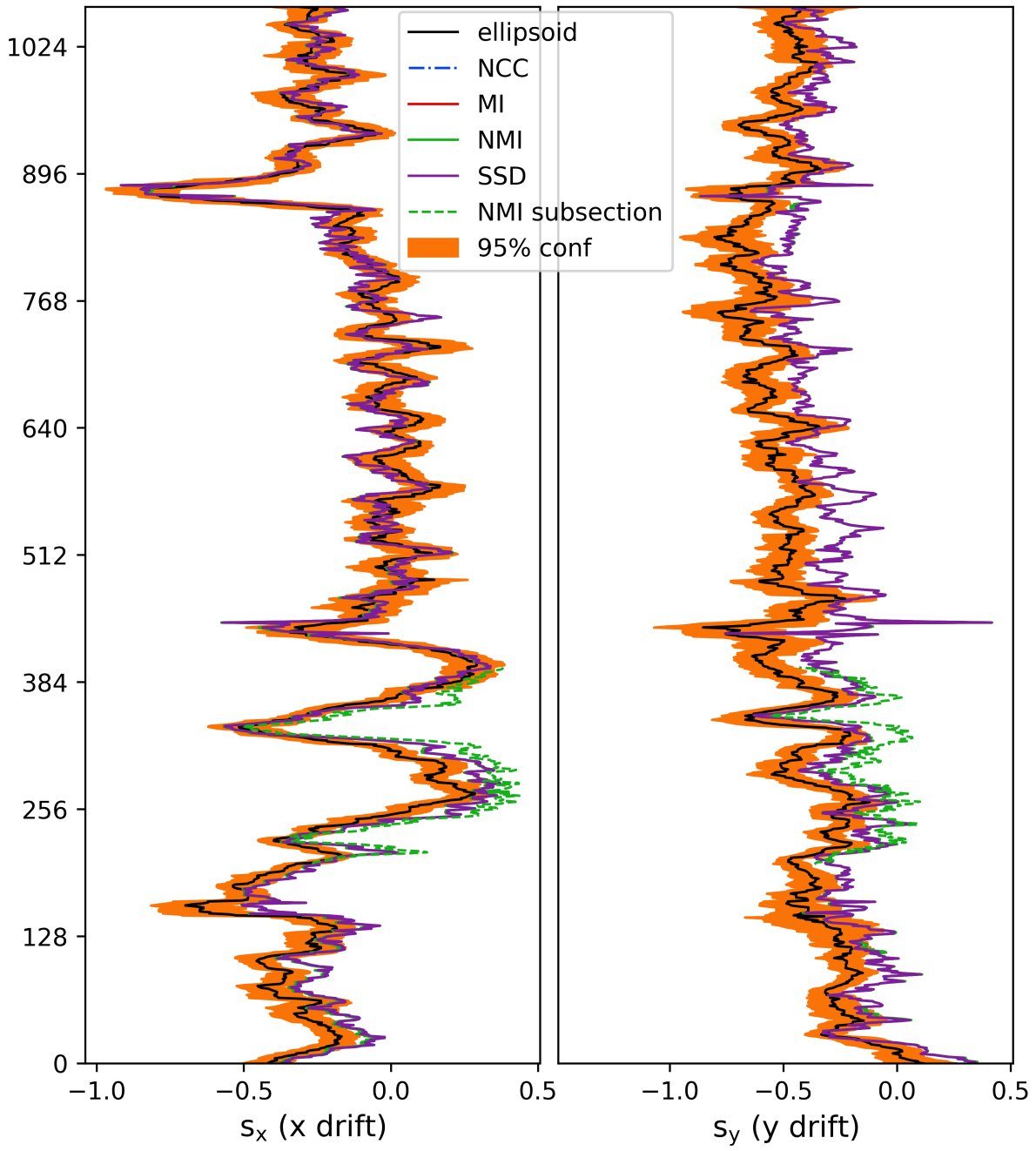


Figure 2.14: The complete drift estimate comparison in all 1065 sections of the FIB-SEM dataset. All sequences except the one denoted ellipsoid were convolved with a box kernel of size 11 to counter the noise in those approaches to make visual inspection possible. (Left) The drift estimate along the x-axis. (right) The drift estimate along the y-axis. For both graphs, NCC, MI and SSD coincide. Further, the bias for the standard, global measures appear to be most pronounced along the y-axis.

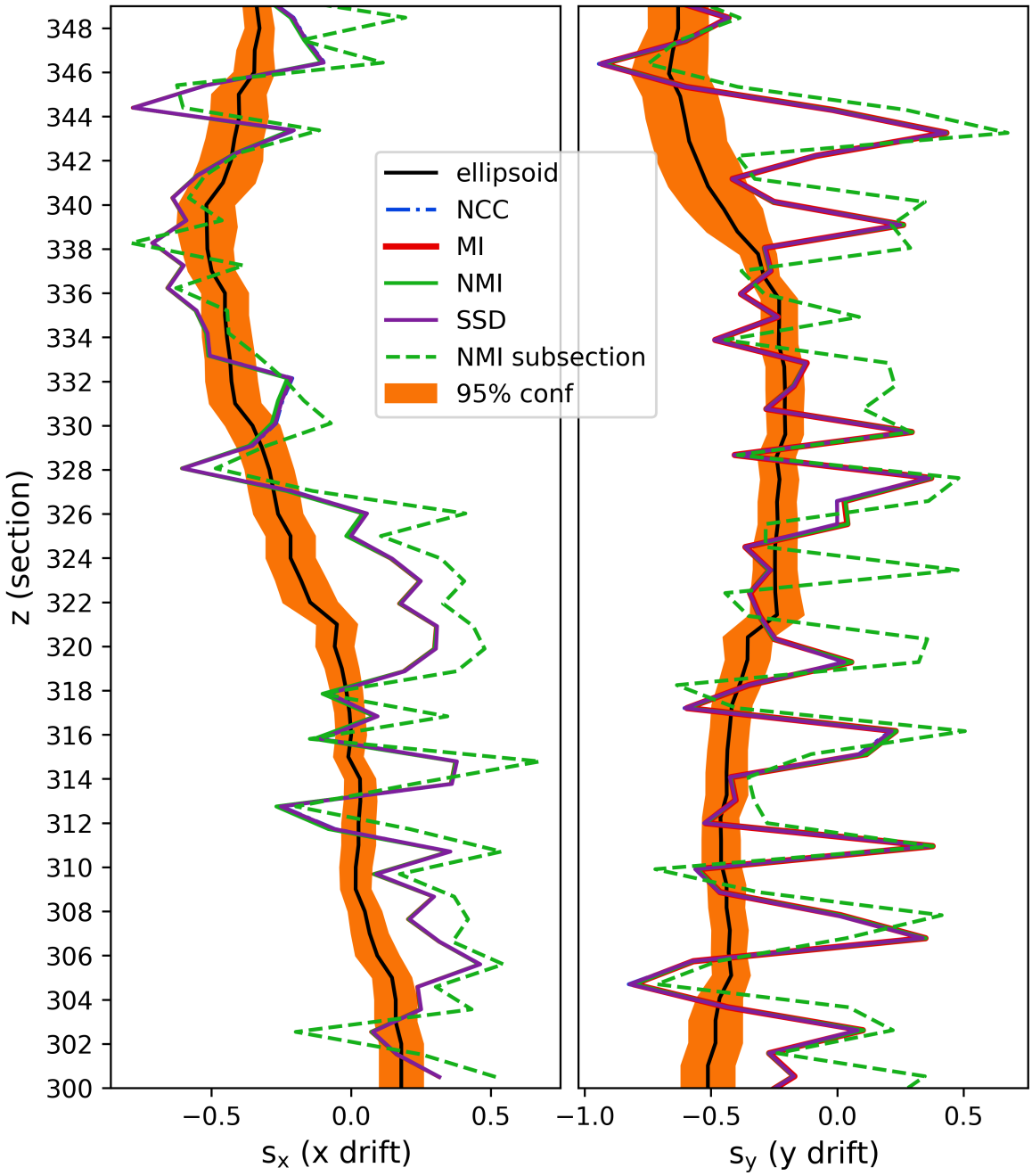


Figure 2.15: A small subsection of the drift estimate in the real dataset without any smoothing present. Here the rapidly varying drift estimate of the standard registration methods is visible.

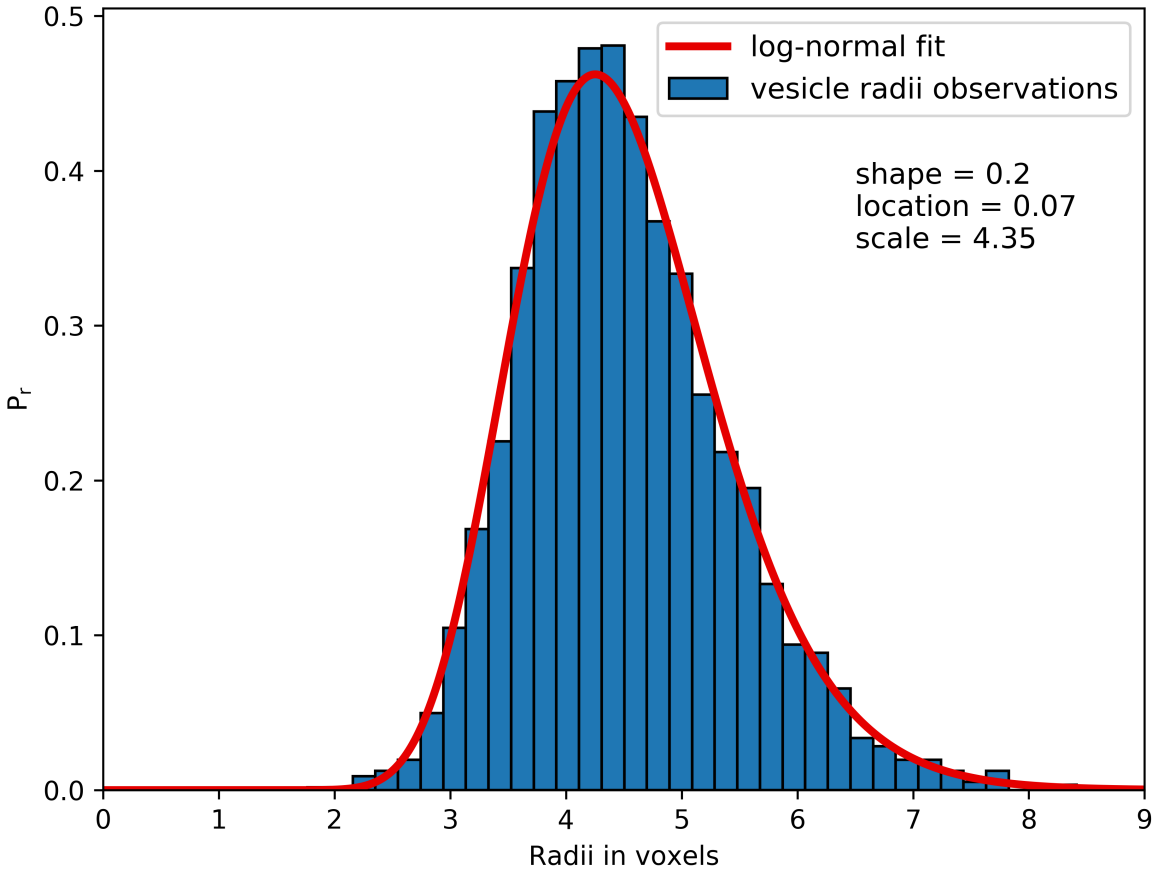


Figure 2.16: Distribution of ellipsoid radii after drift correction in the FIB-SEM dataset (https://cvlab.epfl.ch/data/data-em/). All radii of all the fitted ellipsoids are used to prevent sorting effects which occurs if largest to smallest radii are assessed separately.

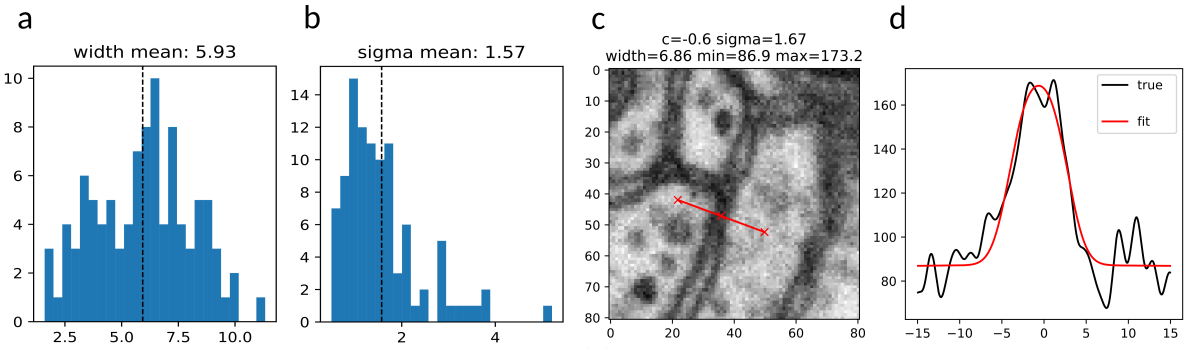
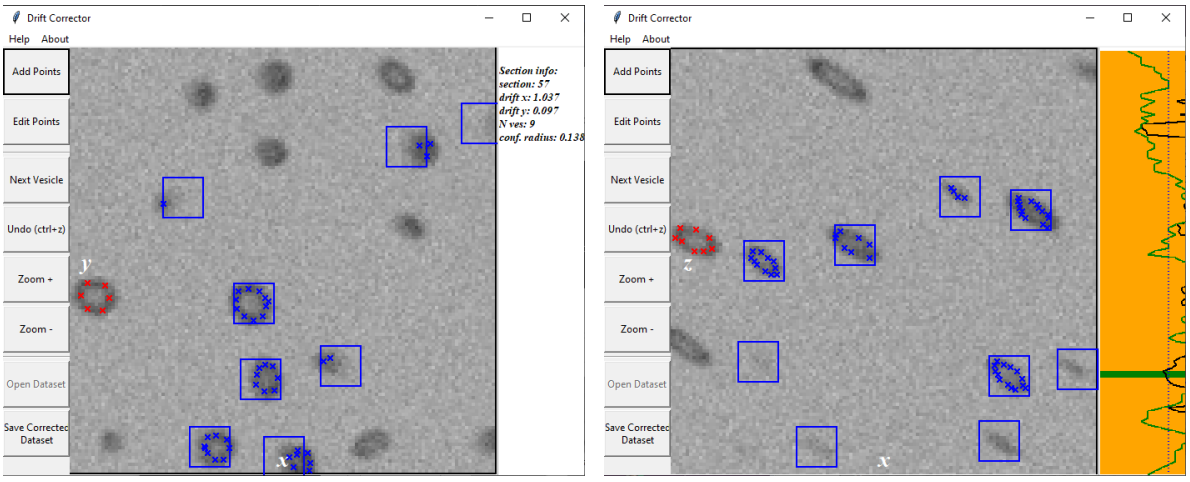


Figure 2.17: Example of boundary fittings used to estimate the amount of Gaussian blurring in the images. (**a**) Box function width parameters of the membrane fit. (**b**) Gaussian function standard deviation (σ) parameters of the membrane fit. (**c**) A single example of a measurement line for which to fit the blurred box function. (**d**) The (**c**) corresponding resampling across the line in black along with the fitted Gaussian blurred box function in red.



(a) Dataset front view (b) Dataset side view

Figure 2.18: The Drift Corrector user-interface shown as a 2.18a front-view and 2.18b side-view.

2.10 Post Publication Notes

Drift Corrector Software

The Drift Corrector Software [49] was written to easily annotate vesicles with instant feedback on the quality of the current estimated drift for each section. The software is implemented in Python using Tkinter for UI elements and pillow to read and write the image, and NumPy, SciPy, and Scikit-Image to handle the underlying calculations.

An example view of the Drift Corrector software can be seen in Figure 2.18. The dataset can be viewed from all axis planes x-y, x-z, and y-z respectively. Adding and removing points is done by clicking, changing viewing depth by scrolling, and changing axis view by clicking the A-key.

When viewing the dataset in the image plane (see Figure 2.18a), the right panel displays relevant information about that particular section. When viewing the dataset from one of the side-views (see Figure 2.18b), the right panel becomes a graphical display of the current certainty in the estimate across all the shown sections. The colors red/yellow/green indicate if the estimate is not possible, uncertain, or certain respectively. The green indication is only given if a) that a minimum of 10 vesicles is annotated and b) that the confidence of estimation is below a preset threshold.

Requests by the user, such as adding points to a vesicle and progressing the annotation to the next vesicle affect a Dataset manager object which, upon completion of each vesicle, requests a recalculation of the drift estimate. Since the Ellipsoid fitting algorithm is very fast, recalculation only takes a split second even after having annotated thousands of vesicles.

The annotation data is automatically saved after each vesicle annotation. When done annotating, producing an output image is handled by combining drift estimates, input data, and output paths into a transformed dataset. The coordinate mapping for image interpolation after transformation was performed using the warp functionality in the Scikit-Image python package.

Chapter 3

Measuring shape relations using r-parallel sets

3.1 Abstract

Geometrical measurements of biological objects form the basis of many quantitative analyses. Hausdorff measures such as the volume and the area of objects are simple and popular descriptors of individual objects, however, for most biological processes, the interaction between objects cannot be ignored, and the shape and function of neighboring objects are mutually influential.

In this paper, we present a theory on the geometrical interaction between objects inspired by K-functions for spatial point-processes. Our theory describes the relation between two objects: a reference and an observed object. We generate the r-parallel sets of the reference object, calculate the intersection between the r-parallel sets and the observed object, and define measures on these intersections. The measures are simple, like the volume or surface area, but describe further details about the shape of individual objects and their pairwise geometrical relation. Finally, we propose a summary-statistics.

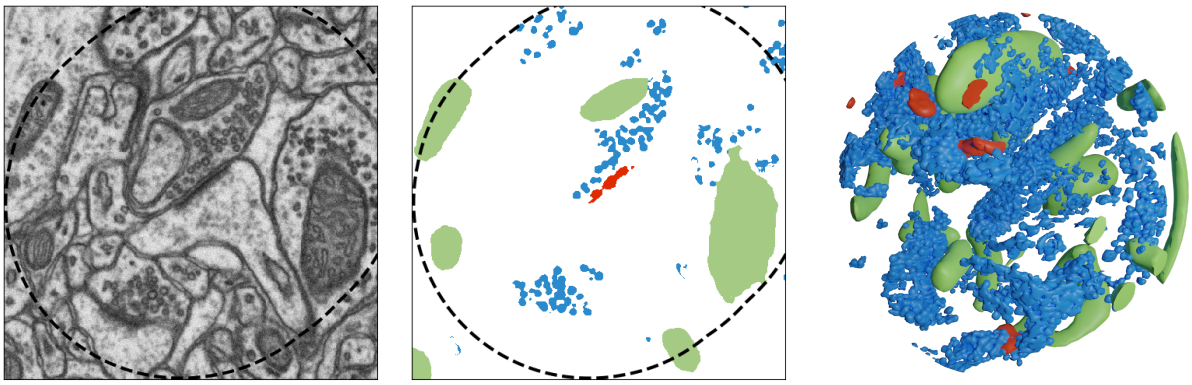
To evaluate these measures, we present a new segmentation of cell membrane, mitochondria, synapses, vesicles, and endoplasmic reticulum in a publicly available FIB-SEM 3D brain tissue data set and use our proposed method to analyze key biological structures herein.

Keywords: Multidimensional Shape Analysis ∘ Hausdorff measure ∘ r-parallel sets ∘ Cross-K function ∘ Germ-Grain Process

3.2 Introduction

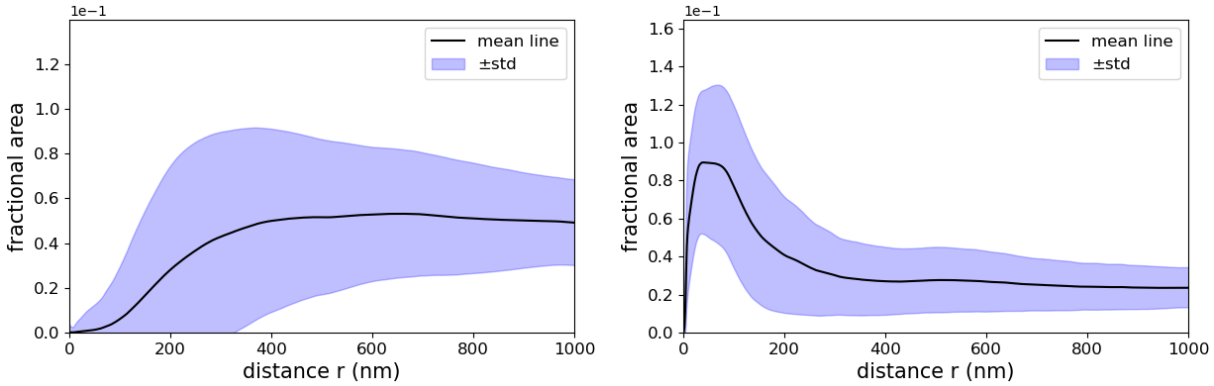
Measuring the geometry and statistics of objects is a fundamental tool used in all areas of the natural sciences. Geometric object-descriptors vary in complexity from simple measures such as point count, area, and volume to parameterized domain-specific shape models. See [62] for a review of shape representations.

In many cases, we are further interested in the relation between objects to answer questions like: How do synaptic vesicles distribute in the neighborhood of a synapse during stress [23]? How are astrocytes distributed w.r.t. the position and shapes of their nearby neuronal cells in amyotrophic lateral sclerosis [31]? What is the relation between the position and shape of the cartilage of the tibia and femur and osteoarthritis [30]? A simple approach is to summarize each object as a point and consider the set of points as



(a) An image slice. (b) The slice's segments. (c) A 3D neighbourhood of segments.

Figure 3.1: An example of the FIB-SEM data set and our segmentation. Red objects are synaptic regions, blue are vesicles, and green are mitochondria.



(a) L_{01} for mitochondria. (b) L_{01} for vesicles.

Figure 3.2: Normalized area comparison of vesicles and mitochondria as a distance to the synapse for a total of 365 synapses and their neighborhood vesicles and mitochondria.

a point-process, which has a well-developed theory and readily available software, e.g. [2]. However, this approach is limited since it does not take the geometry of individual objects into account, thereby ignoring important physical correspondence.

In this article, we propose a set of measures that consider the relation between different types of sets. For example, Figure 3.1 shows the segmentation of an electron microscopy image where different object types have been identified.

Our method measures the relation between reference and observed objects by first calculating one-parameter curves for each pair, and then summarizes these as summary-statistics curves, as illustrated in Figure 3.2. This figure shows the two summary-statistics curves for the relation between the synaptic regions and the mitochondria and vesicles, respectively. From these, we conclude that the mitochondria are rarely observed very close to the synaptic regions, and that there is a high density of vesicles near the synaptic regions. We discuss these figures in more detail in Section 3.8.

We base our method on the cross K-function from the statistics of point-processes, which describes the relation between paired point-processes [11]. A classic example often used to present the cross K-function is to model the occurrence of crimes and the loca-

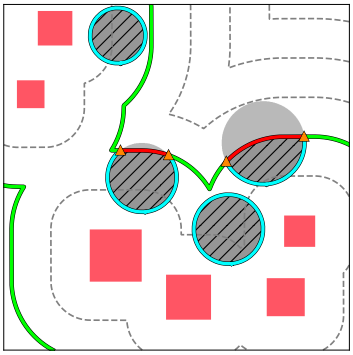


Figure 3.3: An example of how the measures emerge for some simple 2D shapes. The disks are the observed objects and the squares are the reference objects. The green line shows the boundary of the r-parallel, giving rise to one counting measure (orange triangles), two length measures (red and blue contours), and one area measure (hatched grey region).

tions of police stations by point-processes $\mathcal{X}$ and $\mathcal{Y}$, respectively. The cross K-function measures the expected number of crime occurrences $x \in \mathcal{X}$ within distance r of a police station $y \in \mathcal{Y}$ as

$$K(r) = \mathbb{E}_{y \in \mathcal{Y}} \big| \{x \in \mathcal{X} | d(x, y) \leq r\} \big|. \tag{3.1}$$

Here, $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^2$ are discrete point sets, d is a distance measure, often the Euclidean distance, $|\cdot|$ is the set-size operator, and $\mathbb{E}_{y \in \mathcal{Y}}$ is the conditional expectation given $y \in \mathcal{Y}$.

We extend the cross K-function to general geometric objects. We will consider objects $X, Y \subseteq \mathbb{R}^d$, which we will call the observed and the reference object, respectively. These objects may be points but could also be surfaces or solids. We extend the notion of distance to be the shortest distance between two objects. As an example, in Fig. 3.3 we show equidistant curves from the reference objects in red and how these distance curves interact with the observed objects in blue. To quantify the observed objects w.r.t. the reference objects, we calculate a family of sets Y^r consisting of all points within a distance r from Y, which we call the r-parallel sets. For each r-parallel set, we intersect it with the observed object and measure the number of points in the intersection of boundaries, the length of the inner and outer contours, and the area. These are also known as Hausdorff measures related to the intersection between Y^r and X.

Aspects of this method have seen use in literature. In [48], we propose 3 statistical summary measures for population of curves, one of which is the overlap between a curve with the mathematical dilation of another. In [31], they measure the density of astrocyte glial cells by a weighted distance from the postsynaptic density. In [17], they measure the distance at which the astrocyte volume-fraction peaks in relation to dendritic spines and axonal boutons. However, the theoretical foundation has been absent so far. Our contribution is to remedy this shortcoming and broaden the scope to include the full family of n-dimensional Hausdorff measures. To demonstrate the usefulness of our method, we present a segmentation of a publicly available FIB-SEM 3D data set of an adult rodent [27] in greater detail than previously available and use it as a subject for analysis using our proposed measures.

Table 3.1: Interpretation of $\mu_{\varepsilon,\varepsilon'}$ in 2D and 3D as a measure on the intersection $X \cap Y^r$. Shapes and colors in the left column refers to Figure 3.3.

$d=2$	$(\varepsilon,\varepsilon')$	$\mathcal{H}^{d-\varepsilon-\varepsilon'}$	$\partial^\varepsilon X \cap \partial^{\varepsilon'} Y^r$	Interpretation of $\mu_{\varepsilon,\varepsilon'}(X,Y^r)$
	$(0,0)$	Area	$X \cap Y^r$	Area of intersection
	$(0,1)$	Curve length	$X \cap \partial Y^r$	Boundary length of intersection inside interior of X
	$(1,0)$	Curve length	$\partial X \cap Y^r$	Boundary length of intersection inside boundary of X
	$(1,1)$	Point counts	$\partial X \cap \partial Y^r$	Number of points in intersection of boundaries
$d=3$	$(\varepsilon,\varepsilon')$	$\mathcal{H}^{d-\varepsilon-\varepsilon'}$	$\partial^\varepsilon X \cap \partial^{\varepsilon'} Y^r$	Interpretation of $\mu_{\varepsilon,\varepsilon'}(X,Y^r)$
	$(0,0)$	Volume	$X \cap Y^r$	Volume of intersection
	$(0,1)$	Surface area	$X \cap \partial Y^r$	Surface area of intersection inside interior of X
	$(1,0)$	Surface area	$\partial X \cap Y^r$	Surface area of intersection inside boundary of X
	$(1,1)$	Curve length	$\partial X \cap \partial Y^r$	Length of intersection of boundaries

3.3 Method Description

We would like to measure the shape and position of an observed object $X \subseteq \mathbb{R}^d$ with respect to some reference object $Y \subseteq \mathbb{R}^d$. For $r > 0$, we consider the part of X that lies within distance r from Y. This is the set $X \cap Y^r$, where Y^r is the set of all points within distance r from Y. This is called the *r-parallel set* of Y and is given by

$$Y^r = \{\alpha \in \mathbb{R}^d \mid \inf_{y \in Y} d(\alpha, y) \leq r\}. \tag{3.2}$$

Here $d(\alpha, y)$ denotes the distance between α and y, typically the Euclidean distance.

We measure $X \cap Y^r$ by a measure $\mu(X, Y^r)$. In all our applications, μ has the form

$$\mu_{\varepsilon,\varepsilon'}(X, Y^r) = \mathcal{H}^{d-\varepsilon-\varepsilon'}(\partial^\varepsilon X \cap \partial^{\varepsilon'} Y^r), \tag{3.3}$$

where $\mathcal{H}^k$ denotes the k-dimensional Hausdorff measure, $\varepsilon, \varepsilon' \in \{0, 1\}$, and for a closed set $C \subseteq \mathbb{R}^d$, $\partial^0 C = C$ is just C itself, while $\partial^1 C = \partial C$ is the boundary of C. The interpretation of $\mathcal{H}^k$ and $\mu_{\varepsilon,\varepsilon'}(X, Y^r)$ in 2D and 3D is shown in Table 3.1. Since the boundary of a boundary is the empty set, i.e., $\partial\partial^\bullet = \emptyset$, this is the complete list of measures in 2 and 3 dimensions, and these lists generalize naturally to any dimension.

The measure $\mu_{\varepsilon,\varepsilon'}(X, Y^r)$ is a function of r. Under mild conditions on the shapes X and Y, μ_{00}, μ_{01}, and μ_{10} are continuous and

$$\frac{d}{dr}\mu_{00}(X, Y^r) = \mu_{01}(X, Y^r), \tag{3.4}$$

for almost all values of r. For instance, this holds for the example shapes considered in Section 3.3.2.

In applications, we typically observe a collection of objects spread out in space. To model a collection of objects, we equip each object with a reference point. The reference point can for instance be chosen geometrically as the center of mass or the center of the smallest ball containing the object. At other times, the application provides a natural

center point such as the nucleus of a cell. Thus, we obtain a marked point-process $\mathcal{X} = \{x_i, X_i\}_{i\geq 0}$ on $\mathbb{R}^d \times \mathcal{C}$, where the mark space $\mathcal{C}$ is the space of all compact sets in $\mathbb{R}^d$ with a smooth boundary. The point x_i is the reference point, and the associated mark X_i can be thought of as the shape. Such a marked point-process is also known as a *germ-grain process* [42]. The collection of objects, sometimes called a *germ-grain model*, is then

$$\bigcup_{i\geq 0}(x_i + X_i), \tag{3.5}$$

where $v + X = \{v + x \mid x \in X\}$.

Let $\mathcal{X}, \mathcal{Y}$ be germ-grain processes modelling the observed and reference objects, respectively. Writing $v + \mathcal{X} = v + \{x_i, X_i\}_{i\geq 0} = \{v + x_i, X_i\}_{i\geq 0}$, we say that the processes are *jointly stationary* if $(v + \mathcal{X}, v + \mathcal{Y})$ has the same distribution as $(\mathcal{X}, \mathcal{Y})$ for any translation vector $v \in \mathbb{R}^d$. In the following, we let $\bar{\mathcal{X}}$ and $\bar{\mathcal{Y}}$ denote the point-processes of reference points underlying $\mathcal{X}$ and $\mathcal{Y}$, respectively, and let $\rho_{\mathcal{X}}$ and $\rho_{\mathcal{Y}}$ denote their spatial intensities.

A global functional summary statistic is given by

$$K_{\varepsilon,\varepsilon'}(r) = \frac{1}{\rho_{\mathcal{X}}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\left(\sum_{(x,X)\in\mathcal{X}} \mu_{\varepsilon,\varepsilon'}(x + X, Y^r)\right). \tag{3.6}$$

The interpretation of $K_{\varepsilon,\varepsilon'}(r)$ is that, conditioned on Y having a reference point at the origin, it is the expected value of $\mu_{\varepsilon,\varepsilon'}(x + X, Y^r)$ summed over all particles $x + X$ within distance r from the object Y averaged by the expected number of particles from $\mathcal{X}$ pr. unit volume. An estimator is:

$$\begin{aligned}\hat{K}_{\varepsilon,\varepsilon'}(r) =& \frac{1}{\mathcal{H}^d(W)\rho_{\mathcal{X}}\rho_{\mathcal{Y}}} \sum_{(y,Y)\in\mathcal{Y}} \mathbb{1}_{\{y\in W\}} \\ &\times \sum_{(x,X)\in\mathcal{X}} \mu_{\varepsilon,\varepsilon'}(x + X, y + Y^r).\end{aligned} \tag{3.7}$$

Here $\mathcal{H}^d(W)$ is the volume of the sampling window. Note that we only sample reference objects with reference point $y \in W$, but in order to compute $\hat{K}_{\varepsilon,\varepsilon'}(r)$, we must be able to observe all of the associated Y^r even if it intersects the boundary of W, which may not be possible in practice. In Section 3.3.1, we present edge-correction strategies for handling cases where the r-parallel exceeds the observation window W.

Theorem 1. *If $(\mathcal{X}, \mathcal{Y})$ are jointly stationary, then $\hat{K}_{\varepsilon,\varepsilon'}$ is an unbiased estimator for $K_{\varepsilon,\varepsilon'}$.*

Proof. The expected value of $\hat{K}_{\varepsilon,\varepsilon'}$ is

$$\begin{aligned}
&\frac{1}{\mathcal{H}^d(W)\rho_{\mathcal{X}}\rho_{\mathcal{Y}}}\mathbb{E}\sum_{(y,Y)\in\mathcal{Y}}\mathbb{1}_{\{y\in W\}} \\
&\qquad\times\sum_{(x,X)\in\mathcal{X}}\mu_{\varepsilon,\varepsilon'}(x+X,y+Y^r) &(3.8)\\
&=\frac{1}{\mathcal{H}^d(W)\rho_{\mathcal{X}}\rho_{\mathcal{Y}}}\int_W\rho_{\mathcal{Y}}\mathbb{E}_{y\in\bar{\mathcal{Y}}}\Bigg(\\
&\qquad\sum_{(x,X)\in\mathcal{X}}\mu_{\varepsilon,\varepsilon'}(x+X,y+Y^r)\Bigg)dy &(3.9)\\
&=\frac{1}{\mathcal{H}^d(W)\rho_{\mathcal{X}}\rho_{\mathcal{Y}}}\int_W\rho_{\mathcal{Y}}\mathbb{E}_{y\in\bar{\mathcal{Y}}}\Bigg(\\
&\qquad\sum_{(x,X)\in\mathcal{X}}\mu_{\varepsilon,\varepsilon'}(x-y+X,Y^r)\Bigg)dy &(3.10)\\
&=\frac{1}{\mathcal{H}^d(W)\rho_{\mathcal{X}}\rho_{\mathcal{Y}}}\int_W\rho_{\mathcal{Y}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\\
&\qquad\sum_{(x-y,X)\in\mathcal{X}}\mu_{\varepsilon,\varepsilon'}(x-y+X,Y^r)\Bigg)dy &(3.11)\\
&=\frac{1}{\rho_{\mathcal{X}}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\sum_{(x,X)\in\mathcal{X}}\mu_{\varepsilon,\varepsilon'}(x+X,Y^r)\Bigg), &(3.12)
\end{aligned}$$

which is $K_{\varepsilon,\varepsilon'}(r)$. In (3.10), we used the translation invariance of $\mu_{\varepsilon,\varepsilon'}$, and in (3.11), we used the joint stationarity of the point processes. □

The normalization only by the intensity $\rho_{\mathcal{X}}$ may seem unnatural since the resulting K-function will depend on the Hausdorff measure of the objects in $(\mathcal{X},\mathcal{Y})$. As in [39], we, therefore, also consider the normalized measures,

$$\begin{aligned}
N_{\varepsilon'}(Y^r) &= \mathcal{H}^{d-\varepsilon'}(\partial^{\varepsilon'}Y^r), &(3.13)\\
\nu_{\varepsilon,\varepsilon'}(X,Y^r) &= \frac{\mu_{\varepsilon,\varepsilon'}(X,Y^r)}{N_{\varepsilon'}(Y^r)}, &(3.14)\\
L_{\varepsilon,\varepsilon'}(r) &= \frac{1}{\rho_{\mathcal{X}}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\sum_{(x,X)\in\mathcal{X}}\nu_{\varepsilon,\varepsilon'}(x+X,Y^r)\Bigg), &(3.15)\\
\hat{L}_{\varepsilon,\varepsilon'}(r) &= \frac{1}{\mathcal{H}^d(W)\rho_{\mathcal{X}}\rho_{\mathcal{Y}}}\sum_{(y,Y)\in\mathcal{Y}}\mathbb{1}_{\{y\in W\}} \\
&\qquad\times\sum_{(x,X)\in\mathcal{X}}\nu_{\varepsilon,\varepsilon'}(x+X,y+Y^r). &(3.16)
\end{aligned}$$

3.3.1 Edge corrections

In practice, we observe objects inside a finite window W, so instead of the measures $\mu_{\varepsilon,\varepsilon'}(X,Y^r)$, we are only able to measure

$$\mu^W_{\varepsilon,\varepsilon'}(X,Y^r)=\mathcal{H}^{d-\varepsilon-\varepsilon'}(\partial^{\varepsilon}X\cap\partial^{\varepsilon'}Y^r\cap W).$$

If these $\mu^W_{\varepsilon,\varepsilon'}$ are used instead of $\mu_{\varepsilon,\varepsilon'}$ in (3.7), then $\hat{K}_{\varepsilon,\varepsilon'}$ is no longer an unbiased estimator for (3.12).

We solve this problem by introducing edge corrections in (3.7). That is, in the definition of $\hat{K}_{\varepsilon,\varepsilon'}$, we replace the measures $\mu_{\varepsilon,\varepsilon'}(X, Y^r)$ by edge corrected measures $e_{\varepsilon,\varepsilon'}(y, X, Y^r)$ and obtain

$$\begin{aligned}\hat{K}^e_{\varepsilon,\varepsilon'}(r) =& \frac{1}{\mathcal{H}^d(W)\rho_{\mathcal{X}}\rho_{\mathcal{Y}}} \sum_{(y,Y)\in\mathcal{Y}} \mathbb{1}_{\{y\in W\}} \\ & \times \sum_{(x,X)\in\mathcal{X}} e_{\varepsilon,\varepsilon'}(y, x+X, y+Y^r). \end{aligned} \tag{3.17}$$

We suggest the following three edge corrections, based on the most common edge corrections for point-process K-functions [40].

1. Sampling in a smaller window: Here, we sample only particles from $\mathcal{Y}$ with a reference point inside a smaller window where we have full information about the objects and correct for the reduced window size.

   As a sampling window, we may take

   $$W \ominus B_M(0) = \{z \in \mathbb{R}^d \mid z + B_M(0) \subseteq W\}. \tag{3.18}$$

   Here $B_s(c)$ denotes the Euclidean ball of radius s centered at c, and $M = m_y + r$, where m_y is an upper bound on the diameter of objects from $\mathcal{Y}$.

   The resulting estimator corresponds to (3.17) with edge corrections

   $$\begin{aligned} & e_{\varepsilon,\varepsilon'}(y, X, Y^r) = \\ & \quad \mathbb{1}_{\{y\in W\ominus B_M(0)\}} \frac{\mathcal{H}^d(W)}{\mathcal{H}^d(W \ominus B_M(0))} \mu_{\varepsilon,\varepsilon'}(X, Y^r). \end{aligned} \tag{3.19}$$

   This approach yields an unbiased estimator for $\hat{K}_{\varepsilon,\varepsilon'}$ whenever $\mathcal{X}$ and $\mathcal{Y}$ are jointly stationary. However, this will be an inefficient estimator since we throw away information, especially for small sampling windows or large objects.

2. A translative correction: Here, we replace the measure $\mu_{\varepsilon,\varepsilon'}(X, Y^r)$ by an integral over the domain

   $$\begin{aligned} e_{\varepsilon,\varepsilon'}(y, X, Y^r) = & \int_{\partial^\varepsilon X \cap \partial^{\varepsilon'} Y^r \cap W} \\ & \frac{\mathcal{H}^d(W)}{\mathcal{H}^d(W \cap (W - w + y))} \mathcal{H}^{d-\varepsilon-\varepsilon'}(dw). \end{aligned} \tag{3.20}$$

   Theorem 2. *For jointly stationary point-processes $(\mathcal{X}, \mathcal{Y})$, the edge corrections in (3.20) yield an unbiased estimator for $K_{\varepsilon,\varepsilon'}(r)$.*

   In the proofs below, we use the notation

   $$\Omega = \partial^\varepsilon(x + X) \cap \partial^{\varepsilon'}(y + Y^r) \cap W \tag{3.21}$$

   for brevity.

Proof. The expectation of $\hat{K}^e_{\varepsilon,\varepsilon'}(r)$ is

$$\mathbb{E}\bigg(\frac{1}{\rho_{\mathcal{X}}\rho_{\mathcal{Y}}\mathcal{H}^d(W)}\sum_{(y,Y)\in\mathcal{Y}}\mathbb{1}_{\{y\in W\}} \sum_{(x,X)\in\mathcal{X}}\int_\Omega \frac{\mathcal{H}^d(W)}{\mathcal{H}^d(W\cap(W-w+y))}\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw)\bigg).$$

By the stationarity assumption, this equals

$$\begin{aligned}
&\frac{1}{\rho_{\mathcal{X}}}\int_W \mathbb{E}_{0\in\bar{\mathcal{Y}}}\bigg(\sum_{(x,X)\in\mathcal{X}}\int_{\partial^\varepsilon(x+X)\cap\partial^{\varepsilon'}Y^r\cap(W-y)} \frac{1}{\mathcal{H}^d(W\cap(W-w))}\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw)\bigg)dy\\
=&\frac{1}{\rho_{\mathcal{X}}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\bigg(\sum_{(x,X)\in\mathcal{X}}\int_{\partial^\varepsilon(x+X)\cap\partial^{\varepsilon'}Y^r} \int_W \frac{\mathbb{1}_{W-y}(w)}{\mathcal{H}^d(W\cap(W-w))}dy\,\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw)\bigg), \qquad (3.22)\\
=&\frac{1}{\rho_{\mathcal{X}}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\bigg(\sum_{(x,X)\in\mathcal{X}}\mu_{\varepsilon,\varepsilon'}(x+X,Y^r)\bigg), \qquad (3.23)
\end{aligned}$$

which we recognize as $K_{\varepsilon,\varepsilon'}(r)$. □ □

The disadvantage of this edge correction is that the simple Hausdorff measures are replaced by integrals, which will slow down computations.

3. An isotropic correction: Here we replace the measure $\mu_{\varepsilon,\varepsilon'}(X,Y^r)$ by the integral

$$e_{\varepsilon,\varepsilon'}(y,X,Y^r) = \int_{\partial^\varepsilon X\cap\partial^{\varepsilon'}Y^r\cap W} \frac{\mathcal{H}^{d-1}(\partial B_{|w-y|}(y))}{\mathcal{H}^{d-1}(\partial B_{|w-y|}(y)\cap W)}\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw). \qquad (3.24)$$

We say that two germ-grain processes $(\mathcal{X},\mathcal{Y})$ are *jointly isotropic* if $(\{Rx_i,RX_i\}_{i\geq0},\{Ry_i,RY_i\}_{i\geq0})$ has the same distribution as $(\{x_i,X_i\}_{i\geq0},\{y_i,Y_i\}_{i\geq0})$ for any rotation $R\in\mathrm{SO}(d)$.

Theorem 3. *When $(\mathcal{X},\mathcal{Y})$ are both jointly stationary and isotropic, the edge corrections in* (3.24) *yield an unbiased estimator of* (3.12).

Proof. As before, we compute the mean of $\hat{K}_{\varepsilon,\varepsilon'}(r)$:

$$\begin{aligned}
&\mathbb{E}\Bigg(\frac{1}{\rho_{\mathcal{X}}\rho_{\mathcal{Y}}\mathcal{H}^d(W)}\sum_{(y,Y)\in\mathcal{Y}}\mathbb{1}_{\{y\in W\}}\sum_{(x,X)\in\mathcal{X}}\\
&\qquad\int_\Omega\frac{\mathcal{H}^{d-1}(\partial B_{|w-y|}(y))}{\mathcal{H}^{d-1}(\partial B_{|w-y|}(y)\cap W)}\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw)\Bigg)\\
=&\frac{1}{\rho_{\mathcal{X}}\mathcal{H}^d(W)}\int_W\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\sum_{(x,X)\in\mathcal{X}}\\
&\qquad\int_{\partial^\varepsilon(x+X)\cap\partial^{\varepsilon'}Y^r\cap(W-y)}\\
&\qquad\frac{\mathcal{H}^{d-1}(\partial B_{|w|}(y))}{\mathcal{H}^{d-1}(\partial B_{|w|}(y)\cap W)}\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw)\Bigg)dy. \qquad (3.25)
\end{aligned}$$

Letting ν denote the normalized Haar measure on $\mathrm{SO}(d)$, the isotropy assumption yields

$$\begin{aligned}
&\frac{1}{\rho_{\mathcal{X}}\mathcal{H}^d(W)}\int_{\mathrm{SO}(d)}\int_W\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\sum_{(x,X)\in\mathcal{X}}\\
&\qquad\int_{R^{-1}(\partial^\varepsilon(x+X)\cap\partial^{\varepsilon'}Y^r)}\mathbb{1}_W(Rw+y)\\
&\qquad\frac{\mathcal{H}^{d-1}(\partial B_{|Rw|}(y))}{\mathcal{H}^{d-1}(\partial B_{|Rw|}(y)\cap W)}\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw)\Bigg)dy\,\nu(dR)\\
=&\frac{1}{\rho_{\mathcal{X}}\mathcal{H}^d(W)}\int_W\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\sum_{(x,X)\in\mathcal{X}}\int_{\partial^\varepsilon(x+X)\cap\partial^{\varepsilon'}Y^r}\\
&\qquad\int_{\mathrm{SO}(d)}\mathbb{1}_W(Rw+y)\nu(dR)\\
&\qquad\frac{\mathcal{H}^{d-1}(\partial B_{|w|}(y))}{\mathcal{H}^{d-1}(\partial B_{|w|}(y)\cap W)}\mathcal{H}^{d-\varepsilon-\varepsilon'}(dw)\Bigg)dy \qquad (3.26)\\
=&\frac{1}{\rho_{\mathcal{X}}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\sum_{(x,X)\in\mathcal{X}}\mathcal{H}^{d-\varepsilon-\varepsilon'}(\partial^\varepsilon(x+X)\cap\partial^{\varepsilon'}Y^r)\Bigg) \qquad (3.27)\\
=&\frac{1}{\rho_{\mathcal{X}}}\mathbb{E}_{0\in\bar{\mathcal{Y}}}\Bigg(\sum_{(x,X)\in\mathcal{X}}\mu_{\varepsilon,\varepsilon'}(x+X,Y^r)\Bigg) \qquad (3.28)
\end{aligned}$$

which is $K_{\varepsilon,\varepsilon'}(r)$. □ □

This estimator again has the disadvantage that the Hausdorff measures are replaced by integrals. However,

$$\mathcal{H}^{d-1}(\partial B_{|w-y|}(y))/\mathcal{H}^{d-1}(\partial B_{|w-y|}(y)\cap W)=1 \qquad (3.29)$$

whenever y is further away from the boundary than $|y-w|$, so (3.24) reduces to $\mu_{\varepsilon,\varepsilon'}(X,Y^r)$ whenever $y\in W\ominus B_{m_y+r}(0)$. We therefore only need to calculate the edge corrections of the y that are close to the boundary.

Table 3.2: Exact results for simple shapes

	$\Lambda = \mathbb{R}^2$, $\theta = 2\arccos(1 - r/R)$		$\Lambda = \mathbb{R}^3$	
$\mu_{00} =$	$\begin{cases} \frac{R^2}{2}(\theta - \sin\theta), \text{ if } r < 2R \\ \pi R^2, \text{ otherwise} \end{cases}$	(3.30)	$\begin{cases} \frac{\pi r^2(3R-r)}{3}, \text{ if } r < 2R \\ \frac{4\pi R^3}{3}, \text{ otherwise} \end{cases}$	(3.31)
$\mu_{01} =$	$\begin{cases} 2R\sin\frac{\theta}{2}, \text{ if } r < 2R \\ 0, \text{ otherwise} \end{cases}$	(3.32)	$\begin{cases} \pi r(2R - r), \text{ if } r < 2R \\ 0, \text{ otherwise} \end{cases}$	(3.33)
$\mu_{10} =$	$\begin{cases} \theta R, \text{ if } r < 2R \\ 2\pi R, \text{ otherwise} \end{cases}$	(3.34)	$\begin{cases} 2\pi Rr, \text{ if } r < 2R \\ 4\pi R^2, \text{ otherwise} \end{cases}$	(3.35)
$\mu_{11} =$	$\begin{cases} 1, \text{ if } r = 0 \text{ or } r = 2R \\ 2, \text{ if } 0 < r < 2R \\ 0, \text{ otherwise} \end{cases}$	(3.36)	$\begin{cases} 0, \text{ if } r = 0 \text{ or } r = 2R \\ 2\pi\sqrt{2Rr - r^2}, \text{ if } 0 < r < 2R \\ 0, \text{ otherwise} \end{cases}$	(3.37)

3.3.2 Examples of simple object relations

For simple objects, we can evaluate $\mu_{\varepsilon,\varepsilon'}$ analytically. As an example, consider an infinite line/plane in 2D/3D as a reference object and a circle/sphere inside W of radius R as an observed object. Assume that the center of the circle/sphere is at distance R from the line/plane. Then for $r \geq 0$, we get the closed-form expressions given in Table 3.2 by using the formulas for a circular segment and spherical cap, respectively. In this example, $N_{\varepsilon'} = \infty$, hence the normalized functions $\nu_{\varepsilon,\varepsilon'}$ are all 0. We use the expressions in Table 3.2 for verification in Section 3.6.

3.4 Shape Equivalence under the Measures

Representing objects by the measure $\mu_{\varepsilon,\varepsilon'}(X, Y^r)$ encodes selected characteristics of the shape of X. In this section, we investigate to which extend the measures $\mu_{\varepsilon,\varepsilon'}(X, Y^r)$ with a fixed reference object Y determines the observed object X uniquely. Let $S \subseteq \mathcal{C}$ denote the class of possible shapes. For a fixed Y, this leads to an equivalence class $[X_i]$ for each $X_i \in S$ where

$$[X_i] = \{X_j \in S \mid \forall(r, \varepsilon, \varepsilon') : \mu_{\varepsilon,\varepsilon'}(X_i, Y^r) = \mu_{\varepsilon,\varepsilon'}(X_j, Y^r)\}\,. \tag{3.38}$$

We examine the special case $d = 2$ where the reference object Y is the line $x = 0$ meaning that the r-parallels are the sets given by $-r \leq x \leq r$. We consider the class of equivalent shapes S emerging from an object X that can be described solely by two functions $f(x)$ and $g(x)$ on the interval $[a, b]$. Here we let f, g represent the upper and lower contour of X, respectively, with $f(x) > g(x)$ for all x in $[a, b]$. Further, we require $f(a) = g(a)$ and $f(b) = g(b)$ such that the object is closed. Thus, the measures $\mu_{\varepsilon,\varepsilon'}(x)$ are

$$\mu_{00}(x) = \int_a^x (f(x') - g(x'))dx' \tag{3.39}$$

$$\mu_{01}(x) = f(x) - g(x) \tag{3.40}$$

$$\mu_{10}(x) = \int_a^x (\sqrt{1 + f'(x)^2} + \sqrt{1 + g'(x)^2})dx \tag{3.41}$$

$$\mu_{11}(x) = 2 \tag{3.42}$$

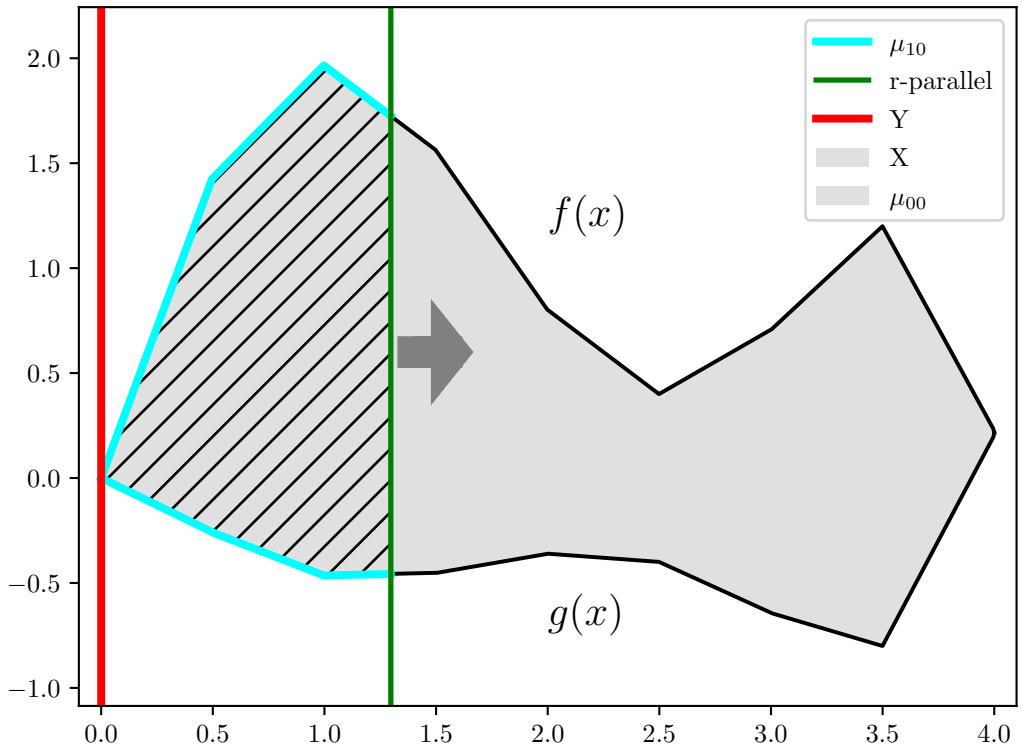


Figure 3.4: Example of an object from two piecewise linear contour functions.

We see that μ_{00} and μ_{01} are related through (3.4), i.e., μ_{01} describes the instantaneous change of μ_{00}. Hence, they constrain the equivalence class of X through f and g equally, with both measures requiring $f(x) - g(x)$ to be fixed for each x. The derivatives $f'(x)$ and $g'(x)$ are constrained by the arc-length of $f(x)$ and $g(x)$ through μ_{10}. Since μ_{10} is the sum of two arc-lengths, ignoring everything else, we could shorten the length of f by lengthening g or vice versa. Similarly, since the sign of the derivative of f and g does not change the arc-length, this measure allows for all combinations of the sign of the derivative. However, in combination with the other measures, we can narrow the set of equivalent shapes down to two possible continuations of the object in every point x. We illustrate this first by assuming f and g are piecewise linear functions (see Fig. 3.4). Consider a linear segment of the object on a subinterval from x_0 to x_1 with $\Delta x = x_1 - x_0$. Let $L = \mu_{10}(x_1) - \mu_{10}(x_0)$ denote the total length of the two linear functions on this subinterval and introduce a length splitting parameter t, such that the length of f and g on the subinterval is t and $L - t$ respectively, (see Fig. 3.5). For any $\mu_{01}(x_0)$, letting t and $\mu_{01}(x_1)$ vary we get a shape space parameterized by t and $\mu_{01}(x_1)$ in which the equivalent shapes are given as the solutions to the equation

$$\mu_{01}(x_1) = \mu_{01}(x_0) \pm \sqrt{t^2 - \Delta x^2} \pm \sqrt{(L-t)^2 - \Delta x^2}. \tag{3.43}$$

The solutions in the shape space can be seen in Fig. 3.6. There's no solution for $L < 2\Delta x$. For $L = 2\Delta x$ there is one solution corresponding to two horizontal lines. For $L > 2\Delta x$, a figure-eight shape appears, flattening at the top for increasing ratio of L to Δx. We see that two possible solutions for each $\mu_{01}(x_1)$ exist converging at unique

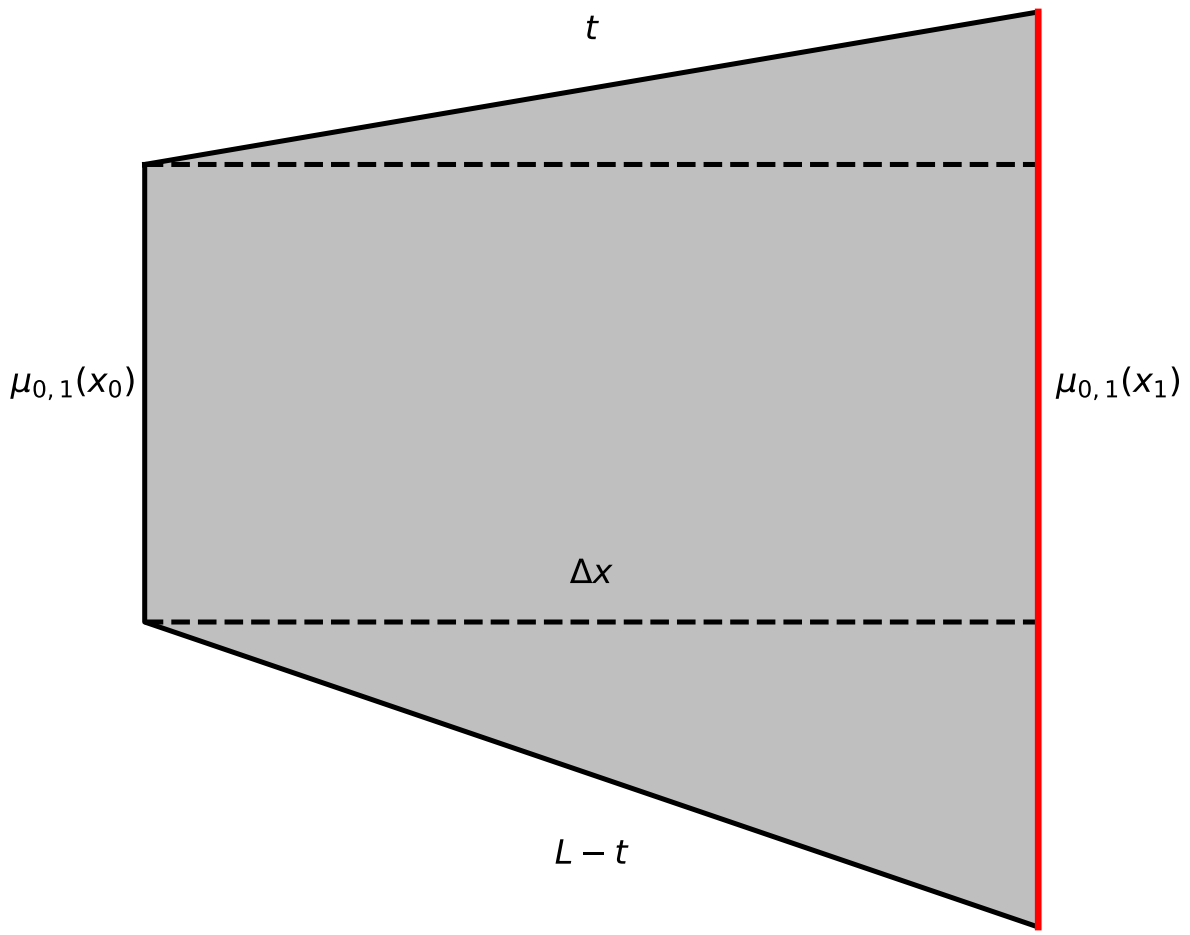


Figure 3.5: Example of a segment of an object X made from piecewise linear contour functions f and g.

solutions at the top and the bottom. Scaling of Δx and L equally corresponds to a scaling of the curve. Changing $\mu_{01}(x_0)$ corresponds to a vertical translation of the curve in the shape space. In conclusion, not all $\mu_{\varepsilon,\varepsilon'}(X, Y^r)$ have a corresponding shape since the curve is bounded, and most of the cases that do, do not have a unique shape except in the case where the object segment is horizontally symmetric in which case the two solutions give the same shape.

We have now handled the possible equivalent shapes of a single linear segment of an object. Consider again a single linear segment on the interval $[x_0; x_1]$. We can equally define this linear segment as two linear segments on $[x_0; x']$ and $[x'; x_1]$ where $x_0 < x' < x_1$. Since these two linear segments represent f and g exactly as before the measures $\mu_{\varepsilon,\varepsilon'}$ are unchanged under this substitution. With one linear segment, we had up to two possible equivalent solutions given the measures. With two linear segments, we have all combinations given a total of up to 4 possible solutions. But since we can now recursively split linear segments into two segments, we enter the domain of *infinitely jagged* fractal functions such as the Weierstrass function without changing the measures of the object. However, in most real-world scenarios, objects often consist of approximately smooth parts that do not present this jaggedness.

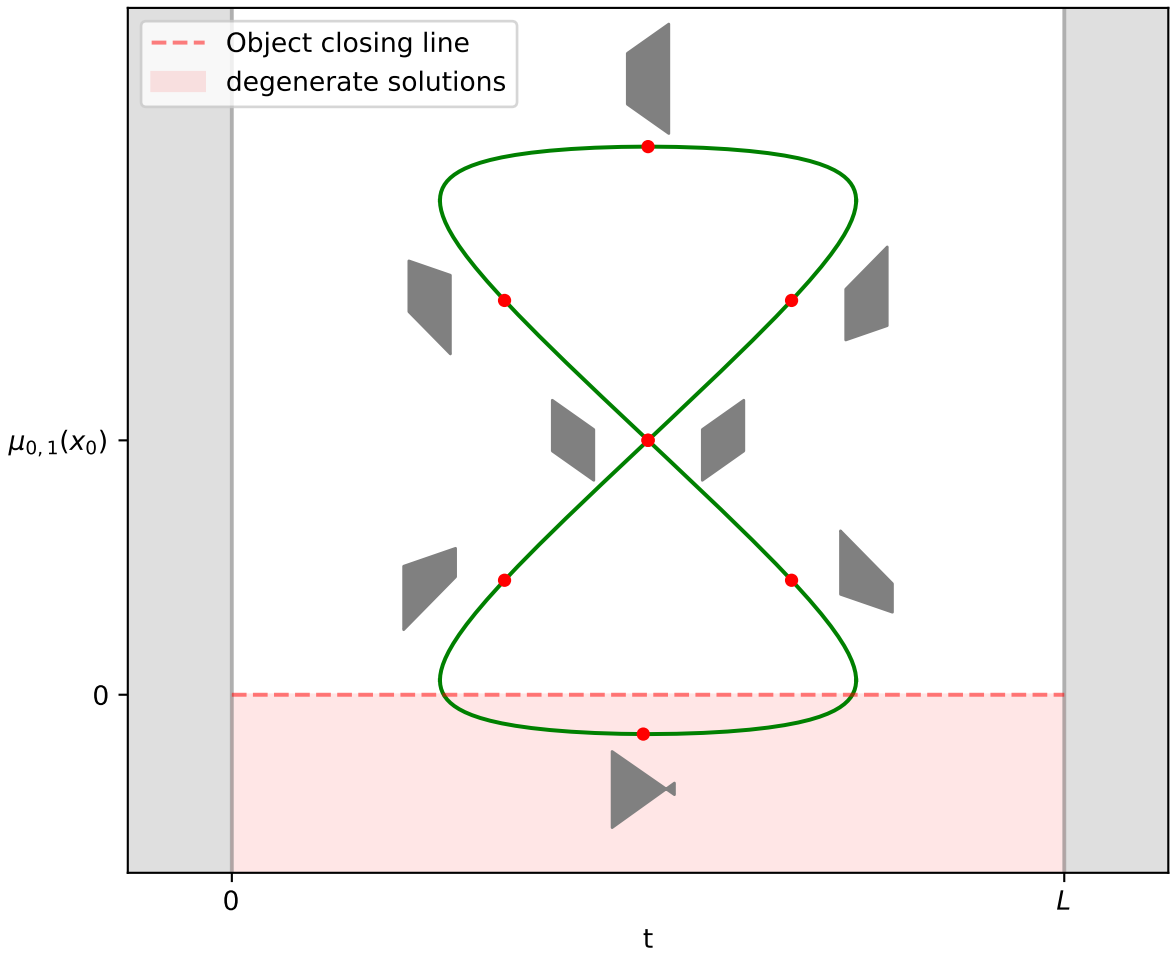


Figure 3.6: Shape space and solution curve of (3.43) for fixed measures with $L > 2\Delta x$. The red dotted line denotes the solution closing the object. Below this line, only degenerate objects exist.

Assume now that f, g are C^1 smooth functions (see Fig. 3.7). In this case, f and g can no longer recursively be broken into smaller segments and reflected since that would break the smoothness constraint in most cases. The exceptions are at points x where $f'(x) = -g'(x)$. Consider a shape with a finite set of such points, and consider the intervals between neighboring points.

Lemma 1. *The measures* (3.39)–(3.42) *are invariant under vertical translation of both* f *and* g.

Proof. Let d denote the vertical translation distance of f and g such that the translated functions are $\hat{f}(x) = f(x)+d$ and $\hat{g}(x) = g(x)+d$. Since $\hat{f}(x)-\hat{g}(x) = f(x)+d-g(x)-d = f(x)-g(x)$, the claim follows for both (3.39) and (3.40). Since the derivative is unaffected by constant terms, $\hat{f}'(x) = f'(x)$, so it also holds for (3.41). For (3.42) it is trivially true. □ □

Note that objects mirrored about $x = 0$ are equivalent. That is, if $f(x), g(x)$ denote functions on $[a; b]$ defining an object, then the functions $\hat{f}(x) = f(-x)$ and $\hat{g}(x) = g(-x)$ on $[-b; -a]$ define an object with equivalent measures. Further, if there exists a point x' such that $f'(x') = -g'(x')$, then the curves may be inflected at this point, i.e., we may

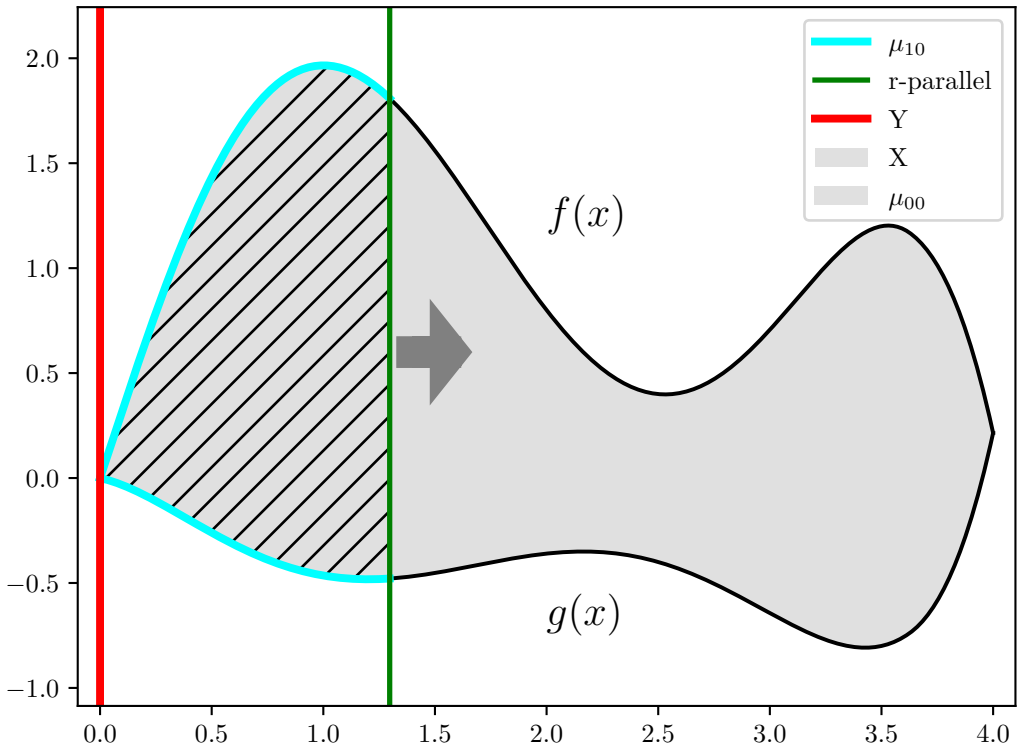


Figure 3.7: Example of an object from two smooth contour functions.

generate new functions

$$\hat{f}(x) = \begin{cases} f(x), & \text{if } x < x' \\ f(x') + g(x') - g(x), & \text{else} \end{cases} \tag{3.44}$$

$$\hat{g}(x) = \begin{cases} g(x), & \text{if } x < x' \\ g(x') + f(x') - f(x), & \text{else} \end{cases} \tag{3.45}$$

These function will still be in C_1. This gives rise to the notion of inflection intervals:

Definition 1. *We define an* inflection interval *to be an interval* $[x_0; x_1]$ *such that for* $x \in [x_0; x_1]$, $f'(x) = -g'(x)$ *or* $f(x) = g(x)$ *if and only if* $x = x_0$ *or* $x = x_1$.

Definition 2. *We define a* symmetric interval *to be an interval* $[x_0; x_1]$ *such that for all* $x \in [x_0; x_1]$, $f'(x) = -g'(x)$.

Note that every object in our constrained example can be separated into intervals overlapping only at the boundary, which are either inflection intervals or symmetric intervals.

Definition 3. *Given a inflection interval* $[x_0, x_1]$*, an inflected shape is:*

$$\hat{f}(x) = \begin{cases} f(x), & \text{if } x < x_0 \\ f(x_0) + g(x_0) - g(x), & \text{if } x_0 \leq x < x_1 \\ f(x_0) + g(x_0) - (g(x_1) + f(x_1)) + f(x), & \text{else} \end{cases} \tag{3.46}$$

$$\hat{g}(x) = \begin{cases} g(x), & \text{if } x < x' \\ g(x_0) + f(x_0) - f(x), & \text{if } x_0 \leq x < x_1 \\ g(x_0) + f(x_0) - (f(x_1) + g(x_1)) + g(x), & \text{else} \end{cases} \tag{3.47}$$

These function are in C_1*.*

Now, we can state the following theorem.

Theorem 4. *The measures* (3.39)–(3.42) *are invariant under inflections of inflection intervals* $[x_0, x_1]$*.*

Proof. For $x < x_0$ the functions are unchanged. For the inflection interval, consider the horizontal line $y = (f(x_0)+g(x_0))/2$, and let $\hat{f}(x) = 2y - g(x)$ and $\hat{g}(x) = 2y - f(x)$ denote the functions after reflection across y. Since $\hat{f}(x) - \hat{g}(x) = 2y - g(x) - (2y - f(x)) = f(x) - g(x)$, the theorem holds for (3.39) and (3.40). We also have $\hat{f}'(x)^2 = (-g'(x))^2 = g'(x)^2$ and likewise $\hat{g}'(x)^2 = (-f'(x))^2 = f'(x)^2$ and thus the theorem holds for (3.41). For (3.42) the theorem is trivially true. For $x \geq x_1$ the functions are unchanged expect for an added constant and by Lemma 1 we conclude that the measures under inflections. □ □

Theorem 5. *Let* n *denote the number of inflection intervals of an object. Ignoring vertical translations and reflections across* Y*, the equivalence class of the object has size* 2^n *and all equivalent objects can be generated by reflecting a subset of the inflection intervals.*

Proof. We know we can generate new objects by reflecting each inflection interval. This gives at least 2^n objects by all combinations of reflections of the n inflection intervals. Assume now there is another object in the equivalence class which is not among the 2^n objects. Let $\hat{f}, \hat{g}$ denote the upper and lower function of this object respectively. Since all measures must be the same for this object, by (3.40) this means $\hat{f}(x) - \hat{g}(x) = f(x) - g(x)$. As all functions are assumed to be C^1, we get

$$\hat{f}'(x) - \hat{g}'(x) = f'(x) - g'(x). \tag{3.48}$$

We then look at the derivative of (3.41). Using the fundamental theorem of calculus, we have

$$\sqrt{1 + \hat{f}'(x)^2} + \sqrt{1 + \hat{g}'(x)^2} = \sqrt{1 + f'(x)^2} + \sqrt{1 + g'(x)^2} \tag{3.49}$$

Together, (3.48) and (3.49) implies either

$$\hat{f}'(x) = f'(x) \tag{3.50}$$
$$\hat{g}'(x) = g'(x) \tag{3.51}$$

or

$$\hat{f}'(x) = -g'(x) \tag{3.52}$$
$$\hat{g}'(x) = -f'(x) \tag{3.53}$$

Thus we have that $\hat{f}, \hat{g}$ is either f, g directly or its horizontal reflection which are both among the 2^n objects already in the equivalence class, contradicting the assumption that this is a new object. □ □

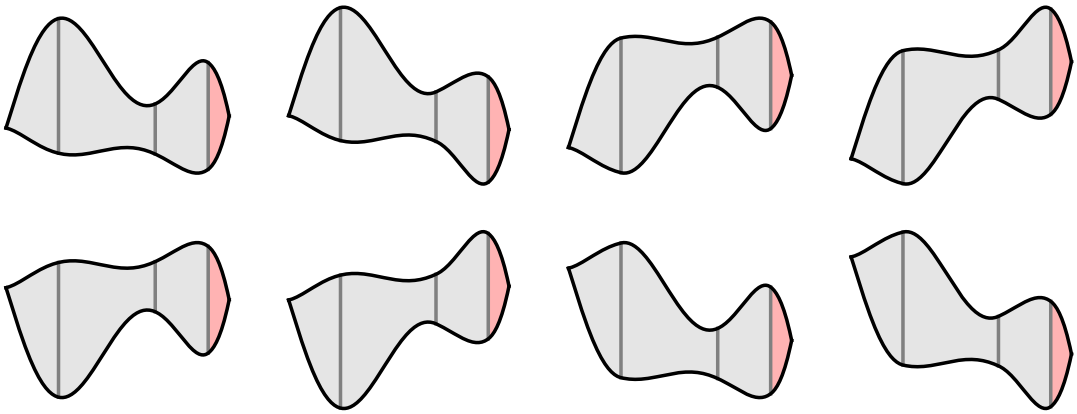


Figure 3.8: Example of an equivalence class of an object. Here the equivalence class has $n = 3$ inflection intervals and one symmetric interval shown in red.

The complete equivalence class for object consisting of C^1 functions f and g are therefore all vertical translations of 2^n objects symmetrically on both sides of Y. If we ignore translations and reflections in the horizontal and vertical axis, we have 2^{n-1} unique objects equivalent under the measures because half of them are the reflections of another shape across the horizontal. We show an example of the 2^n equivalent shapes in Fig. 3.8.

3.5 Implementation

For implementation, we have experimented with the following 3 approaches.

1. Grid-based: The simplest approach is to work exclusively on a regular grid. We represent Y^r only through a distance map D on the window W with $D(w) = \min_{y \in Y}\{|w - y|\}$ for $w \in W$. Our sample points are given on a regular grid. The distance map D can be approximated by methods such as the Fast Marching Method. We then represent X as a $\{0, 1\}$ mask on the regular grid. Counting the overlapping voxels of the mask consisting of the voxels w for which $D(w) < r$ and the mask of X gives an estimate of μ_{00}. Using morphological operations, subtracting X from the dilation of X approximates the boundary of X. This can similarly be done for Y^r. Note here that we are limited in accuracy by the pixel representation, and that the error increases significantly when working with the boundary measures μ_{01}, μ_{10} and μ_{11}, as voxel masks do not, in general, represent boundaries accurately.

2. Parameterized surfaces: When Y^r can be parameterized easily, we can represent X and Y as contours and meshes for $d = 2$ and $d = 3$, respectively. Libraries such as the python library *Shapely* can calculate set intersections, set unions, and set differences on contours as well as area and curve length. Similarly, libraries such as *PyMesh* support the same operations but for meshes.

3. Mesh-based: Our favorite algorithm is based on triangulated surface-meshes of the objects X and Y^r which we have implemented for $d = 3$:

   (a) Given surface meshes of X and Y, we produce a tessellation of the interior of X, Y, and Y^r, whose vertices we will denote V_X etc. Moreover, $V_{\partial X} \subset V_X$ denotes the surface vertices.

(b) For each r-parallel set Y^r, we identify interior, intersecting, and exterior simplices in V_X. For the intersecting simplices, we estimate the intersecting surface by linear interpolation.

(c) For μ_{10} and μ_{01}, we sum the surface of intersections, for μ_{00}, we sum the volume of the interior simplices and the relevant part of the intersecting simplices. For μ_{11} we calculate the length of the intersecting line.

For shortest vertex to mesh distance, we used *PyMesh*, and we used the Fast Marching method implementation in the `scikit-fmm` python package with linear interpolation implemented in Scipy. The integration of the measures over the meshes was implemented in C++ compiled with SWIG for use in Python.

For a run-time analysis of triangulation and tetrahedralization of label images, we refer the reader to the corresponding articles on TetGen [47] or TetWild [21]. Calculating the distance function as basis for the r-parallels can be accomplished by solving the Eikonal equation using a method such as the Fast Marching method. The Fast Marching method has time complexity $\mathcal{O}(n \log n)$ [46], where n is the number of grid points in the rectilinear grid used to discretize the domain. For a faster result on a rectilinear grid, the level-set re-initialization method is often faster trading accuracy for speed. Alternatively, the distances can be calculated directly on the meshes by mesh to point algorithms, which have time complexity $\mathcal{O}(V \log(F))$ where V is the number of vertices in the observed object mesh and F is the number of faces in the reference object mesh.

The integration of the meshes is done by iterating over r. For each value of r, we consider the intersection between the boundary of Y^r and the X-simplex mesh. Each simplex of X can either be completely inside, partially inside or completely outside of Y^r. Simplicies that are partially inside Y^r are subdivided into smaller simplices that are completely inside or outside Y^r. Determining the overlap and performing subdivision can be done in constant time. Thus, for N different values of r and M simplicies, the total time complexity is $\mathcal{O}(NM)$.

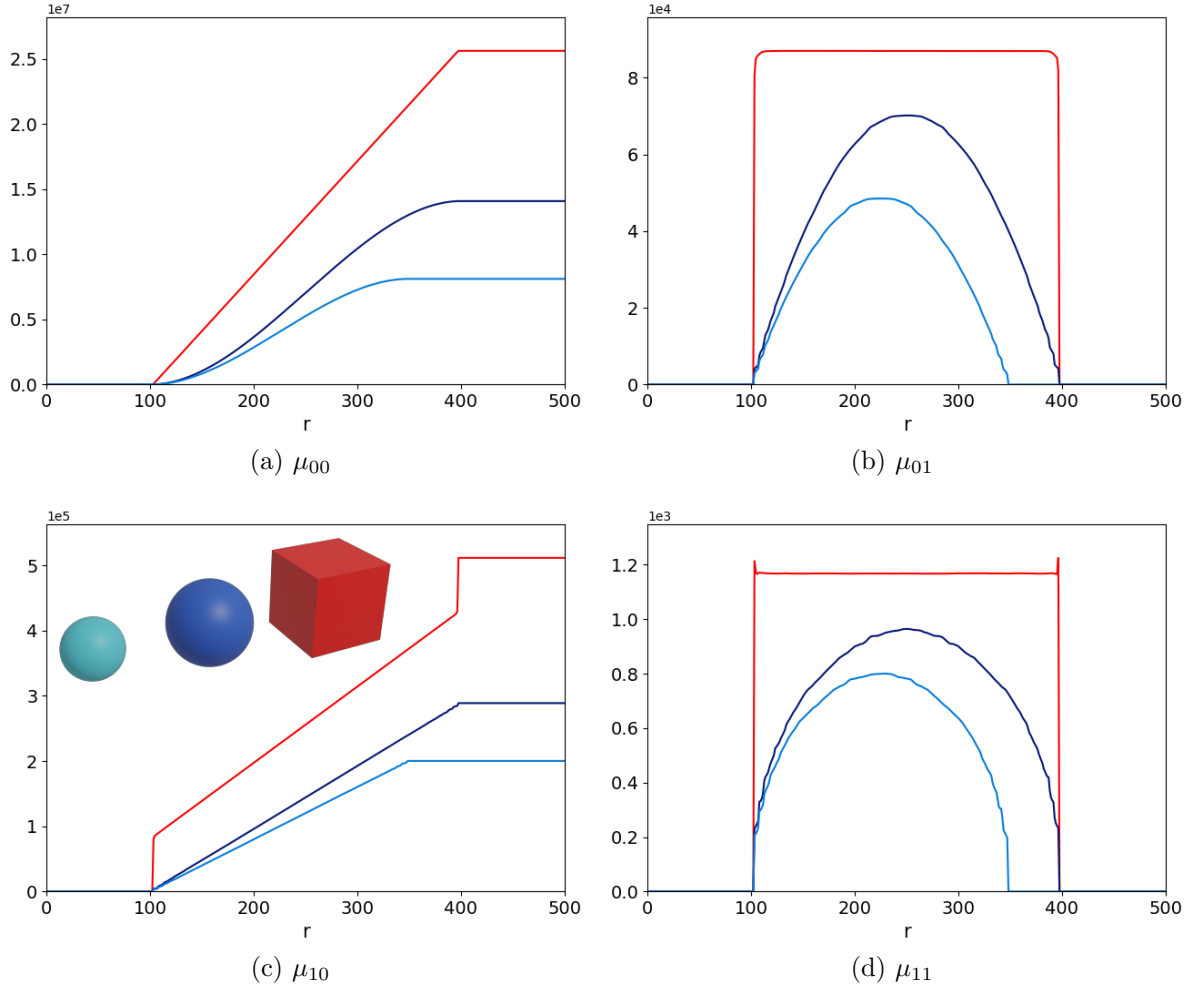
We then experiment on randomly generated vertices and faces, running on Lenovo ThinkPad T470, Intel Core i7-7500U 2.70GHz CPU. Shown in Table 3.3, we see an acceptable run-time for most applications. In this implementation there is still room for optimizations since most of the current brute force loops are both trivially parallelizable and have a significant overlap in calculations.

3.6 Experiments on synthetic objects

In the following, we will give examples of experiments conducted on synthetic data.

As a first experiment and in the spirit of the analytical examples in Section 3.3.2, consider $\mathbb{R}^3$ with an infinite plane as the reference object, 2 spheres and a cube as observed objects near the plane and with a cubic observation window aligned with the reference object. Top and bottom row in Fig. 3.9 show the experimental evaluation of $\mu_{\varepsilon,\varepsilon'}$. The experiments show that μ_{00} is a monotonically increasing function of the integral of the volume of the observed object from 0 to r with μ_{01} as its derivative. The surface measure μ_{10} for the spheres is a linearly increasing function, while the cube has two discontinuous steps caused by the alignment of the cube with the observation plane. The curve measure μ_{11} is quadratic for the spheres and constant for the cube.

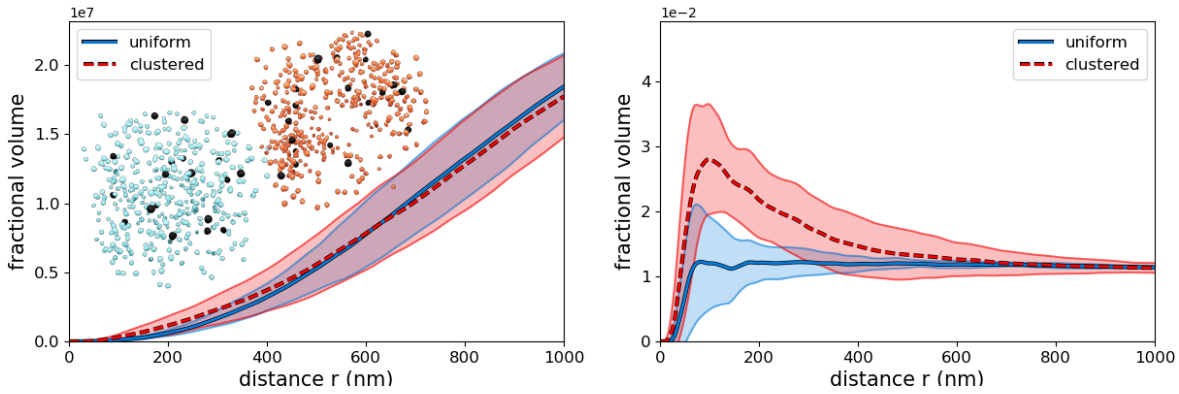
As a second synthetic experiment, we consider sets of spheres $\mathcal{X}$ and $\mathcal{Y}$ randomly distributed in a window, as shown in Fig. 3.10. Looking at Fig. 3.10a, we see two $\mu_{00}(X, Y^r)$



(a) μ_{00} (b) μ_{01}

(c) μ_{10} (d) μ_{11}

Figure 3.9: Example of measures.The observed objects X is either of 2 spheres of different sizes and a cube, as shown in (c) with colors corresponding to the curves. The reference object Y is an infinite plane at a minimum distance of 100 units from the relevant object. The observation window is a cube of side-length 500 with a side coinciding with Y and otherwise centered around the observed object.



(a) $\mu_{00}(X, Y^r)$ volume measure. (b) $\nu_{00}(X, Y^r)$ normed volume.

Figure 3.10: Example of the volume measures on uniformly distributed and clustered spheres. A slight difference can be seen directly in the $\mu_{00}(X, Y^r)$ graph, but the clustering is clearly visible in $\nu_{00}(X, Y^r)$.

Table 3.3: Run-time results for the mesh integrator at $d = 3$.

μ_{00}, μ_{01}			
mesh tetrahedra	discretization steps	mean ($n = 200$)	variance ($n = 200$)
1k	1k	54.27ms	1.86µs
10k	1k	426.05ms	571.68µs
100k	1k	3.12s	2.47ms
1M	1k	28.76s	163.89ms
1k	10k	386.73ms	111.04µs
10k	10k	4.31s	1.06ms
100k	10k	18.10s	60.41ms
1M	10k	358.50s	175.11s
μ_{10}, μ_{11}			
mesh faces	discretization steps	mean ($n = 200$)	variance ($n = 200$)
1k	1k	6.85ms	0.34µs
10k	1k	73.56ms	4.02µs
100k	1k	711.67ms	1.08ms
1M	1k	7.97s	17.03ms
1k	10k	9.85ms	0.69µs
10k	10k	338.33ms	12.39µs
100k	10k	5.07s	21.17ms
1M	10k	57.80s	1.30s

volume graphs showing a slight difference between the two groups. For the normalized measure $\nu_{00}(X, Y^r)$, shown in Fig. 3.10b, the effect of the normalization factor is striking. Now we clearly see a distinction between the two groups: For both curves, $\nu_{00}(X, Y^r)$ is close to zero for small values of r. For the blue spheres, $\nu_{00}(X, Y^r)$ (in blue) converges quickly to a horizontal line, while the for the red spheres, $\nu_{00}(X, Y^r)$ (in red) peaks at $r \sim 175$ and converges to the blue horizontal line for $r \gtrsim 500$. From these curves we thus conclude that for both experiment the spheres do not overlap with the Y shapes, that the blue sphere are randomly distributed, while the red spheres cluster near the Y shapes with a characteristic distance of about $r \sim 175$, but appear randomly distributed for distances $r \gtrsim 500$.

3.7 Experiments with edge correction

To assess the effectiveness of the edge corrections (3.19), (3.20) and (3.24), we generate synthetic germ-grain realizations by placing a number of non-overlapping observed objects of 3 variants and a reference object in form of a triangle. We consider two windows W and W_{large}. We calculate the edge corrected $K^e_{\varepsilon,\varepsilon'}$-estimator on W and compare with the ground truth on non-edge corrected $K^e_{\varepsilon,\varepsilon'}$-estimator on W_{large}. An example region can be seen in Figure 3.11.

For the edge correction of (3.19), we disregard all measurements where y did not fall into the smaller window $W \ominus B_M(0)$. For the translative edge correction, each pixel area element is weighted by the area ratio (3.20) before summation to a final measure. Similarly for the isotropic edge correction, we weight each pixel area by the circle length ratio in (3.24). We discretized the circle boundary by 512 points for simple interior arc-length calculation.

We average the measures over 1000 simulated realisations of $\mathcal{X}, \mathcal{Y}$. Shown in Fig-

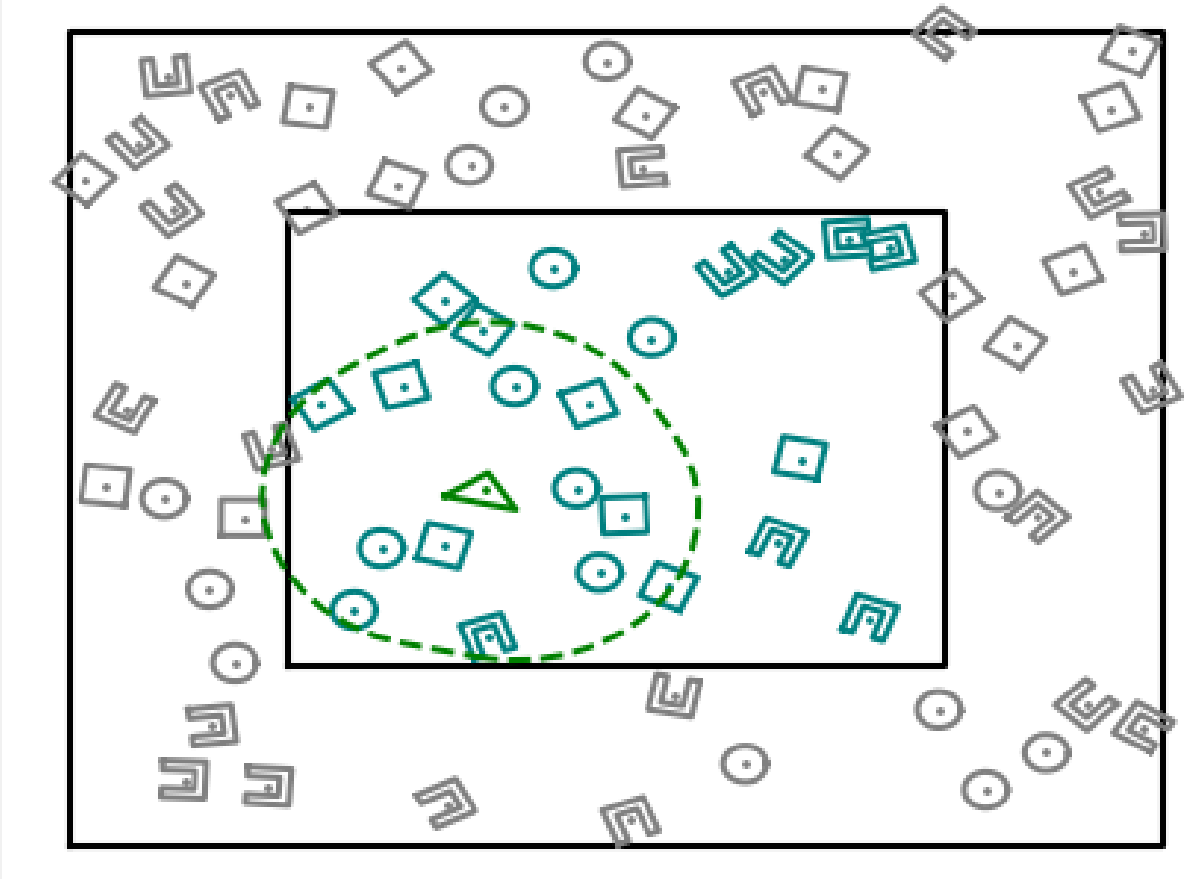


Figure 3.11: Example of the synthetic germ-grain realizations used for testing edge correction. We show the observed objects $\mathcal{X}$ as gray and teal sets with Poisson placed germs x and three different grain types X selected uniformly at random for each realization. We show Y and Y^r in green and W and W_{large} as black contours.

ure 3.12, we see the uncorrected reading underestimates the measure as expected while the corrected version gives estimates better consistent with the expected measure.

3.8 Experiments on cellular ultrastructures

Communication by neurons in humans is mainly achieved by a combination of electric potential changes and chemical signaling. The vesicles serve as transient containers of the chemicals released towards another neuron at a connection point, the synapse, upon voltage potential changes in the neuron. Synaptic vesicles are therefore almost exclusively observed directly next to synapses. The mechanisms of replenishment of the vesicles are thought in part to be done by two main routes. The first is by only partly release of the neurotransmitters, the vesicle membrane is thus preserved [25, 15]. The second is by a slow endosomatic route, where an endosome is formed from the membrane, pinched off into vesicles and filled with neurotransmitters. An open question is if the cells keep a reservoir of vesicles at some distance to the synapse with some evidence [37]. To assess the above, we examine a publicly available FIB-SEM dataset of the CA1 hippocampus region of a healthy adult rodent. Original dimensions before registration were $2048 \times 1536 \times 1065$ with a voxel size of $5 \times 5 \times 5$ nm. We have segmented the complete volumetric image into

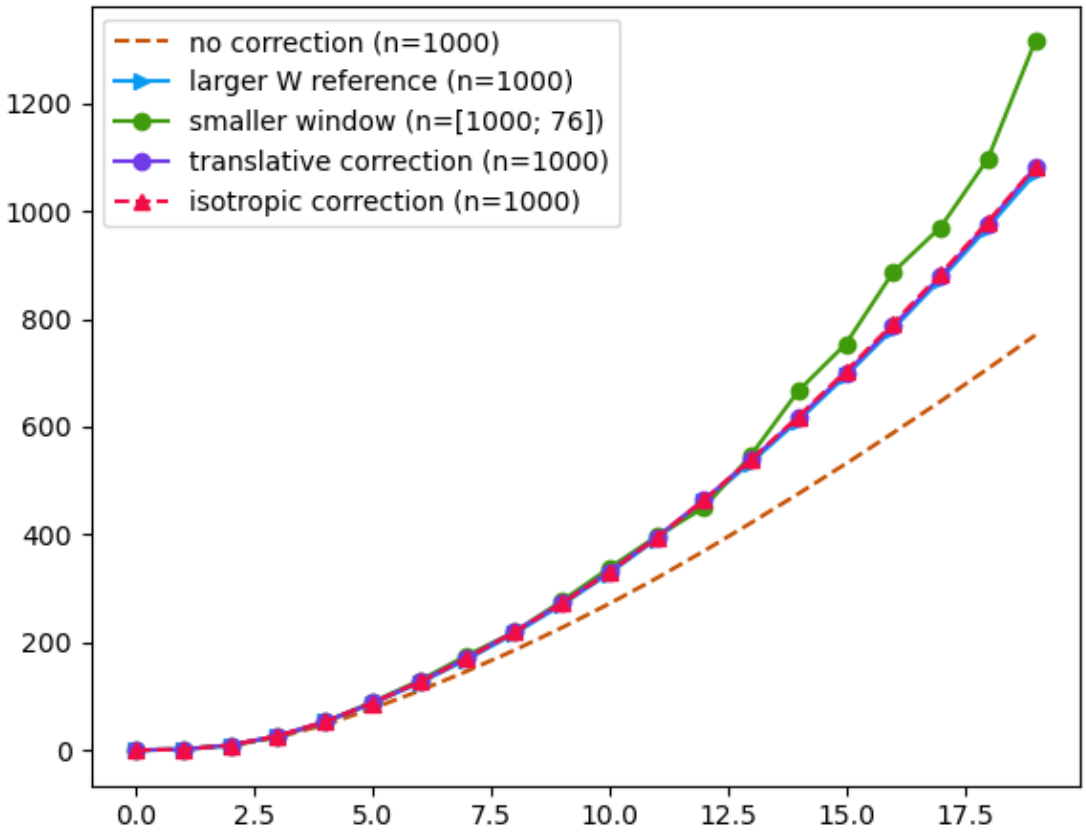


Figure 3.12: The two-dimensional area measure using different edge correction strategies. We show μ_{00} without correction alongside translative correction, isotropic correction, and correction by shrinking the set of y used in the measured. For comparison, we also display the measure using a sufficiently large window to remove edge effects. The result is an average of 1000 simulations.

cell wall, synapses, mitochondria, vesicles, endoplasmatic reticulum, and the segmentation is available at [51].

The FIB-SEM were segmented by a neural network U-Net model [41], and cleaned up using an Avizo Amira pipeline [43]. The volume was registered to correct for drift using a model based approach described in [50]. From the masks, we generate mesh reconstructions using a Marching Cubes Lewiner implementation in the SciPy Python package [60]. The meshing is further refined with the PyMesh Python package, and the TetWild C++ library to do mesh simplification and tetrahedralization [21]. Examples of the resulting segmentation is shown in Fig. 3.1

In Fig. 3.2, we show L_{01} for mitochondria and vesicles when using the synapse as the reference object. No correction for cell walls have been performed. From Fig. 3.2a we see that the mitochondria is absent close to the synapse, with a gradually increasing presence until around 800 nm, where the expected presence of mitochondria goes towards the global area fraction of mitochondria in the whole sample. From Fig. 3.2b we see a presence of vesicle at the very close range $\approx$ 25 nm followed by a proportionally greater

measure of peaking around $25 - 100$ nm. Before the measure tends to the global area fraction, we see a slight increase in vesicles near the 500 nm range which could indicate a presence of a vesicle reservoir at that range for these neuronal processes.

3.9 Conclusion

We have presented a novel method that extends the theoretical foundations of K-function summary-statistics in the field of spatial point statistics to geometric objects, adds elegant reasoning about the shape of one object with respect to a reference object, and includes some existing shape-relation measures. We show that the method can be used to display spatial relations such as spreading or clustering compared to uniformly random distribution, and that the method is sensitive to properties such as cross-sectional area and thickness. A core strength is that the method is built on the n-dimensional Hausdorff measure enabling us to intuitively understand the shape relations. Statistical tests have been developed for the comparison of K-functions in the context of spatial point-processes [18] and are readily available for the comparison between groups.

Conflict of interest

The authors declare that they have no conflict of interest.

3.10 Post Publication Notes

Implementing Mesh Integration in a Distance Map

This section seeks to cover the important details which were not included in Section 3.5 of the article. We first show why we avoided the simple morphological approach to estimating the measures. We then go into the implementation details on the integration of meshes in a distance map.

Morphological edge-measures may over- and under-estimate surface area

Although widely used, measuring contour length by pixel counting carries with it a significant degree of error. For an overview, see [54]. We here show an example of how it gives a bias in the sphericity measure

$$\psi = \frac{\pi^{1/3}(6 \cdot \text{Volume})^{2/3}}{\text{Area}} \ . \tag{3.54}$$

Note that for a ball, this measure is maximized with the value

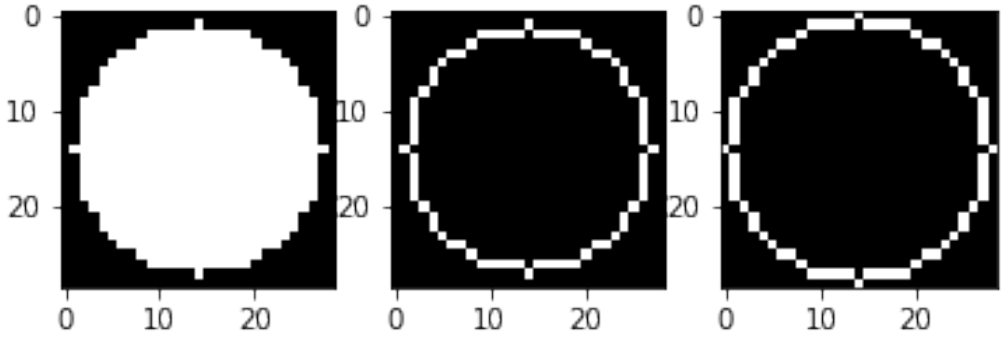
$$\frac{\pi^{1/3}(6\frac{4}{3}\pi r^3)^{2/3}}{4\pi r^2} = 1 \ . \tag{3.55}$$

For all other physically possible combinations of volume and surface area, ψ will be between 0 and 1. We here generate pixelated spherical balls B_r with radii r equal to $1, 3, 5, \ldots, 501$ respectively. We estimate their volume as the total number of pixels belonging to the ball, and the surface area in two ways, 1) by the total number of pixels belonging to $B_r/(B_r \ominus B_1)$ (erosion edge detection), and 2) by the total number of pixels belonging to $(B_r \oplus B_1)/B_r$ (dilation edge detection). A section of a ball and its surface pixels can be seen in Figure 3.13a. Using the estimated measures of volume and surface area, we get two graphs of sphericity measures as shown in Figure 3.13b which shows a systematic overestimation both by the erosive and dilative methods. Therefore, the measure also exceeds the upper limit of 1 of the measure. From Figure 3.14, we can see the volume is estimated correctly using voxels but the surface area is significantly underestimated. The meshing of a ball is a higher-order representation of the object which thus naturally will give a better estimate. The sphericity measure will be lower than 1 approaching 1 for increasing mesh refinement.

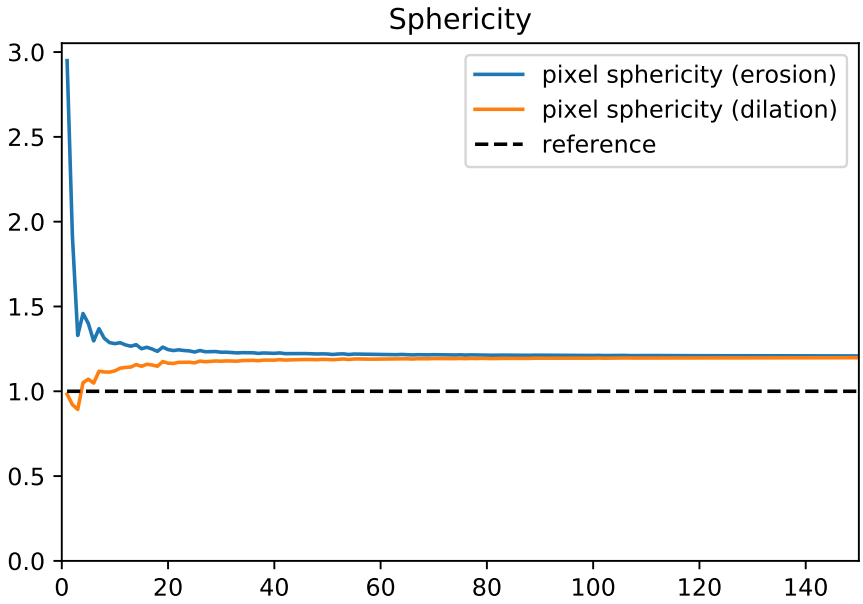
Implementation details for mesh-based measures

The code is written in Python with the central components implemented in C++ bound to Python using SWIG. The measures μ_{00} and μ_{01} require a full tetrahedral mesh while μ_{10} and μ_{11} require only a surface triangulated mesh. This called for two implementations differing only in that one works on triangles while the other works on tetrahedra. We will focus mainly on the implementation for tetrahedral meshes. The implementation for triangular meshes is similar but significantly simpler. For both methods, the calculations are broken into two main phases.

1. An initial one-time phase, which remaps each tetrahedron into a fitting coordinate system making calculations easier.



(a) Voxel ball and its morphological edges



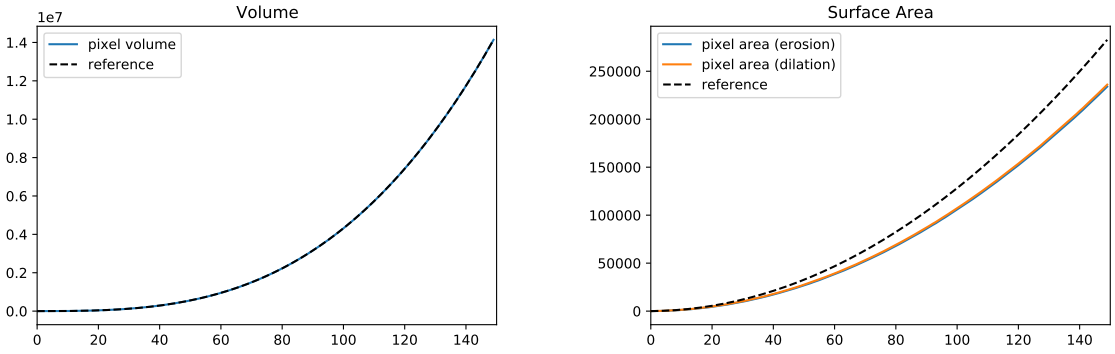
(b) Sphericity measure

Figure 3.13: Sample voxel ball and estimated edge using morphological operations and corresponding graphs of the sphericity measure.

2. The integration phase, where, for each r-value, the tetrahedra are evaluated to be either inside, outside, or on the edge and integrated with a suitable method.

The initial phase

The initial phase assumes a given mesh as vertex coordinates and corresponding tetrahedra as well as an array of distances from the reference object to each vertex. We first estimate the distance function locally inside the tetrahedron as an affine function. For the case of tetrahedrons, let $\mathbf{r} = \{r_i\}$ denote the distance from the reference object to each vertex, $\mathbf{v}_i$. For simplicity, we sort the vertices in increasing order of r_i. The parameters a, b, c, d



(a) Voxel Volume

(b) Voxel Surface Area

Figure 3.14: The two constituent fundamental measures for voxel measure estimates in comparison to analytical true values. The reference volume is calculated by $4/3\pi r^3$, and the surface area is calculated by $4\pi r^2$.

of the linear function $ax + by + cz + d = 0$ is solved for through the linear system

$$\begin{bmatrix} \mathbf{v}_1^T & 1 \\ \mathbf{v}_2^T & 1 \\ \mathbf{v}_3^T & 1 \\ \mathbf{v}_4^T & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} \quad . \tag{3.56}$$

From this, the gradient of the linear function is $\mathbf{g} = [a, b, c]^T$ and the gradient magnitude is $|\mathbf{g}| = \sqrt{a^2 + b^2 + c^2}$. Using this, we first translate each vertex such that $\mathbf{v}^1$ is at the origin. We then rotate the tetrahedron around the origin such that the gradient vector aligns with the x-axis and scale the tetrahedron in x by the inverse of the gradient magnitude $1/|\mathbf{g}|$. Finally, we translate the tetrahedron in x by r_1. I.e. the tetrahedron is transformed such that the distance value, r, corresponds exactly to the x-coordinate (see Figure 3.15). This is very helpful in the integration phase as, aside for a few edge cases, we can integrate the tetrahedron as a function of x. The scaling parameters and vertex coordinates are stored such that we can recover the measures in the integration phase. Similarly, the total volume of each tetrahedron is pre-computed in this phase because it covers one of the two common cases in the integration phase.

The integration phase

Let T_i denote a set representing the i'th tetrahedron in the object mesh of X with Y^r denoting the reference object. The integration phase numerically calculates the following integrals

$$\tilde{\mu}_{ij} = \sum_i^N \tilde{\mu}_{ij}(T_i, Y^r) \quad , \tag{3.57}$$

where the $\tilde{\cdot}$ indicates the measure is a numerical approximation of the measures μ_{ij}. In other words, we can estimate the measure by a summation of local estimates on each tetrahedron. When the r-parallel set does not intersect a tetrahedron the measures are easily calculated as they are either not counted or counted fully, eg. by 0 volume or the

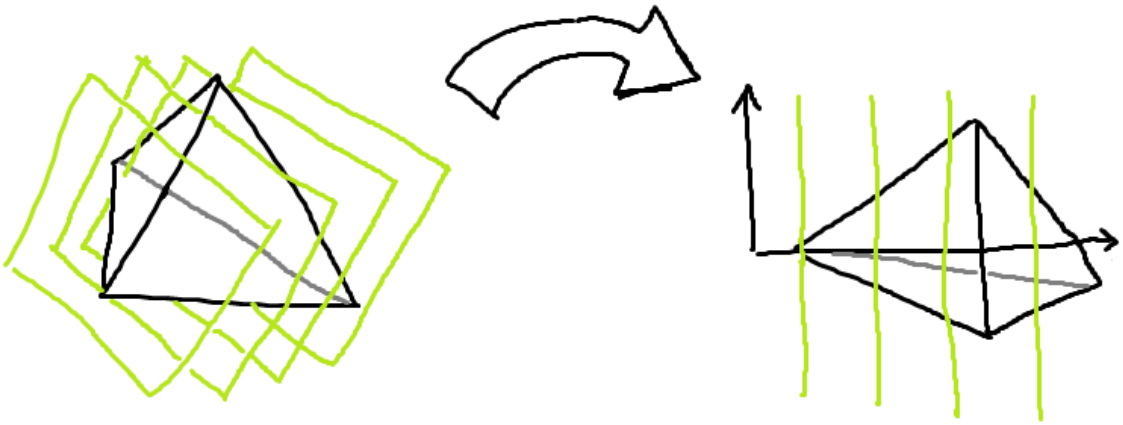


Figure 3.15: Each tetrahedron is aligned such that an increase in x corresponds exactly to an increase in the r-parallel distance function.

full tetrahedron volume of which a known formula exists, as an example. Let $r_{\min}, r_{\max}$ denote the minimum and maximum distance value of the vertices in the tetrahedron. For $r \leq r_{\min}$, the measures are

$$\tilde{\mu}_{00}(T_i) = 0 \tag{3.58}$$
$$\tilde{\mu}_{01}(T_i) = 0 \tag{3.59}$$
$$\tilde{\mu}_{10}(T_i) = 0 \tag{3.60}$$
$$\tilde{\mu}_{11}(T_i) = 0 \tag{3.61}$$

and for $r \geq r_{\max}$, the measures are

$$\tilde{\mu}_{00}(T_i) = \text{volume}(T_i) \tag{3.62}$$
$$\tilde{\mu}_{01}(T_i) = 0 \tag{3.63}$$
$$\tilde{\mu}_{10}(T_i) = \text{area}(T_i) \tag{3.64}$$
$$\tilde{\mu}_{11}(T_i) = 0 \tag{3.65}$$

If the r-parallel intersects the tetrahedron T_i, we subdivide the tetrahedron into smaller tetrahedra. To achieve this, we first define intermediate vertex coordinates between the 4 original vertices. Since we have translated, scaled, and rotated the tetrahedra, we can easily define the vertices as vector functions of $r = x$. That is, f_{ij} is a linear function between vertices v_i and v_j with $f_{ij}(r_i) = v_i$ and $f_{ij}(r_j) = v_j$ (see Figure 3.16b). In the following, we denote tetrahedra here by $T(p_1, p_2, p_3, p_4)$ meaning $T_i = T(v_1, v_2, v_3, v_4)$ and triangles similarly by $T(p_1, p_2, p_3)$. How the tetrahedron is divided depends on how the value, r, of the r-parallel relates to the vertex distances r_1, r_2, r_3, r_4. If $r_1 < r \leq r_2$, we have

$$\tilde{\mu}_{00}(T_i) = \text{volume}(T(v_0, f_{01}(r), f_{02}(r), f_{03}(r)) \ , \tag{3.66}$$
$$\tilde{\mu}_{01}(T_i) = \text{area}(T(f_{12}(r), f_{13}(r), f_{14}(r))) \ , \tag{3.67}$$

and similarly, if $r_3 \leq r < r_4$, we have

$$\tilde{\mu}_{00}(T_i) = \text{volume}(T_i) - \text{volume}(T(v_4, f_{14}(r), f_{24}(r), f_{34}(r)) \ , \tag{3.68}$$
$$\tilde{\mu}_{01}(T_i) = \text{area}(T(f_{12}(r), f_{13}(r), f_{14}(r))) \ . \tag{3.69}$$

These two sub-intervals are the simplest. In the interval $r_2 < r < r_3$ the intersection is not a triangle but a quadrilateral (see Figure 3.16a). The intersection area for the measure, μ_{01}, is thus the total area of the two triangles $T(f_{13}(r), f_{14}(r), f_{23}(r))$ and $T(f_{14}(r), f_{23}(r), f_{24}(r))$. Upon careful inspection, one can also realize that volume inside the tetrahedron up to the r-parallel can be covered by three distinct, non-overlapping tetrahedra in several ways. One is by the tetrahedra

$$T(v_1, f_{13}(r), f_{14}(r), f_{23}(r)) \ ,$$
$$T(v_1, f_{14}(r), f_{23}(r), f_{24}(r)) \ , \text{and}$$
$$T(v_1, v_2, f_{23}(r), f_{24}(r)) \ .$$

The measures are thus calculated by

$$\tilde{\mu}_{00}(T_i) = \text{volume}(T(v_1, f_{13}(r), f_{14}(r), f_{23}(r)))+ \tag{3.70}$$
$$= \text{volume}(T(v_1, f_{14}(r), f_{23}(r), f_{24}(r)))+ \tag{3.71}$$
$$= \text{volume}(T(v_1, v_2, f_{23}(r), f_{24}(r))) \tag{3.72}$$
$$\tilde{\mu}_{01}(T_i) = \text{area}(T(f_{13}(r), f_{14}(r), f_{23}(r)))+ \tag{3.73}$$
$$= \text{area}(T(f_{14}(r), f_{23}(r), f_{24}(r))) \tag{3.74}$$

A note on parallelism

Both the initial phase as well as the integration phase is highly parallelizable since each tetrahedron in the initial phase is independent of each other, and can thus be calculated simultaneously. Similarly, each queried r-value in the integration phase does not depend on previous calculations, meaning these can be calculated in parallel as well. We do this using open-mp.

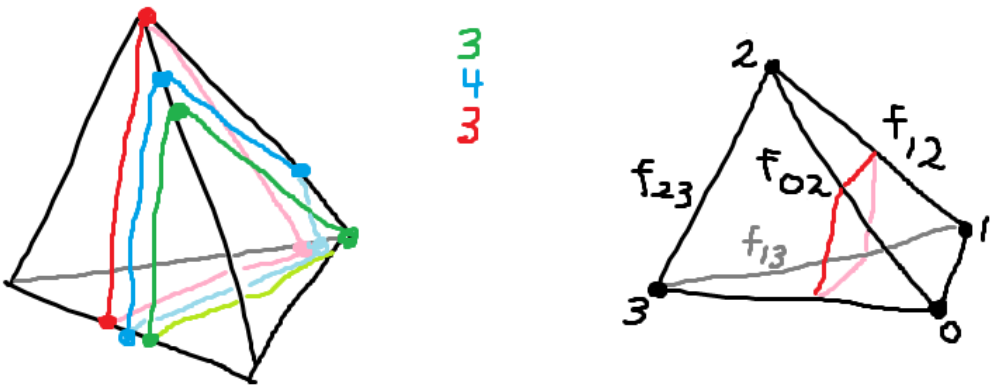
Bounded Shape Measure

In many applications, directly using the euclidean distance does not does not fully model the distribution of objects the due to the presence of bounding objects, obstructions, or propagation speed constraints in the domain. Examples are the movement of objects inside a tube, such as blood cells in capillaries, mitochondria inside the axon terminal, or the propagation of molecules through tissue with semi-permeable membranes.

This idea has been partially explored previously in [52], but it can also both be understood in spatial statistics and stochastic geometry as either a point-process or marked point-process together with a special distance metric.

The metric arises from the solution to the Eikonal Equation with a variable velocity. With the velocity in the Eikonal equation equal to a constant 1, the solution is exactly the shortest distance from the initial zero contours to any point in the domain. If the velocity is variable, the solution is better understood as the time it takes a wave to propagate from the initial zero contours to any point in the domain.

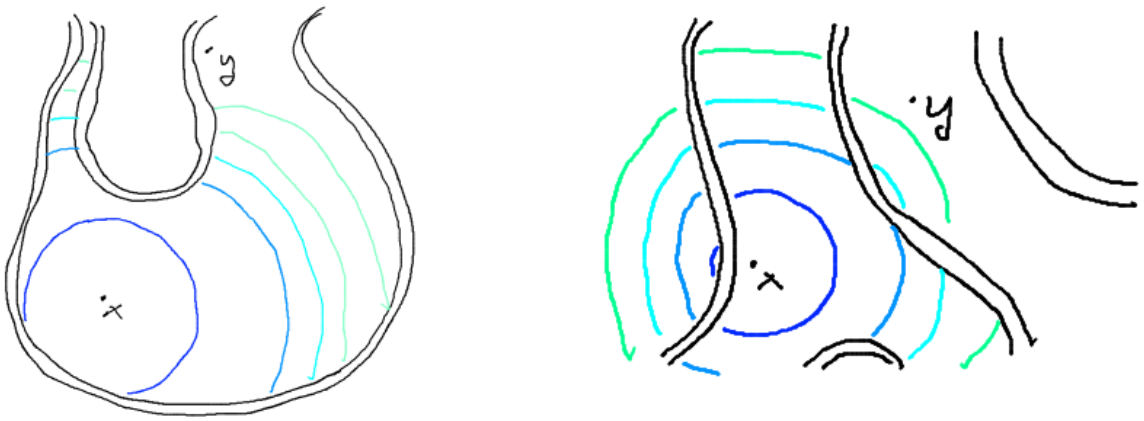
In the simple case where the velocity is the same everywhere, it is easy to see that the shortest distance is a metric. We have $d(x, x) = 0$ because the shortest distance to



(a) Vertices in cuts

(b) Vertex functions

Figure 3.16: Illustrations of where r-parallel intersecting vertices are placed for a tetrahedron given an arbitrary distance function angle.



(a) Bounded wave propagation

(b) Wave limited partially by semi-permeable membrane

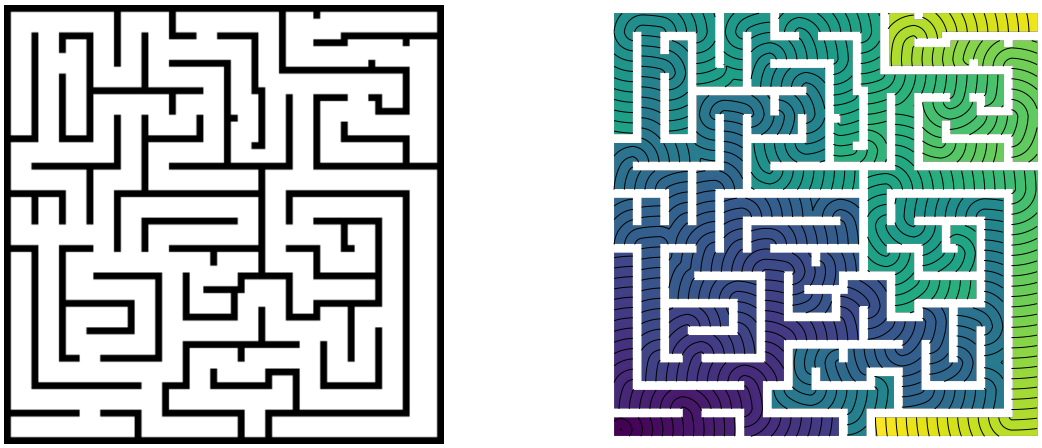
Figure 3.17: Examples of a distance defined by a propagation wave.

itself is 0. The shortest distance between two points is the same regardless of direction, meaning $d(x, y) = d(y, x)$. Finally, the triangular inequality holds because if we assume $d(x, y) > d(x, z) + d(z, y)$ that would contradict that $d(x, y)$ is the shortest distance between x and y since we found a shorter through z.

The distance metric $d(x, y)$ can be calculated by methods such as the Fast Marching Method or similar.

A case where a bounded distance map is significantly different is in a case such as a labyrinth shown in Figure 3.17. The distance from the bottom left to the top right is not just significantly higher in this case, but the gradient directions of the distance map are highly determined by the location of the boundaries.

At synapses in brain tissue, we have two distinct compartments, the pre-and post-synaptic compartments which each contain different objects but are causally linked by their functional interdependence. Measuring the objects inside both compartments separately, and defining one of the measurement distance directions to be negative we can put



(a) A highly limiting connected domain (b) A distance map from a bottom left source point

Figure 3.18: An example where the shortest path metric is well suited and makes a substantial difference.

both measurements together in a combined function. This gives a natural directionality to the measurement from the pre-synaptic compartment to the post-synaptic with the active zone as the origin. This makes for a logical way to compare groups of curves as the pairing is natural, they belong to the same synapse, and if it exists, could show a grouping in the combined measurement functions.

Chapter 4

A Spherical Statistic for Sparse Point Patterns

4.1 Introduction

In this chapter, we will study the estimation of summary statistics of points from sparsely sampled sections of a 3-dimensional volume. We will develop the theory to address the sparseness of the point pattern, firstly, by a minimum of assumptions to bridge the gap between the sections directly in the derivation of the summary statistics, and secondly, by viewing the points as a density to then estimate the density between the sections by use of a machine learning approach as well as a registration approach. We are motivated by the following case.

We are interested in understanding how human glial cells raised in vitro migrate and expand following their injection into the living tissue of animals. Given their potential as therapeutic vectors for glial replacement in a multitude of neurological disorders, understanding their migration and expansion patterns becomes especially important. In that context, understanding how these cells migrate and expand following injection will help optimize the delivery strategy and maximize the effect.

In a cell injection migration study, cells are injected into the living tissue of a group of young animals. The animals are sacrificed at specific time intervals from the time of injection. The tissue is then sectioned, prepared, and imaged followed by cell point annotation in specified regions in each section. If the cells were to move randomly from the injection site, the process could be modeled as Brownian motion from the injection site. This means the cells would spread like a normal distribution from the injection site with a standard deviation directly correlated to the cell speed and migration time [58], and the average distance traveled is proportional to the square root of the speed and time traveled [13]. However, the cells often do not migrate randomly. Some cells have been found to follow specific tracts in the tissue [56] and we can hypothesize that cells may be drawn towards areas of need by way of some signaling process.

The K-function from spatial statistics is a simple and powerful summary statistic [39]. It describes the expected number of points as a function of the distance from any point. The K-function is estimated by cumulatively count how many points fall within an increasing distance r from each of the points and then add and normalize the resulting functions. However, this requires that every section is annotated lest the count will be incorrect. Cell counting is time-consuming thence often performed only on a limited number of sections at regular intervals. This gives rise to the following theoretical description

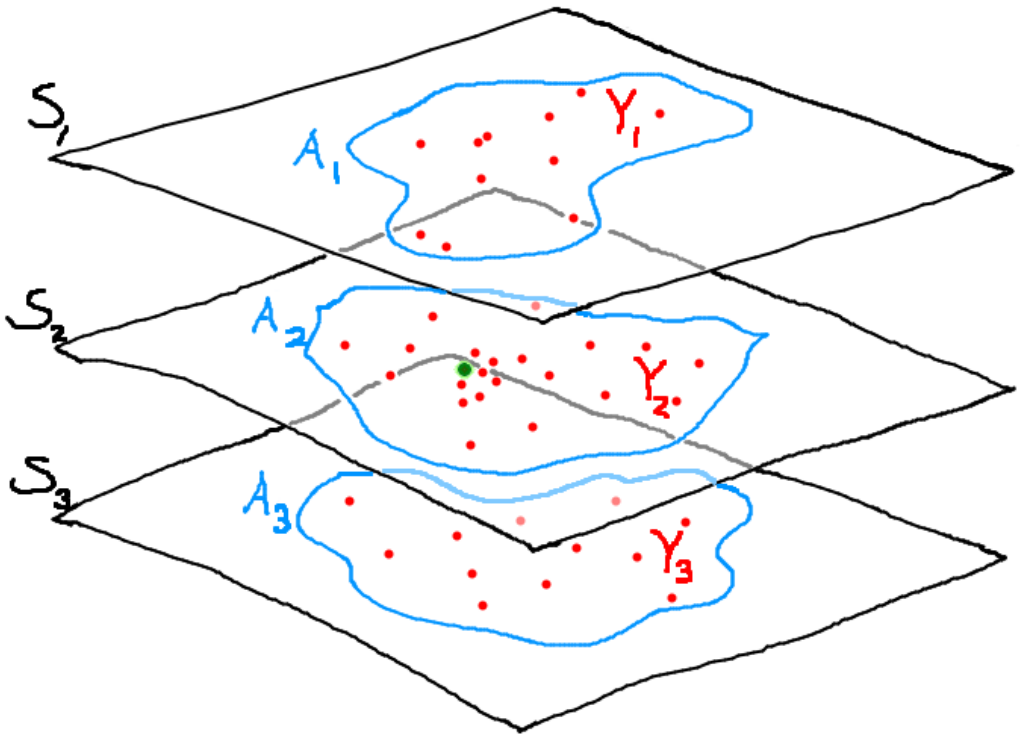


Figure 4.1: Example of the sparse sections and regions of a cell injection migration study. S_i denotes sections where point annotations are known, A_i is the region in each section where the cells are counted. Y_i is the points and the injection site is denoted by a green dot.

of our knowledge not yet explored.

4.2 Method

Let each observed point y_j be restricted to a domain A_i such that $y_j \in Y_i \subseteq A_i \subseteq S_i$, where Y_i is a finite observed realization of a sub-domain A_i restricted to a section S_i, where S_i is a plane in $\mathbb{R}^3$ (see Figure 4.1). Each section has in the practical case a thickness but only the planar position of the points is known. Importantly, the thickness of the sections is smaller than the distance between the sections. Without loss of generality, we assume the injection site is at the origin, meaning, that for each point y_j, the distance to the injection site is $|y_j|$.

We assume first that we have full knowledge of the point pattern in $\mathbb{R}^3$. Let $f(x)$ denote the probability density function describing the probability of finding a cell at $x \in \mathbb{R}^3$. We can calculate the probability of finding a cell within distance r of the injection site by

$$P(r) = \int_{B_r} f(x)\, dV \ , \tag{4.1}$$

where B_r is a ball of radius r centered at the injection site. Further, letting M denote the total number of injected cells, we define $\Lambda(r)$ to be a cumulative point prediction function

as

$$\Lambda(r) = MP(r) \ . \tag{4.2}$$

We can interpret the function, $\Lambda(r)$, as the expected number of points within distance r of the injection site. This is therefore directly related to the intensity function λ for the spatial point process by

$$\Lambda(r) = \int_{B_r} \lambda(x)\, dx \ . \tag{4.3}$$

To account for the missing data between the sections, we add one key assumption. We assume that λ is uniform for any subset $H \subseteq \partial B_r$. I.e. we can define the function

$$\mu(r) = \mathbb{E}\left[\frac{1}{|H|}\int_H \lambda(x)\, d\alpha\right] \ , \tag{4.4}$$

where $d\alpha$ is a small area element. Assuming that $\Lambda(r)$ is differentiable, we can get an alternative definition of $\mu(r)$ by dividing the expected number of points within distance r by the area of the shell of B_r. We get

$$\mu(r) = \frac{\frac{d}{dr}\Lambda(r)}{\frac{d}{dr}V(r)} = \frac{1}{4\pi r^2}\frac{d}{dr}\Lambda(r) \ . \tag{4.5}$$

where $V(r) = |B_r|$. Due to the missing information between the sections, we don't know $\Lambda(r)$. However, if we were to estimate $\mu(r)$, then we can also get an estimate of $\Lambda(r)$. To estimate $\mu(r)$, we calculate approximate subset versions of $\Lambda(r)$ and $V(r)$ based on the data we have within the annotated section. Let $A = \bigcup_i A_i$ denote the sparse domain for which we know the location of the points. We estimate domain-limited versions of $\Lambda(r)$ and $V(r)$ by

$$\hat{\Lambda}_A(r) = \sum_{i=1}^{\hat{M}} 1_{|y_i|<r} \ , \tag{4.6}$$

$$\hat{V}_A(r) = \sum_i w|B_r \cap A| \ , \tag{4.7}$$

where $\hat{M}$ is the number of observed points, and 1_P equal to 1 if the predicate P is true and 0 otherwise, and w is the thickness of the sections S_i. w is here assumed to constant in the sectioning direction. For small w this is approximately true. From (4.5), we get

$$\Lambda(r) = \int_0^r \mu(x)\frac{d}{dx}V(x)dx \ . \tag{4.8}$$

Calculating μ in a dense space as in (4.5) is not possible due to the limited information, but due to the assumption of uniform point density on ∂B_r, we can get a sparse estimate restricted to A. We have

$$\hat{\mu}_A(r) = \frac{\frac{d}{dr}\hat{\Lambda}_A(r)}{\frac{d}{dr}\hat{V}_A(r)} \ . \tag{4.9}$$

Since $\hat{\mu}_A$ is an estimator of μ, we replace and get

$$\hat{\Lambda}(r) = \int_0^r \hat{\mu}(x)\frac{d}{dx}V(x)dx = 4\pi \int_0^r x^2\hat{\mu}(x)dx \ . \tag{4.10}$$

In turn we can estimate the total number of injected cells by

$$\hat{M} = \hat{\Lambda}(\infty) \ . \tag{4.11}$$

In (4.6), $\hat{\Lambda}$ is estimated from discrete points and thus it and $\hat{\mu}_A$ can not be differentiated. $\hat{V}$ has similar issues because the boundary might be formed by discrete points, and with regions ending, it is at most piecewise differentiable. The simplest approach is to calculate discretely on half-open intervals $I_i = [r_i; r_{i+1})$. For general function f, we replace $\frac{d}{dr}f(r)$ in (4.9) and (4.10) by the finite difference

$$\Delta f(r_i) = \frac{f(r_i) - f(r_{i-1})}{r_i - r_{i-1}} \ , \tag{4.12}$$

to get an interval approximation of $\hat{\Lambda}$ given by

$$\hat{\Lambda}(r_i) \approx \sum_{i=1}^{n} \left(\frac{\Delta\hat{\Lambda}_A(r_i)}{\Delta\hat{V}_A(r_i)} \Delta V(r_i) \left[r_i - r_{i-1}\right] \right) \tag{4.13}$$

$$= \sum_{i=1}^{n} \left(\frac{\hat{\Lambda}_A(r_i) - \hat{\Lambda}_A(r_{i-1})}{\hat{V}_A(r_i) - \hat{V}_A(r_{i-1})} \left(V(r_i) - V(r_{i-1})\right) \right) \tag{4.14}$$

$$= \frac{4}{3}\pi \sum_{i=1}^{n} \left(\frac{\hat{\Lambda}_A(r_i) - \hat{\Lambda}_A(r_{i-1})}{\hat{V}_A(r_i) - \hat{V}_A(r_{i-1})} \left(r_i^3 - r_{i-1}^3\right) \right) \ . \tag{4.15}$$

An alternative approach is to approximate $\hat{\Lambda}$ and $\hat{V}$ by differentiable functions.

4.2.1 A higher order approach

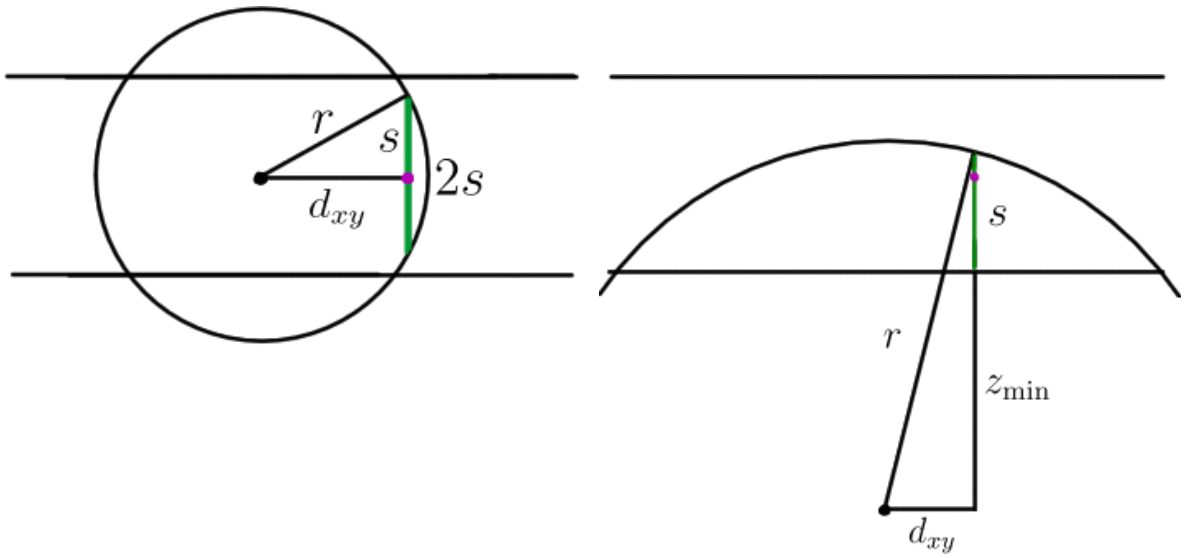
Since we know the thickness of the sections, we can get a better estimate of both $\hat{\Lambda}_A(r)$ and $\hat{V}_A(r)$, we will here show how.

To get a better estimate of $\hat{\Lambda}_A$, we model the likelihood of the unknown z-coordinate within the section by assuming the point is equally likely to be at any depth in the section. Then, instead of counting the point discretely, we integrate the uniform point likelihood over the thickness corresponding the distance from the injection point r. This gives us two cases. Either the point is in the section which has the injection site, or it is in one of the other sections (See Figure 4.2).

The the original estimate of $\hat{\Lambda}$ in (4.6) is a discrete step function. The steps corresponds to the step of each $P(r)$ for the points, each with a single step at $|y_i|$, the distance between the point and the injection side. Using the thickness of the sections, we instead estimate $P(r)$ as a gradual scaling across the assumed uniform distribution across the section it belongs.

In the case where the point is inside the section, the integration of the point is spread along a line of length $2s$, the side lengths s of two equal right angle triangles. We divide this length by the width of the section to get the uniform density as a function of $s \in [0; w/2]$, where w is the width of the section. Let $\alpha \in [0; 1]$ denote the gradual integration of some point as a function of r. We get α as

$$\alpha = \frac{2s}{w} \ . \tag{4.16}$$



(a) Site inside section (b) Site outside section

Figure 4.2: Measuring points as a scaling of a linear point density function across a section. The injection site is here shown as a black dot and the point we are integrating is purple. We show the integrated amount as the green line.

We get the side length s by the triangle side length

$$s = \sqrt{r^2 - d_{xy}^2} \ , \tag{4.17}$$

where d_{xy} is x, y planar distance. In the case where the point is outside the section, we first find $z_{\min}$, the distance to the closest section edge and calculate the scaling factor

$$\alpha = \frac{s}{w} \ . \tag{4.18}$$

This time we have to account for the distance to the section, and we only get one triangle side length. We thus calculate s by

$$s = \sqrt{r^2 - d_{xy}^2} - z_{\min} \ . \tag{4.19}$$

To get a measure of the volume $\hat{V}_A(r)$, we have 4 cases which give a non-zero volume (see Figure 4.3). If the injection site is inside the section we have two cases. If $r \leq w/2$, the volume is simply a ball volume

$$V_{\text{ball}} = \frac{4}{3}\pi r^3 \ . \tag{4.20}$$

If $r > w/2$, we have some excess volume in the form of two spherical cap volumes. The volume of a spherical cap is known [36]. Given a spherical cap with height h of a ball with radius r, the volume is

$$V_{\text{cap}} = \frac{\pi h^2}{3}(3r - h) \ . \tag{4.21}$$

We calculate h as

$$h = r - \frac{w}{2} \ . \tag{4.22}$$

The volume is then

$$V_{\text{section}}(r) = V_{\text{ball}} - 2V_{\text{cap}} \tag{4.23}$$
$$= \frac{4}{3}\pi r^3 - \frac{2\pi h^2}{3}(3r - h) \tag{4.24}$$
$$= \frac{4}{3}\pi r^3 - \frac{2}{3}\pi\left(r - \frac{w}{2}\right)^2\left(2r + \frac{w}{2}\right) \quad . \tag{4.25}$$

In case the injection site is outside the section we have 3 cases. 1) if $r \leq z_{\min}$, the volume is zero. 2) if $z_{\min} < r \leq z_{\max}$, where $z_{\max}$ is the distance to the furthest section edge, the volume is a spherical cap with height h, given by

$$h = r - z_{\min} \quad . \tag{4.26}$$

In the final case 3), when $r > z_{\max}$, the volume inside the section is the subtraction of the volume of one spherical caps which extends beyond the section with height

$$h_{\text{full}} = r - z_{\min} \quad , \tag{4.27}$$

with the volume of a smaller excess spherical cap which has height

$$h_{\text{excess}} = r - z_{\max} \quad . \tag{4.28}$$

The volume when the injection site is outside and $r > z_{\max}$ therefore becomes

$$V_{\text{outside}} = \frac{\pi z_{\max}^2}{3}(3r - z_{\max}) - \frac{\pi z_{\min}^2}{3}(3r - z_{\min}) \quad . \tag{4.29}$$

Experiments

To show the above method works, we experiment on a uniform and a Gaussian distribution of generated point. We generate a point set of 50000 points either following a uniform or a Gaussian distribution. We then define a number of sections and a section width w such that w is smaller than the distance between the sections. We keep the original set of points for ground truth comparison and make a filtered point set of all the points within distance $w/2$ of the section z-coordinate (see Figure 4.4b). We estimate $\hat{\Lambda}_A(r)$ and $\hat{V}_A(r)$ by the higher order approach, integrating over the section thickness. We then estimate $\hat{\Lambda}(r)$ using the discrete estimate (4.15). Both for the uniform and and Gaussian point pattern, we see the observed values are expectely low compared to the reference measurement while the estimated statistic is much closer. We also see the discrete integration of $\Delta\hat{\Lambda}$ to get $\hat{\Lambda}$ result in an measurable error shown in Figure 4.4d. This is most likely because each value in $\Delta\hat{\Lambda}$ incurs a small error which accumulated over the integration.

4.3 Case 1: known injection site, normally distributed points

To understand the cell point pattern we could ask if the cells spread uniformly since injections which could be interpreted as fast random movement within the domain? do the cells repel each other such as is seen for astrocytes? Are the cells normal distributed

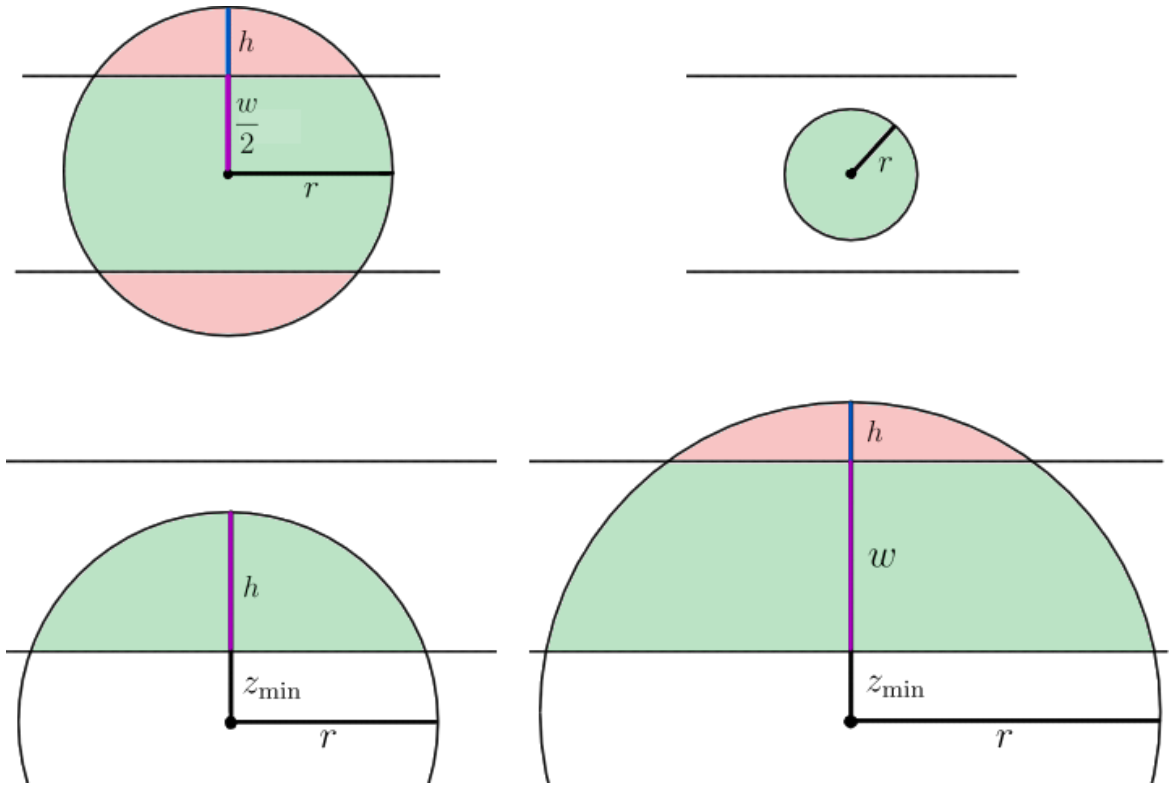


Figure 4.3: The four cases for a volume integration over thick sections. The sections are shown from a side view in a 2D projection. The green parts here shows the volume of which we are interested. The red parts shows the excess volume which we subtract.

around the injection site indicating random slow movement or do the cells cluster around certain biological features?

With the estimation of the $\Lambda(r)$ function, we can compare to the distributions for different point patterns. Let's consider the case where points are normally distributed around the injection site. For each point coordinate (X, Y, Z) having distribution functions $P_X = \mathcal{N}(0, \sigma), P_Y = \mathcal{N}(0, \sigma), P_Z = \mathcal{N}(0, \sigma)$. Λ can be thought of as a stochastic variable N given by

$$N = \sqrt{(X^2 + Y^2 + Z^2)} \tag{4.30}$$

with distribution

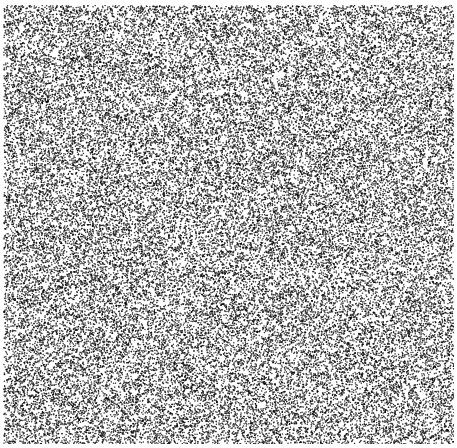
$$f_N = \int_{B_r} P_X P_Y P_Z \, dV = \int_{B_r} \frac{1}{\sqrt{2\pi\sigma}^3} e^{-(x^2+y^2+z^2)/(2\sigma)} \, dV \tag{4.31}$$

$$= \frac{1}{\sqrt{2\pi\sigma}^3} \int_0^{\pi} \int_0^{2\pi} \int_0^{r} n^2 e^{-n^2/(2\sigma)} \sin(\varphi) \, dn \, d\theta \, d\varphi \tag{4.32}$$

$$= \frac{4\pi}{\sqrt{2\pi\sigma}^3} \int_0^{r} u^2 e^{-u^2/(2\sigma)} \, du \tag{4.33}$$

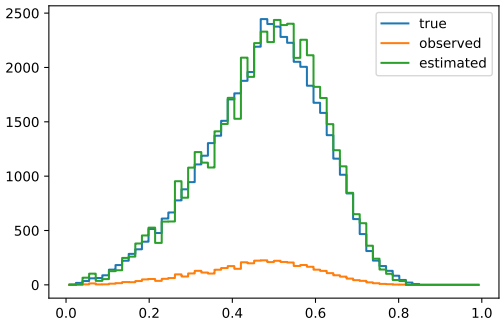
In two dimensions, the distribution function has a nice closed form solution, namely the Rayleigh distribution. However, in 3D we are left with a non-elementary error function. The distribution function f_N is given by

$$f_U(\sigma) = \frac{4\pi}{\sqrt{2\pi\sigma}^3} \left(\sqrt{\pi/2} \, \sigma^{3/2} \mathrm{erf}\left(\frac{r}{\sqrt{2}\sqrt{\sigma}} \right) - r\sigma e^{-r^2/(2\sigma)} \right) \tag{4.34}$$

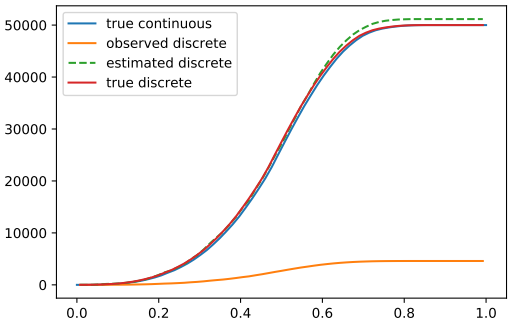


(a) Uniform point pattern

(b) Observed Part



(c) $\Delta\hat{\Lambda}(r)$



(d) $\hat{\Lambda}(r)$

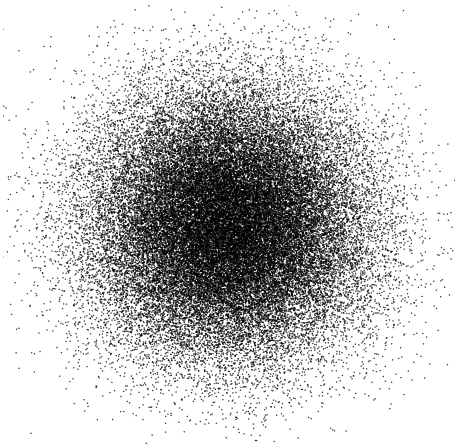
Figure 4.4: An example of estimating the point function on synthetically generated Poisson point process.

To assess if and how the point pattern varies from this, we can do numerical fitting of this distribution to $\Lambda(r)$ and compare the two curves. Assessing this difference can tell us how closely the point pattern matches a normal distribution which would indicate random movement from the injection site.

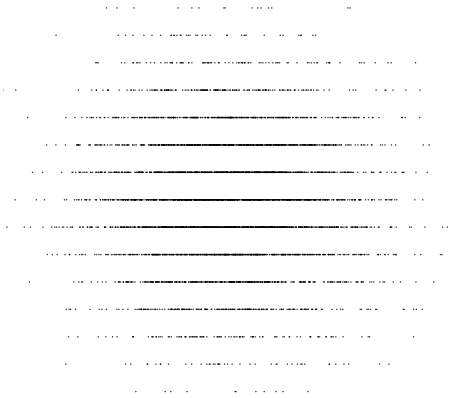
4.4 Case 2: Unknown injection site, normally distributed points

The location of the injection site might not be known or annotated imprecisely. In the latter case, assuming a normal distribution of points, the result is that $\Lambda(r)$ no longer follows a Rayleigh distribution, but instead displays as a Rician distribution. In both cases, estimating the injection point is useful.

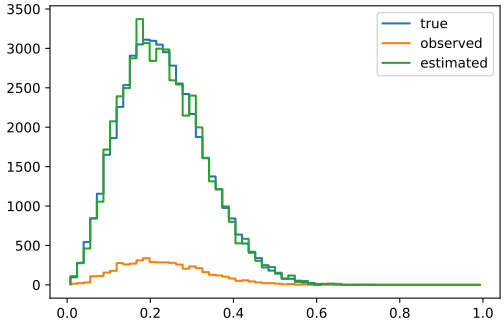
To estimate the injection site we assess planar cuts of the intensity λ of the underlying point function. Firstly, we look at the intensity function marginally in the plane. That is, given $\lambda(x, y, z_i)$, where x, y denote the planar coordinates within each section and z_i



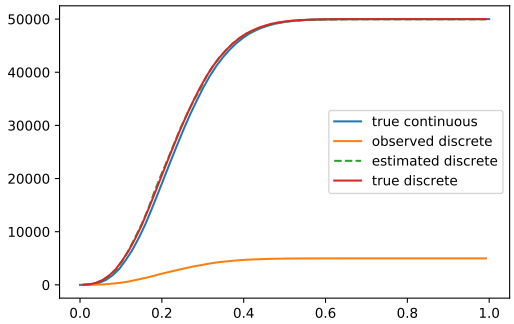
(a) Uniform point pattern



(b) Observed Part



(c) $\Delta\hat{\Lambda}(r)$



(d) $\hat{\Lambda}(r)$

Figure 4.5: An example of estimating the point function on synthetically generated Gaussian point process.

its discrete z-coordinates, we calculate the marginal planar intensity

$$G_{\text{planar}}(x,y) = \sum_i^N \lambda(x,y,z_i) \tag{4.35}$$

$$= \sum_i^N ce^{\frac{-(x-\overline{x})^2-(y-\overline{y})^2-(z_i-\overline{z})^2}{2\sigma^2}} \tag{4.36}$$

$$= c\left(\sum_i^N e^{\frac{-(z_i-\overline{z})^2}{2\sigma^2}}\right) e^{\frac{-(x-\overline{x})^2-(y-\overline{y})^2}{2\sigma^2}} \tag{4.37}$$

$$= \alpha e^{\frac{-(x-\overline{x})^2-(y-\overline{y})^2}{2\sigma^2}} \quad . \tag{4.38}$$

From the above we see the planar intensity is a scaled normal distribution in 2D with mean value preserved. This means that a good estimate for the injection site in the planar coordinates is to use the mean point position as this is the maximum likelihood estimator of the center of a normal distribution.

In order to get an estimate that traverses the planes in which we have missing data, we use the property that if we integrate the full domain of the first two coordinates of a three-dimensional normal distribution $f(x, y, z)$, we expect to get discrete values of a scaled 1-dimensional normal distribution. We have

$$G_{\text{traverse}}(z_i) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \lambda(x, y, z_i)\, dx\, dy \;\; = \beta e^{\frac{-(z-\overline{z})^2}{2\sigma^2}} \tag{4.39}$$

We then use curve fitting to estimate the scaling and importantly, the final injection site coordinate by the mean of the underlying normal distribution. The standard deviation can also be estimated here but a more accurate estimate is most likely found using the entire set of points in the first two dimensions. As an example, we generate a three-dimensional normal distribution and evaluate it in discretized images for each plane and normalize such that the density $\sum_{i,j,k} f(\mathbf{p}_{ijk}) = 1$, where $\mathbf{p}_{ijk}$ is a pixel coordinate with planer index i, j and section index k. We estimate the injection site by the mean

$$\mathbf{p}_{\text{injection}} = \sum_{i,j,k} \mathbf{p}_{ijk} f(\mathbf{p}_{ijk}) \;\; . \tag{4.40}$$

Here, the x- and y-coordinates will be estimated to machine precision while the z-coordinate will be erroneous due to the missing data. For discrete images, we replace the integrals by summations in (4.39) to get

$$\hat{G}_{\text{traverse}}(k) = \sum_{i,j} f(\mathbf{p}_{ijk}) \;\; , \tag{4.41}$$

and using the scipy.optimize.curve_fit curve-fitting tool in Python, we fit the parameters of the scaled normal distribution described in (4.39). Since the error of the standard approach is affected by how close the injection site is to one of the sections, we calculate the error in the estimate for different displacements of the sections with respect to the injection site. The result can be seen in Figure 4.6. Using curve fitting results in no error in this idealized example while estimating directly from the mean gives an error that varies depending on the distance between the closest section and the injection site.

4.5 General case: density estimation

An alternative to correcting for the missing data by normalization is to use the available points to estimate 2D densities for the points in each section and then estimate the density in between using these 2D densities. In other words, if D_{z_1}, D_{z_2} denotes two 2D density functions for the sections at coordinate z_1 and z_2, respectively, we will estimate the density of a section D_z between z_1 and z_2 by some function T such that $D_z = T(D_{z_1}, D_{z_2})$. A simple approach is to let T correspond to a linear approximation between D_{z_1} and D_{z_2}. Letting $\alpha = \frac{z-z_1}{z_2-z_1}$, we have

$$D_z = \alpha D_{z_2} + (1-\alpha)\, D_{z_1} \;\; , \tag{4.42}$$

with $z_1 \leq z \leq z_2$. We will use this approach as a reference to evaluate the quality of our other approaches. The next sections describe two approaches, namely, combining the linear approximation with a registration approach and a U-net dual input approach. Finally we discuss a potential way to combine both of these approaches.

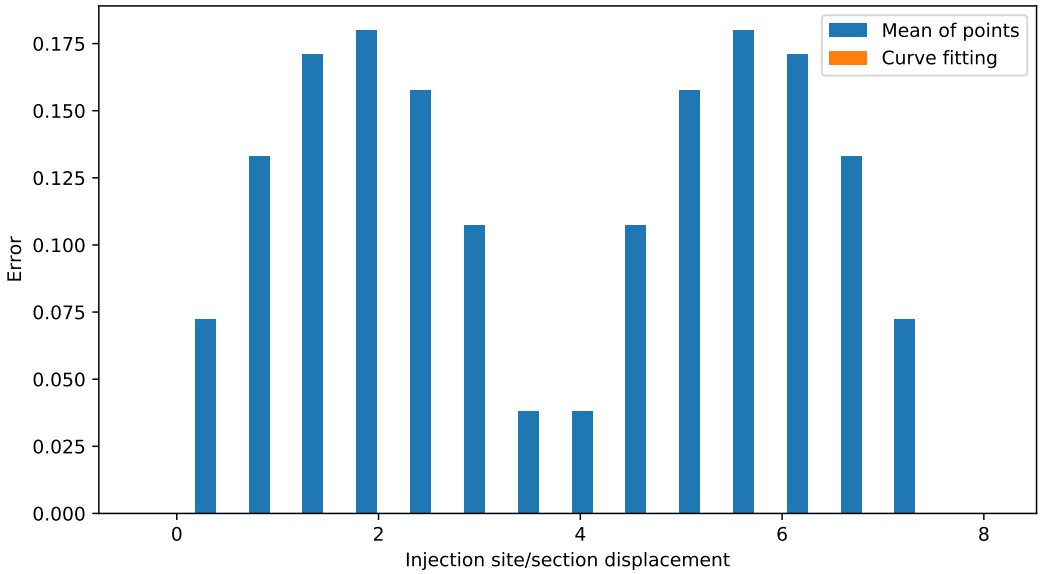


Figure 4.6: The injection site error as a function of the distance to the nearest section to the injection site using a simple density mean and scaled Gaussian curve fitting.

Registration approach

In this approach we adopt the linear approximation, but calculate a diffeomorphic transform between the two consecutive sections. Let $T(D_2, V) = D_1$ and $T(D_1, -V) = D_2$ denote a coordinate mapping T of density images D_2 to D_1 and D_1 to D_2. Here V denotes a vector field corresponding to the diffeomorphic coordinate map, and $T(D_i, 0) = D_i$. We then construct the estimated density image by

$$D_z = \alpha T(D_{z_2}, \alpha V) + (1-\alpha)\, T(D_{z_1}, -(1-\alpha)V) \ . \tag{4.43}$$

Implementation was done in python using the dipy package which implements cross-correlation diffeomorphic image registration [1].

U-net approach

In this section we will use a modified U-Net neural network structure to estimate the point densities in between the know sections. We view each section as a 2D density, and estimate multiple densitites in between. Given enough estimates, we effectively get a dense estimate of the density in 3D. To do this, we propose the following process:

1. Define a dual input prediction model which can estimate in-between section densities using adjacent section densities.

2. Pre-train the model on generated synthetic data with sections being varying distance apart.

3. Retrain the model on a the real data with different know spacing and/or a subset of extra sections annotated specifically for training.

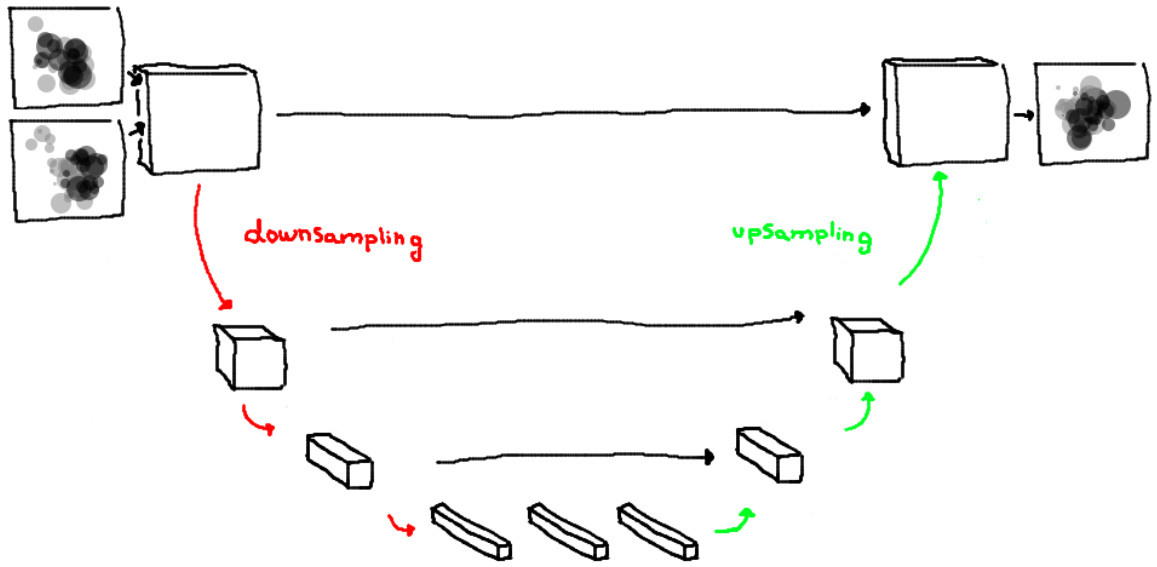


Figure 4.7: The dual-input U-net model for density prediction. Here each cubical block represents a set of feature maps progressively encoding the image space information into an increasing stack of features at the bottom of the network before reversing the process on the upsampling or decoding part of the network. Arrows from left to right indicate the possibility to transfer spatially matching features from the encoding pipeline to the decoding pipeline.

4. Estimate the density in between sections by bisecting from know or estimated sections. I.e. if D_0 and D_1 can be used to estimate $D_{0.5}$, then you can use D_0 and $D_{0.5}$ to get $D_{0.25}$ and so forth.

The initial model chosen was a modified U-Net. The U-Net is a tried and tested model commonly used for dense pixel classification. Here, we use it as an initial guess on a model that might work as a baseline. The U-Net was modified to take two input images. The images are stacked before being passed to standard U-Net model. In the final output layer the activation function was removed to not limit the range of values which could be outputted. A schematic of the U-Net model is seen in Figure 4.7. The network was defined in Python in a combination of Pure Tensorflow and Keras and optimized minimizing the Mean Squared Error using a Stochastic Gradient Decent optimizer.

Synthetic data

To generate synthetic data for training the network, we designed a stochastic data generation method simulating the process which happens in the target tissue. Recall that we inject cells into live tissue, let some time pass such that the cells migrate and integrate into the tissue before viewing the points and densities in sections of the 3D tissue volume. The cells migrate with varying speed depending on if they are inside or outside specific neural tracts which hasten migration.

We model the cell wandering as a diffusive process, as a Brownian motion with direction at each step chosen uniformly at random and with a step size depending on an underlying scalar field $V(x, y, z)$. To generate V we initialize a 3D array of zeros and

define tracts for the cells to move along. One main tract is created by the concatenation of two Brownian motions from the center of the volume. Each Brownian motion path is continued until they reach the edge of the image. The Brownian motion tracts are then regularized using a dampened Laplacian smoothing operation by, for curve points $p_1, \ldots, p_N$, iteratively setting

$$p_i \leftarrow (1-\alpha)p_i + \alpha(p_{i-1} + p_{i+1})/2 \ , \tag{4.44}$$

with $0 < \alpha < 1$ chosen to prevent oscillating behavior. Additionally, 1-5 random collateral tracts from the center point were also created similarly to the main tract. The pixels closest to each point in the tract curves were then set to 1 before filtering the image with a large Gaussian kernel. Some smoothed random noise was added to imitate random permeability of the tissue. Finally V was rescaled such that the largest distance at each step was 5. The result is a distribution of points clustered around the injection site with branches of points moving along the tracts.

For each realization of the point process a volume of 256^3 units was created with unique V. A random number between 3500 and 6500 of cells was placed in the center of the volume, at the intersection of all the defined tracts in V. Each point was stepped 1000 times in the V weighted Brownian motion. The points were then subdivided into layers with a thickness of 8. The section densities were estimated using a kernel density estimate. By default the sum of each section is 1 representing the spatial probability given one point in each section (see Figure 4.8).

We can scale the densities by the number of points to get an estimate of the intensity function $\lambda(x)$ (see Figure 4.9). Notice that the top and bottom sections shown in the top left and bottom right, respectively, are therefore significantly more transparent.

Experiments

The modified U-Net model was trained on 3583 point process realizations. From each realization, pairs of images were picked with a section spacing of 3 and 5 such that the middle image could be picked uniquely as the target image. With a train to validation ratio of approximately 1 to 40, the training set thus had 93604 examples saving 2348 for validation.

To assess the quality of prediction we predicted the entire set of validation examples and measured a number of different error measures. For D_{pred} the predicted density and D_{true} the true reference in the image domain Ω, the error measures are

- SAE: Sum of absolute error

$$\sum_{\Omega} |D_{\text{pred}} - D_{\text{true}}| \tag{4.45}$$

- SSE: Sum of Squared Error

$$\sum_{\Omega} (D_{\text{pred}} - D_{\text{true}})^2 \tag{4.46}$$

- SGNE: Sum of Gradient Norm Differences

$$\sum_{\Omega} |\nabla D_{\text{pred}} - \nabla D_{\text{true}}| \tag{4.47}$$

Figure 4.8: Unscaled synthetic density data. Top left is the first section bottom right is the last in row-wise order.

- BC: Bhattacharyya Distance

$$-\ln\left(\sum_{\Omega}\sqrt{D_{\text{pred}}D_{\text{true}}}\right) \tag{4.48}$$

- SDKL: Symmetrised Kullback–Leibler Divergence.

$$D_{KL}(D_{\text{pred}}, D_{\text{true}}) + D_{KL}(D_{\text{true}}, D_{\text{pred}}) \tag{4.49}$$

where

$$D_{KL}(D_1, D_2) = \sum_{\Omega} D_1 \log\left(\frac{D_1}{D_2}\right) \tag{4.50}$$

Figure 4.9: Synthetic density data sacled by the number of points in each section. Top left is the first section bottom right is the last in row-wise order.

Result are can be seen in Table 4.1 and are clearly very mixed, with the U-Net being beaten by a simple average in about half the cases. Interestingly, the gradient norm error also showed the mean was better the U-Net even though the average is a zeroth order measure with no gradient information present. The simple mean estimator performs surprisingly well. The mean could possibly be the optimal estimator if we assume that the underlying cell distribution is solely dictated by large scale noise which is partly present in this synthetic data.

4.6 Discussion

While performing adequately, naively using a standard neural network did not give groundbreaking results. One of the likely reasons is that the tracts in the data can be seen as a warping of the distribution within each section. However, diffeomorphic im-

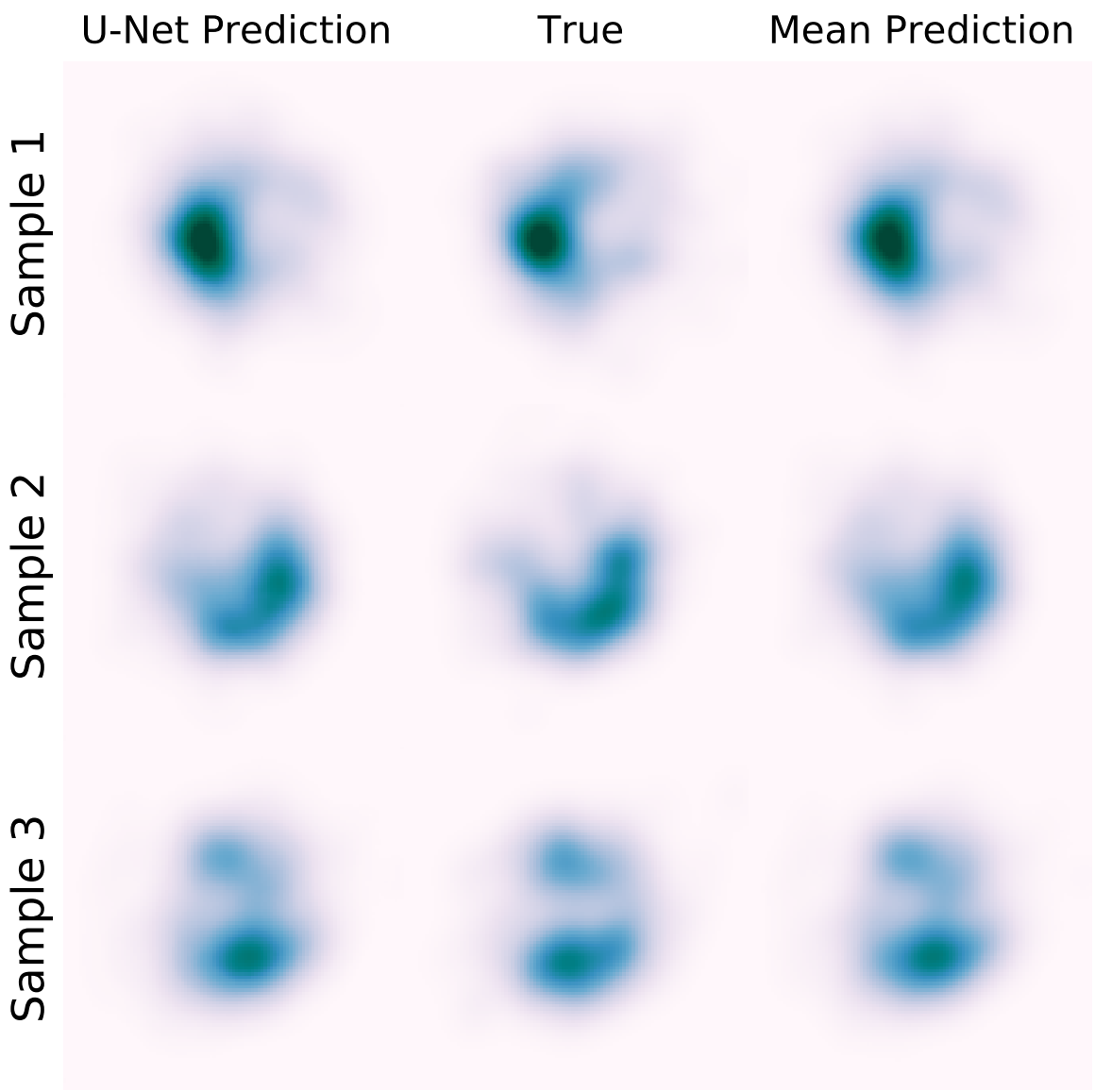


Figure 4.10: Predicted image density samples.

Table 4.1: Measures of differences between the estimated densities and the true reference. SAE: Sum of absolute error, SSE: Sum of Squared Error, SGNE: Sum of Gradient Norm Differences, BC: Bhattacharyya Distance, SDKL: Symmetrised Kullback–Leibler Divergence.

	SAE	SSE (10^{-6})	SGNE	BC	SDKL
Mean	0.1933	2.1720	**0.0401**	0.0121	**0.0706**
Reg	0.1978	2.3222	0.0410	**0.0083**	0.0741
U-net	**0.1920**	**2.1452**	0.0415	0.0118	0.0904

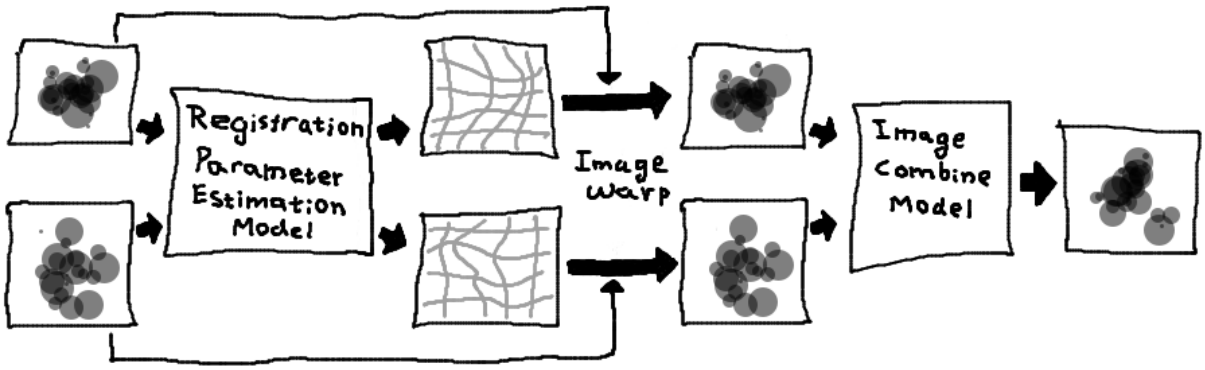


Figure 4.11: A dual-input stacked registration and density prediction model.

age registration alone does not seems to be able to present superior results. One reason could be that in the current implementation, the densities are not warped in a way that does guarantees they stay as densities. In this approach specifically, we do not model that the density of a point pattern should change when the divergence of the vector-field is non-zero. Spreading of the points should lower the density while contraction should increase the density. This constraint can be applied to deformation fields, but we did not explore this at this time. A combined approach is possible because there has been some developments in deep learning to estimate image registration parameters which could then be stacked with an image combination model. A proposed overall network structure is shown in Figure 4.11. Here I suggest using both images to estimate a warping for each image separately and then combine them in a second network. If training each network separately before stacking or training them simultaneously gives better results will have to be seen.

Chapter 5

Shape Measure from Currents in RKHS

This chapter contains the article found in pre-print [19], and is, at the time of writing, submitted for peer-review. This article is written by Pernille Emma Hartung Hansen. My contribution here is as a supportive role in the development of the method through discussion and ideas. I was also the main responsible for the development and implementation of the code for calculating the measures, generate synthetic data, and producing figures.

5.1 Abstract

Analysis of images of sets of fibers such as myelin sheaths or skeletal muscles must account for both the spatial distribution of fibers and differences in fiber shape. This necessitates a combination of point process and shape analysis methodology. In this paper, we develop a K-function for shape-valued point processes by embedding shapes as currents, thus equipping the point process domain with metric structure inherited from a reproducing kernel Hilbert space. We extend Ripley's K-function which measures deviations from spatial homogeneity of point processes to fiber data. The paper provides a theoretical account of the statistical foundation of the K-function and its extension to fiber data, and we test the developed K-function on simulated as well as real data sets. This includes a fiber data set consisting of myelin sheaths, visualizing the spatial and fiber shape behavior of myelin configurations at different debts.

5.2 Introduction

We present a generalization of Ripley's K-function for shape-valued point processes, in particular, for point processes where each observation is a curve in $\mathbb{R}^3$, a fiber. Fiber structures appear naturally in the human body, for example in tracts in the central nervous system and in skeletal muscles. The introduced K-function captures both spatial and shape clustering or repulsion, thus providing a powerful descriptive statistic for analysis of medical image of sets of fiber or more general shape data. As an example, Fig. 1 displays myelin sheaths in four configurations from different debts in a mouse brain. We develop the methodology to quantify the visually apparent differences in both spatial and shape distribution of the fibers.

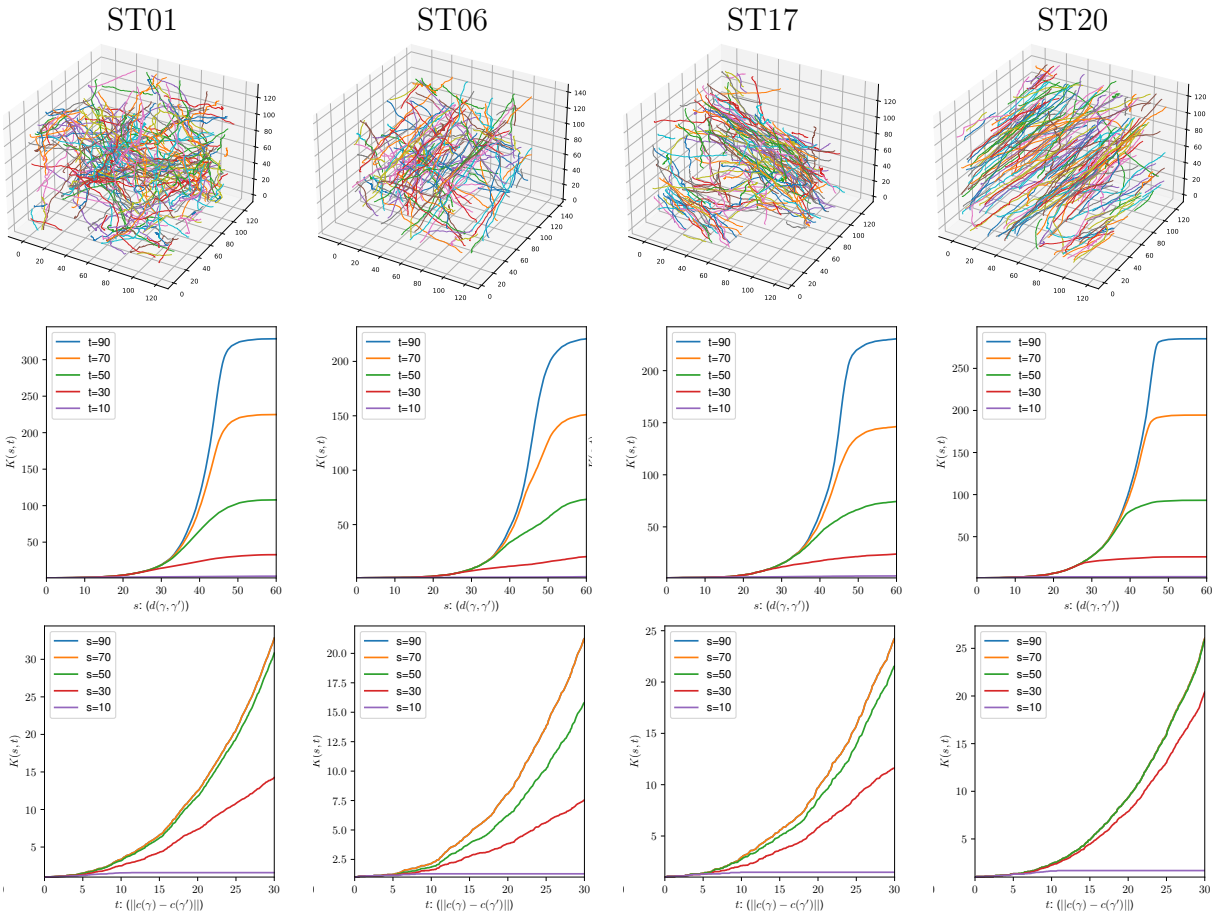


Figure 5.1: K-functions for samples of myelin sheaths. Each column corresponds to measured on a data set. (Top row): the centerlines of the myelin sheathed axons. (Middle row): The K-function for fixed values of t. (Bottom row): The K-function for fixed values of s.

5.2.1 Background

Ripley's K-function [38] is a well-known tool for analyzing second order moment structure of point processes [2] providing a measure of deviance from complete spatial randomness in point sets. For a stationary point process, $K(t)$ gives the expected number of points within distance t from a typical point. An estimator of Ripley's K-function for a point set $\{p_i\}_{i=1}^n$ inside an observation window W is,

$$\hat{K}(t) = \frac{1}{n\hat{\lambda}} \sum_{i \neq j} 1[\text{dist}(p_i, p_j) < t] \tag{5.1}$$

where $\hat{\lambda} = \frac{n}{|W|}$ is the sample intensity, $|W|$ is the volume of the observation window, and 1 is the indicator function. By comparing $\hat{K}(t)$ with the K-function corresponding to complete spatial randomness, we can measure the deviation from spatial homogeneity. Smaller values of $\hat{K}(t)$ indicate clustering whereas the points tend to repel each other for greater values.

Generalizations of Ripley's K-function have previously been considered for curve pieces in [8] and several approaches were presented in [48] for space curves. In this paper, we present a K-function inspired by the currents approach from [48] and provide the theoretical account for the statistical foundation.

The challenges when generalizing the K-function to shape-valued point processes arise in defining a distance measure on the shape space and determining a meaningful descriptive quantity that is well-defined and that we are able to estimate. In this paper, we provide a well-defined K-function for point processes in general metric spaces and use the embedding of shapes as currents to obtain a distance measure of shapes.

5.2.2 Contributions and outline

We construct the following two-parameter K-function for a curve-valued point process X

$$\hat{K}(t,s) = \frac{1}{|W|\lambda} \sum_{\gamma \in X : c(\gamma) \in W} \sum_{\gamma' \in X \setminus \{\gamma\}} 1[\|c(\gamma) - c(\gamma')\| \leq t, d_m(\gamma, \gamma') \leq s] \tag{5.2}$$

for $t, s > 0$, where $c(\gamma)$ is the center point of the curve γ, d_m the minimal currents distance with respect to translation, and λ is the spatial intensity of the center points. By introducing a second distance parameter, we are able to separate the spatial distance of the curves from the difference in shape, allowing us to measure both spatial and shape homogeneity.

The paper thereby presents the following contributions:

1. A K-function for shape-valued point processes along with a theoretical account for the statistical foundation.

2. We suggest a certain fiber process which we argue corresponds to complete randomness of points, an analogue of the Poisson process.

3. An application of the K-function to several generated data set and a real data set of myelin sheaths.

5.3 Shapes as currents

5.3.1 Shape-valued point processes

We model a random collection of shapes as a point process on the space of shapes. Shape spaces are usually defined as the space of embeddings $B_e(\mathcal{M}, \mathbb{R}^d)$ of a manifold $\mathcal{M}$ into $\mathbb{R}^d$ [3]. For example, the space of closed curves in $\mathbb{R}^3$ is $B_e(S^1, \mathbb{R}^3)$, where S^1 denotes the 1-sphere, and the space of fibers is $B_e(I, \mathbb{R}^3)$ for some real interval I.

Formally, a point process X on a metric space S is a measurable map from some probability space $(\Omega, \mathcal{F}, P)$ into the space of locally finite subsets of S. Thus, for each $\omega \in \Omega$, $X(\omega) \subseteq S$, and for every compact Borel set $B \subseteq S$, $X(\omega) \cap B$ is a finite set. Measurability of X means that all sets of the form $\{\omega \in \Omega | \#(X(\omega) \cap B) = m\}$, where $m \in \mathbb{N}_0$ and $B \subseteq S$ is a Borel set, must be measurable.

There are different ways to endow $B_e(\mathcal{M}, \mathbb{R}^d)$ with a metric [34]. In this paper, we consider the representation of shapes as currents embedded in the dual space of a reproducing kernel Hilbert space (RKHS). Thus, the RKHS metric induces a metric for our shape space. This can be combined with the Euclidean metric on $\mathbb{R}^d$ to obtain a suitable metric on $B_e(\mathcal{M}, \mathbb{R}^d)$. This approach is very useful due to its generality and computability, as it requires very little information about the shape.

5.3.2 Shapes as currents

Shapes are usually more difficult to work with than points, as they usually cannot be captured in any finite dimensional vector space. An approach already considered for anatomical structures [12] [59] is embedding shapes as currents. We will give a brief introduction to this setup and refer to [12] for a detailed description. We can characterize a piece-wise smooth curve $\gamma \in B_e(I, \mathbb{R}^d)$ by computing its path-integral of all vector fields w

$$V_\gamma(w) = \int_\gamma w(x)^t \tau(x) d\lambda(x), \tag{5.3}$$

where $\tau(x)$ is the unit tangent of γ at x and λ is the length measure on the curve. Likewise, an oriented hypersurface S embedded in $\mathbb{R}^d$ can be characterized by its flux integral of vector fields w

$$V_S(w) = \int_S w(x)^t n(x) d\lambda(x), \tag{5.4}$$

where $n(x)$ is the unit normal at x and λ is the surface area measure on S. These are both examples of representing shapes as currents, i.e. as elements in the dual space of the space of vector fields on $\mathbb{R}^d$. Formally, the space of m-currents C_m is the dual space of the space $C^0(\mathbb{R}^d, (\Lambda^m \mathbb{R}^d)^*)$ of differential m-forms.

It is not only curves and hypersurfaces that can be represented as currents. Let $\mathcal{M}$ be an oriented rectifiable sub-manifold of dimension m in $\mathbb{R}^d$ with positively oriented basis of the tangent space $u_1(x), ..., u_m(x)$ for all $x \in \mathcal{M}$. The sub-manifold $\mathcal{M}$ can be embedded into the space of m-currents as the current

$$T_\mathcal{M}(w) = \int_\mathcal{M} I(x) w(x) \Big(\frac{u_1(x) \wedge ... \wedge u_m}{|u_1(x) \wedge ... \wedge u_m|} \Big) d\lambda(x) \tag{5.5}$$

where $w \in C^0(\mathbb{R}^d, (\Lambda^m \mathbb{R}^d)^*)$ is an m-differential form and $I : T \to \mathbb{R}$ is a scalar function satisfying $\int_T |I(x)| d\lambda(x) < \infty$ [12]. Since shapes are embedded sub-manifolds, this means that shapes can be embedded into C_m.

5.3.3 Reproducing kernel Hilbert space metric on shapes

The space of m-currents C_m is continuously embedded into the dual space of a reproducing kernel Hilbert space (RKHS) H with arbitrary kernel $K_H : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d \times d}$ [12]. It follows from Riesz representation theorem that $v \in H$ can be embedded in the dual space H^* as the functional $\mathcal{L}_H(v) \in H^*$ defined by $\mathcal{L}_H(v)(w) = \langle v, w \rangle_H$ for $w \in H$.

Elements $v(y) = K_H(x, y)\alpha$ form a basis for H where $x, \alpha \in \mathbb{R}^d$, and basis elements in H are lifted to basis elements in H^* as $\delta_x^\alpha := \mathcal{L}_H(v)$ which are called the Dirac delta currents.

The element V_γ from (5.3) can be written in terms of the basis elements $\delta_{x_i}^{\tau_\gamma(x_i)}$ where $\tau_\gamma(x_i)$ are the unit tangent vectors of γ at x_i. This means that the curve γ is embedded into H^* as the 1-current

$$V_\gamma(w) = \int_\gamma w(x)^t \tau_\gamma(x) d\lambda(x) = \int_\gamma \delta_x^{\tau_\gamma(x)}(w) d\lambda(x) \tag{5.6}$$

where λ is the length measure on the curve. Furthermore, it is approximated by the Riemann sum of Dirac delta currents $V_\gamma(w) \approx \tilde{V}_\gamma(w) = \sum_i \delta_{x_i}^{\tau(x_i)\Delta x_i}(w)$ where x_i are

sampled points along the curve according to λ. The dual space H^* inherits the inner product from the inner product on the RKHS via the inverse mapping $\mathcal{L}_H^{-1}$, so that the inner product for two curves γ_1 and γ_2 in H^* is

$$\langle V_{\gamma_1}, V_{\gamma_2} \rangle_{H^*} = \int_{\gamma_1} \int_{\gamma_2} \tau_{\gamma_1}^t(x) K_H(x,y) \tau_{\gamma_2}(y) d\lambda_{\gamma_2}(x) d\lambda_{\gamma_1}(y). \tag{5.7}$$

Writing $||V_\gamma||_{H^*}^2 = \langle V_\gamma, V_\gamma \rangle_{H^*}$, we finally arrive at the currents distance of two curves γ_1 and γ_2

$$d_c(V_{\gamma_1}, V_{\gamma_2}) = ||V_{\gamma_1} - V_{\gamma_2}||_{H^*} = \Big(||V_{\gamma_1}||_{H^*}^2 + ||V_{\gamma_2}||_{H^*}^2 - 2\langle V_{\gamma_1}, V_{\gamma_2} \rangle_{H^*} \Big)^{1/2}. \tag{5.8}$$

In practice, we usually don't know the orientation of the curves, thus we choose to consider the minimal distance between them,

$$d(V_{\gamma_1}, V_{\gamma_2}) = \min\{ d_c(V_{\gamma_1}, V_{\gamma_2}), d_c(V_{\gamma_1}, V_{-\gamma_2}) \} \tag{5.9}$$

where $-\gamma_2$ denotes the curve with opposite orientation of γ_2. If the orientation of the data is important, this step may be omitted. From (5.6) we see that K_H serves as a weight of the inner product between $\tau_1(x_i)$ and $\tau_2(y_j)$ depending on the positions x_i and y_j.

5.3.4 A note on short lines and generalized Gaussian kernels

To illustrate the distance metric, consider the generalized Gaussian kernel

$$K_\sigma^p(x,y) = \exp\Big(\frac{-|x-y|^p}{2\sigma^p} \Big) \mathrm{Id}, \tag{5.10}$$

where $\sigma, p \in (0, \infty]$, and consider two lines of equal length parametrized by $l_u(t) = x_u + ut$, $l_v(t) = x_v + vt$ where $x_u, x_v, u, v \in \Re^d$ and $0 < t \leq T \in \Re$. For very short lines far from each other, i.e., $T/|x_u - x_v| \to 0$, we have,

$$\frac{d(\tilde{V}_{l_u}, \tilde{V}_{l_v})^2}{T^2} \to d_0 - 2 \exp\big(\frac{-|x_u - x_v|^p}{2\sigma^p} \big) d_1, \tag{5.11}$$

where $d_0 = u^t u + v^t v$ and $d_1 = \max(u^t v, -u^t v)$. Since the d_0 and d_1 are constants, and since the exponential and the square root functions are both monotonic, then in the limit, $d(\tilde{V}_{l_u}, \tilde{V}_{l_v})/T$ is one-to-one with $|x_u - x_v|^p$ which is one-to-one with $|x_u - x_v|$. Thus, for very short lines, d/T is one-to-one with the euclidean distance between the points x_u, and x_v. Further, in the limit $p \to \infty$ we have,

$$\frac{d(\tilde{V}_{l_u}, \tilde{V}_{l_v})^2}{T^2} \to \begin{cases} d_0 - 2\, d_1, & \text{when } |x_u - x_v| < \sigma, \\ d_0 - 2 \exp\big(-\frac{1}{2}\big)\, d_1, & \text{when } |x_u - x_v| = \sigma, \\ d_0, & \text{otherwise.} \end{cases} \tag{5.12}$$

Thus, for very short lines and very large exponents, $(d_0 - d^2/T^2)/(2d_1)$ converges to a unit step function in $|x_u - x_v|$ where the step is at σ.

5.4 The K-function

5.4.1 Statistical Setup

Let S be the image of the embedding of $B_e(I, \mathbb{R}^d)$ into C_1. For brevity, γ is identified with its representation in C_1. We model a random collection of curves as a point process X in S.

Let $c : S \to \mathbb{R}^d$ be a center function on the space of fibers in $\mathbb{R}^d$ that associates a center point to each fiber. A center function should be translation covariant in the sense that $c(\gamma + x) = c(\gamma) + x$ for all $x \in \mathbb{R}^d$. It could be the center of mass or the midpoint of the curve with respect to curve length. Let S_c denote the space of centered fibers wrt. c, i.e., those $\gamma \in S$ for which $c(\gamma) = 0$. We define $\gamma_c := \gamma - c(\gamma) \in S_c$ to be the centering of γ.

For Borel sets $B_1, A_1 \subset \mathbb{R}^d$ and $B_2, A_2 \subset S_c$, define the first moment measure

$$\mu(B_1 \times B_2) = \mathbb{E} \sum_{\gamma \in X} 1[c(\gamma) \in B_1, \gamma_c \in B_2]$$

and the second moment measure

$$\alpha((A_1 \times A_2) \times (B_1 \times B_2)) = \mathbb{E} \sum_{\gamma, \gamma' \in X}^{\neq} 1[c(\gamma) \in A_1, \gamma_c \in A_2] 1[c(\gamma') \in B_1, \gamma'_c \in B_2].$$

We assume that μ is translation invariant in its first argument, i.e.

$$\mu(B_1 \times B_2) = \mu((B_1 + h) \times B_2)$$

for any $h \in \mathbb{R}^d$. This is for instance the case if the distribution of X is invariant under translations. This implies that $\mu(\cdot \times B_2)$ is proportional to the Lebesgue measure for all B_2. Thus we can write

$$\mu(B_1 \times B_2) = |B_1| \nu(B_2)$$

for some measure $\nu(\cdot)$ on S_c. Note that the total measure $\nu(S_c)$ is the spatial intensity of the center points, i.e. the expected number of center points in a unit volume window. In applications, this will typically be finite. In this case, we may normalize ν to obtain a probability measure which could be interpreted as the distribution of a single centered fiber.

We define the reduced Campbell measure

$$C^!(A_1 \times A_2 \times F) = \mathbb{E} \sum_{\gamma \in X} 1[c(\gamma) \in A_1, \gamma_c \in A_2, X \setminus \{\gamma\} \in F]$$

where the "!" represents the removal of the point γ from X. By disintegration,

$$C^!(A_1 \times A_2 \times F) = \int_{A_1 \times A_2} P^!_{c,\gamma_c}(F) \mu(\mathrm{d}(c, \gamma_c)).$$

By the standard proof, we get for any measurable function $h : \mathbb{R}^d \times S_c \times \mathcal{N} \to [0, \infty)$

$$\mathbb{E} \sum_{\gamma \in X} h(c(\gamma), \gamma_c, X \setminus \{\gamma\}) = \int_{\mathbb{R}^d \times S_c} \mathbb{E}^!_{c,\gamma_c} h(c, \gamma_c, X) \mu(\mathrm{d}(c, \gamma_c)). \tag{5.13}$$

In particular,

$$\alpha((A_1 \times A_2) \times (B_1 \times B_2)) = \int_{A_1 \times A_2} \mathbb{E}^!_{c,\gamma_c} \sum_{\gamma' \in X} 1[c(\gamma') \in B_1, \gamma'_c \in B_2] \mu(\mathrm{d}(c, \gamma_c)). \quad (5.14)$$

Assume also that α is invariant under joint translation of the arguments A_1, B_1. Then

$$\mathcal{K}_{c,\gamma_c}(B_1 \times B_2) := \mathbb{E}^!_{c,\gamma_c} \sum_{\gamma' \in X} 1[c(\gamma') \in B_1, \gamma'_c \in B_2] \quad (5.15)$$

$$= \mathbb{E}^!_{0,\gamma_0} \sum_{\gamma' \in X} 1[c(\gamma') \in B_1 - c, \gamma'_c \in B_2] \quad (5.16)$$

$$= \mathcal{K}_{0,\gamma_0}((B_1 - c) \times B_2). \quad (5.17)$$

Assume that also $\mathbb{E}^!_{c,\gamma_c} h(c, \gamma_c, X)$ does not depend on c, which is true if the distribution of X is invariant over translations. Then, using (5.13) and the factorization of μ,

$$\mathbb{E} \sum_{\gamma \in X} 1[c(\gamma) \in W] h(c(\gamma), \gamma_c, X \setminus \{\gamma\}) = \int_{W \times S_c} \mathbb{E}^!_{c,\gamma_c} h(c, \gamma_c, X) \mu(\mathrm{d}(c, \gamma_c))$$

$$= \int_{W \times S_c} \mathbb{E}^!_{0,\gamma_0} h(0, \gamma_0, X) \mu(\mathrm{d}(c, \gamma_0)) = |W| \int_{S_c} \mathbb{E}^!_{0,\gamma_0} h(0, \gamma_0, X) \nu(\mathrm{d}\gamma_0) \quad (5.18)$$

From this it follows that

$$E_h = \frac{1}{|W|} \sum_{\gamma \in X} 1[c(\gamma) \in W] h(c(\gamma), \gamma_c, X \setminus \{\gamma\})$$

is an unbiased estimator of $\int_{S_c} \mathbb{E}^!_{0,\gamma_0} h(0, \gamma_0, X) \nu(\mathrm{d}\gamma_0)$. Furthermore, if $\nu(S_c)$ is finite,

$$\int_{S_c} \mathbb{E}^!_{0,\gamma_0} h(0, \gamma_0, X) \nu(\mathrm{d}\gamma_0) = \nu(S_c) \mathbb{E}_{\tilde{\nu}} \mathbb{E}^!_{0,\Gamma_0} h(0, \Gamma_0, X)$$

where Γ_0 is a random centered fiber with distribution $\tilde{\nu}(\cdot) = \nu(\cdot)/\nu(S_c)$ and $\mathbb{E}_{\tilde{\nu}}$ is expectation with respect to this distribution of Γ_0.

5.4.2 K-function for fibers

In order to define a K-function, we must make an appropriate choice of h. A seemingly natural choice for h that coincides with [48], is

$$h(c, \gamma_c, X) = \sum_{\gamma' \in X} 1[d(\gamma_c + c, \gamma') \leq t] = \sum_{\gamma' \in X} 1[d(\gamma, \gamma') \leq t].$$

However this choice allows the K-function to be a.s. infinite, due to the fact that $d(\gamma, \gamma') \leq \sqrt{2(||\gamma||^2_{H^*} + ||\gamma'||^2_{H^*})}$. If every curve in X has $||\gamma||_{H^*} \leq M$, e.g. if the length of fibers is bounded, then choosing $t \geq 2M$ results in any fiber in X having infinitely many neighbors within distance t.

A solution is to separate the spatial distance of the curves from the difference in shape by introducing another radius parameter for the distance between center points. Accounting for spatial distance with this parameter, we choose to minimize the influence of spatial distance by measuring the currents distance between the centered curves.

Thus, we choose h as

$$h(c, \gamma_c, X) = \sum_{\gamma' \in X \setminus \{\gamma\}} 1[\|c(\gamma) - c(\gamma')\| \leq t, d(\gamma_c, \gamma_c') \leq s] \tag{5.19}$$

for $s, t > 0$, where $||c(\gamma) - c(\gamma')||$ is the usual distance in $\mathbb{R}^d$ between center points.

Thus we define the empirical K-function for $t, s > 0$ as

$$\hat{K}(t, s) = \frac{1}{|W|\nu(S_c)} \sum_{\substack{\gamma \in X : c(\gamma) \in W, \\ \gamma' \in X \setminus \{\gamma\}}} 1[\|c(\gamma) - c(\gamma')\| \leq t, d(\gamma, \gamma') \leq s]. \tag{5.20}$$

Since $\nu(S_c)$ is the intensity of fiber centers, it is estimated by $N/|W|$ where N is the observed number of centers $c(\gamma)$ inside W. The K-function is the expectation of the empirical K-function

$$K(t, s) = \mathbb{E}\hat{K}(t, s) = \mathbb{E}_{\tilde{\nu}} \mathbb{E}^!_{0,\Gamma_0} h(0, \Gamma_0, X) = \mathbb{E}_{\tilde{\nu}} \mathcal{K}_{0,\Gamma_0}(B(0, t) \times B_c(\Gamma_0, s))$$

where $B(x, r) = \{y : ||x - y|| \leq r\}$ and $B_c(\gamma, s) = \{\gamma' : d(\gamma, \gamma') \leq s\}$.

5.4.3 K-function for general shapes

The currents metric and the K-function easily extends to shape-valued point processes with values in $B_e(\mathcal{M}, \Omega)$ for more general manifolds $\mathcal{M}$ and $\Omega \subset \mathbb{R}^d$. Shapes $\mathcal{A}, \mathcal{B} \in B_e(\mathcal{M}, \mathbb{R}^d)$ are embedded as m-currents $V_\mathcal{A}$ and $V_\mathcal{B}$ as in (5.5). Since C_m is continuously embedded into the dual RKHS H^*, we get the distance measure d_c between shapes.

If $c : B_e(\mathcal{M}, \Omega) \to \mathbb{R}^d$ is a center function, then we can generalize the K-function to a point process X with values in $B_e(\mathcal{M}, \Omega)$. Identifying elements of $B_e(\mathcal{M}, \Omega)$ with their embedding in H^*, we can write the same K-function

$$\hat{K}(t, s) = \frac{1}{|W|\lambda} \sum_{\substack{\mathcal{U} \in X : c(\mathcal{U}) \in W, \\ \mathcal{U}' \in X \setminus \{\mathcal{U}\}}} 1[\|c(\mathcal{U}) - c(\mathcal{U}')\| \leq t, d_m(\mathcal{U}, \mathcal{U}') \leq s] \tag{5.21}$$

for $t, s > 0$, where d_m is constructed as in Section 3.2 and λ is the spatial intensity of the center points.

5.5 Experiments

To obtain a measure of spatial homogeneity, Ripley's K-function for points is usually compared with the K-function for a Poisson process, $K_P(t) = \text{vol}(B_d(t))$, corresponding to complete spatial randomness. We are now in a more complicated situation where the K-function has two parameters and we do not have a notion of complete randomness of fibers. The aim of the experiments on generated data sets is to analyze the behavior of the K-function on different types of distributions and suggest a fiber process that corresponds to complete randomness. This will serve as a way to compare the results in 4.2.

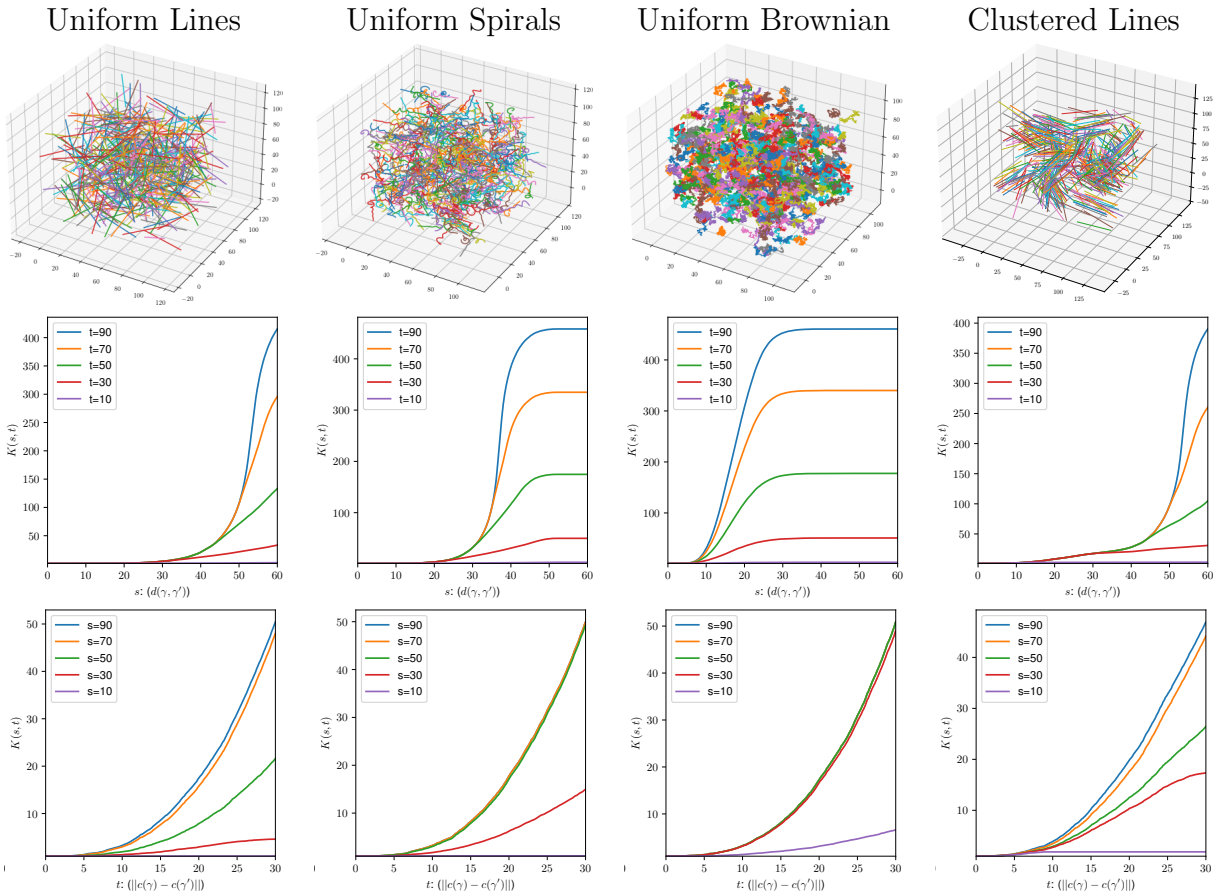


Figure 5.2: K-function on the generated data, where each column corresponds to one data set. (Top row): The data sets X_1, X_2, X_3 and X_4 described in 4.1. (Middle row): The K-function of the data set above for fixed values of t. (Bottom row): The K-function of the data set above for fixed values of s.

5.5.1 Generated data sets

The four generated data sets X_1, X_2, X_3 and X_4 each contain 500 fibers with curve length $l = 40$ and center points in $[0, 100]^3$ and is visualized in the first row of Fig. 5.2. Each data set is created by sampling center points from a distribution on $\mathbb{R}^3$ and fibers from a distribution on S_0, that is then translated by the center points. For the first three data sets, the center points are generated by a Poisson process and the fibers are uniformly rotated lines in X_1, uniformly rotated spirals in X_2 and Brownian motions in X_3. The data set X_4 has clustered center points and within each cluster the fibers are slightly perturbed lines.

To avoid most edge effects, we choose the window $W \approx [13, 87]^3 \subset \mathbb{R}^3$ for the calculation of the K-function. Furthermore, we choose a Gaussian kernel K_σ as in (5.10) with $p = 2$ and $\sigma = \frac{100}{3}$. Finally, c is defined to be the mass center of the curve.

The first row of Fig. 5.2 shows the generated data sets and the respective K-functions are visualized the second and third row. In the second row, $s \mapsto K(t, s)$ is plotted for fixed values of t. For example, the graphs with $t = 50$ show the expected numbers of fibers within currents distance s, where the distance of center points are 50 at most. Lastly in the third row, $t \mapsto K(t, s)$ is plotted for fixed values of s. Similarly, the graphs with $s = 70$ show the expected numbers of fibers with center point distance t when the currents distances are 70 at most. Thus, the graphs in the second row capture the fiber

shape difference of each data set whereas the graphs in the third row capture spatial difference.

It is distribution X_3 that we consider to a natural suggestion for a uniform randomness distribution of fibers. This is because Brownian motions are well-known for modelling randomness, thus representing shape randomness. And by translating these Brownian motion with a Poisson process, we argue that this distribution is a good choice.

Considering only the data sets with uniformly distributed center points, i.e., the first three columns of Fig. 2, we see a big difference in the second row of plots. This indicates that the K-function is sensitive to the change in shape. The K-function for the Brownian motions captures much more mass for smaller radii compared to the lines, with the spirals being somewhere in-between. The plots in the third row are very much as expected, since we generated the center points from a Poisson process. Finally, the second row plot for X_4 indicate a slight shape clustering when compared to the uniformly rotated lines. This makes sense, since each cluster is directed differently.

5.5.2 Application to myelin sheaths

Myelin surrounds the nerve cell axons and is an example of a fiber structure in the brain. Based on 3D reconstructions from the region motor cortex of the mouse brain, centre lines were generated in the myelin sheaths. The data sets ST01, ST06, ST17 and ST20 displayed in the first row of Fig. 1 represent the myelin sheaths from four samples at different debts.

For real shape-valued data sets, it very common that only parts of the shapes are observed. This is the case for many fiber data sets as well. This fact is important to have in mind when choosing c, since we should have a clear idea of when $c(\gamma)$ is observed, in order to get an unbiased estimate.

Since myelin sheaths tend to be quite long, we chose to divide the fibers of length greater than 40 into several fibers segments of length 40. This has the benefits, that the mass center is an appropriate choice for c and that the results are comparable with the results of Fig. 2, since the curves are of similar length.

The results of the estimated K-function on the four data sets ST01, ST06, ST17 and ST20 are visualized in Fig. 1, where $s \mapsto K(t, s)$ is plotted in the second row for fixed values of t and $t \mapsto K(t, s)$ is plotted in the third row for fixed values of s. The plots in the second row showing the fiber shape are very similar, resembling the fiber distribution of X_2. We notice a slight difference in ST20, where the graphs have a more pronounced cut off. When noticing the scale of the y-axis, we see that the expected number of neighbor fibers vary significantly between the data sets.

The third row plots indicate that the center point distributions of each data set is similar to the center point distribution of X_1, X_2 and X_3, which we generated from a Poisson process. For ST17, we notice a slight clustering of center points for $t \in [10, 15]$. The biggest difference is for the graphs for $t = 30$, indicating that the neighbors for fibers i ST20 are of more similar shape than the others.

Chapter 6

Applications in the Analysis of Biological Structures

In parallel to the main contributions, I have contributed to several projects, two of which required similar approaches and which both saw use in my shape measure method and served both as inspiration for the method and as a clear use of it. A few developments were made in connection to these two applications; these are presented here. In both applications, we measure the shape characteristics of one type of object in relation to another. In both cases, we have a control group and a disease model group and we want to test their statistical difference. This Chapter, therefore, outlines the data pipelines created to perform the analyses, the methods we used to perform image segmentation, and how we did a significance test on the difference between the two groups based on the shape measurement functions.

6.1 Case: Measuring Mitochondrial changes around Nodes of Ranvier in ALS model mice

ALS and the Nodes of Ranvier

Amyotrophic lateral sclerosis (ALS) is a neurodegenerative disease where motor neurons in the brain and spinal cord die, resulting in patients with muscle weakness and atrophy leading to paralysis [24]. As a consequence, patients suffer significantly reduced quality of life and most patients die withing 2-4 years after appearence of the first symptoms [29]. Neurons signalling is enhanced by the myelination of the axons of the neurons (See Figure 6.1). Myelination is a key component in both the efficiency and speed of signaling between neurons and to coordinate signal timings [16, 26]. Mutations in genes related to demyelination, the sporadic lack of myelination of neurons, has been observed in patients with ALS [63] and white matter pathology is a consistent finding [7] is a consistent finding, making it a key goal to understand if there is a causal connection between the two. In the brain, oligodendrocytes myelinate neuronal axons. These neuronal support cells (glia) each envelop parts of the neuron forming a myelin sheath covering parts of the axon in a repeating pattern. Between these myelin sheaths are myelin junctions often showing as small gaps which are called the Nodes of Ranvier. Since demyelination is observed in ALS patients, changes near the nodes, including node morphology, could be key to understanding the pathology of ALS.

In this work we assessed a large variety of questions in collaboration with Stine Has-

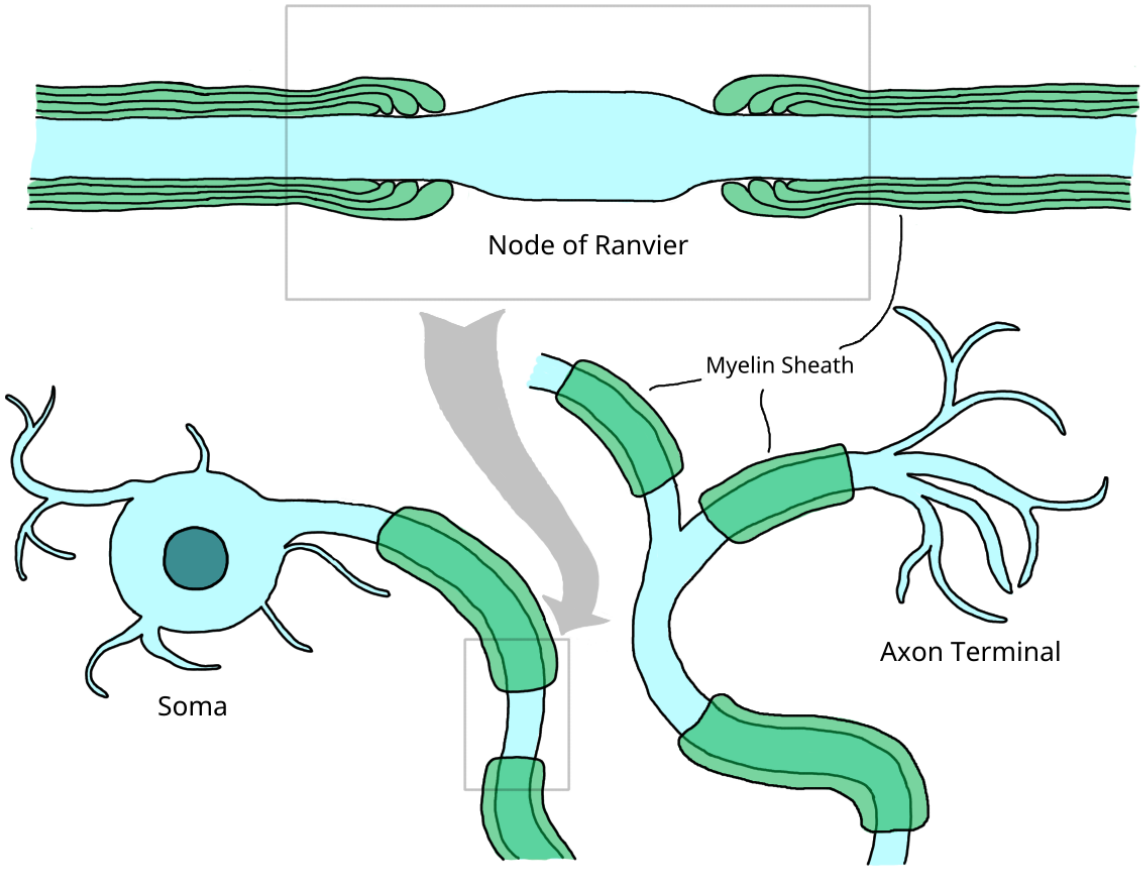


Figure 6.1: An ilustration of a neuron showing the axon, myelin sheaths and how the Node of Ranvier appear.

selholt. Stine formulated the biological questions, did sample preparation, and Serial Block-face Scanning Electron Microscopy (SBEM) imaging of 5 normal and 4 ALS model animals for which we could perform our study. From each animal we had an imaged block of approximately 32 μm^3 in size with a resolution of $700 \times 6000 \times 6000$ anisotropic $(45, 5.6, 5.6)$ nanometer pixels in 16-bit integer encoding. With each dataset at a size of roughly 70 Gb in uncompressed form, specialized methods for handling on common hardware was required.

For this work I helped design data pipelines and methods for handling multiple processing and analysis steps in the project. Specifically, I have annotated significant amounts of mitochondria for training and validating a neural network. I have likewise annotated almost 300 Nodes of Ranvier across thousands of sections for use as reference objects to measure the mitochondria using the method described in Chapter 3. I have described and quantified the morphology and features of each node slated for later analysis and publication. Through the work I have assisted supervision of two students tackling questions of Node of Ranvier counting and myelin thickness estimation. An overview of the work of which I was involved can be seen in Figure 6.2.

Annotation of SBEM Images

For prediction of mitochondria, we annotated a set of training data for each of the 9 datasets. In each dataset we annotated 2 full sections of size 6000×6000 pixels. A fully

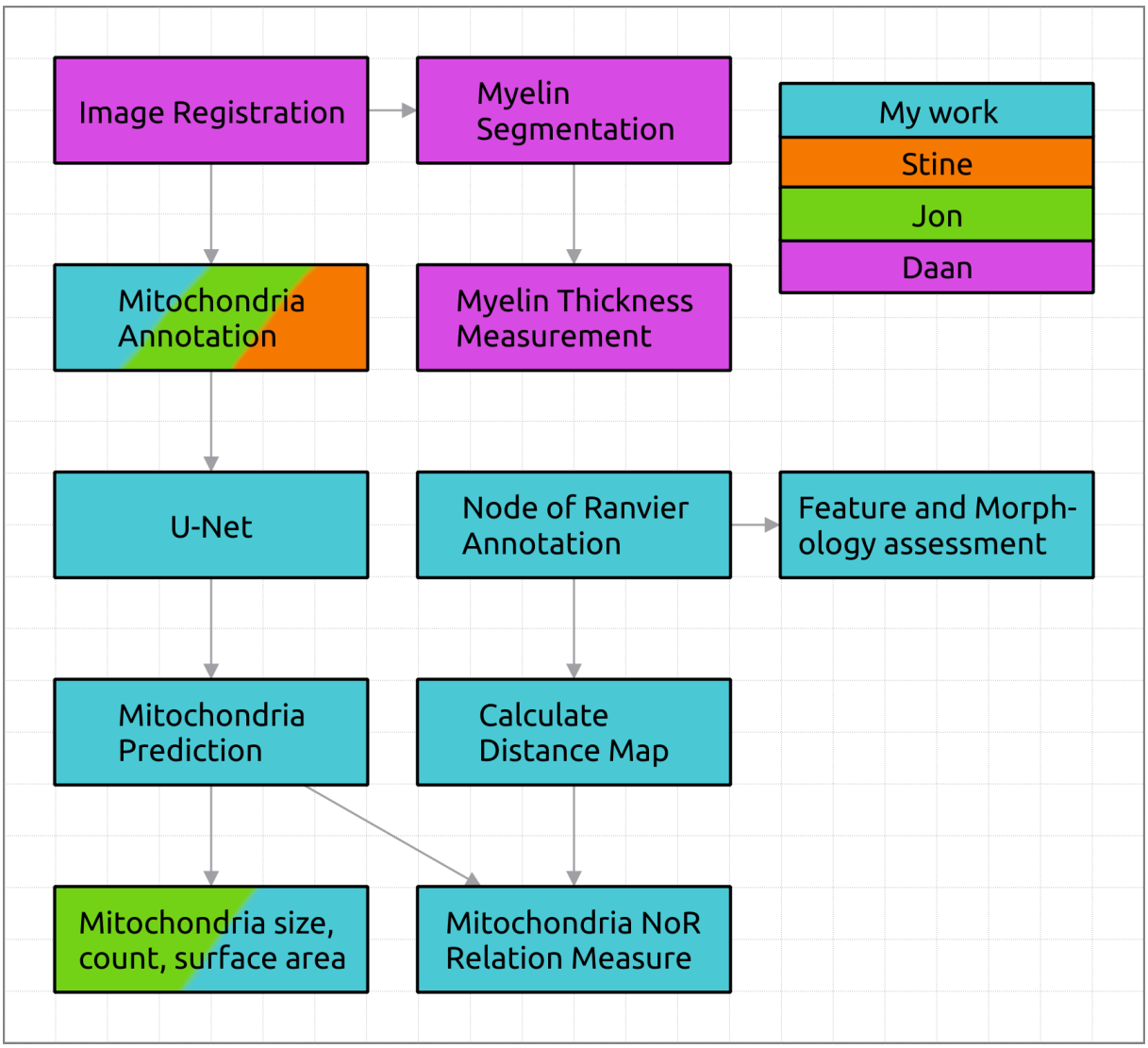


Figure 6.2: Overview of some of the key objectives I have been involved in in this project and how my work tied into the work of my colleagues and collaborators. Namely Stine Hasselholt, Jon Sporring, and Daan Janssen.

annotated section can be seen in Figure 6.3.

To give a representative training and validation split, each annotated section was tiled by a 3 by 3 grid and the 3 diagonal tiles was used for validation keeping the rest for training.

To calculate the measures around the nodes we needed an annotation of each of these nodes. A key obstacle here is that the node of Ranvier consist of cell membrane like most other structures in the tissue and is absent from any other directly distinguishing features. The nodes can therefore only be determined as a gap in the myelin of an axon. Additionally, the cell membranes were not so well-preserved in the images. Automatic approaches are therefore currently not able to detect these with the accuracy we needed. Through this work, Daan Janssen spent his externship exploring ways to accomplish this, ending up with an approach that, given myelin strands in the form of curves, worked on matching the strand end-points by finding end-points which were spatially and directionally close. This ended up not being accurate enough, probably because

- the node itself often is a point of directional change of the axon, possibly because

the myelin is comparatively stiffer. This means the direction of the myelin strands not necessarily corresponds for matching strands

- node of ranvier can be spatially close to each other, giving rise to wrong matching between strands
- many nodes have multiple myelinated and unmyelinated branches
- some myelin were not detected properly in a previous step, and errornous strands were present which did not correspond to a myelin sheath.
- while the literature usually puts a limited expected length of the NoRs, we have found the length to vary greatly meaning spatial closeness of two strand end points isn't always a good determining factor.

In the end, I decided to manually annotate the contour of each node in each section across all the datasets. A total of around 300 nodes were annotated using on the order of 10000 curves. An example of the annotation across a single node can be seen in Figure 6.4.

The morphology of the Node of Ranvier

Having annotated all nodes across the 9 datasets, I decided to assess a number of morphological features. For each node I classified them by their brancing pattern as m-c, where m denotes the number of myelinated brances and, c, denotes the number of unmyelinated and collateral brances. Thus a node marked with 2-1 has two myelinated brances and 1 unmyelinated branch. The pattern 1-c only has 1 myelin brance representing the ending of an axon. These are not accounted for.

I also assessed the length of the nodes. Given that many nodes had multiple branches, I decided to calculate the length in two ways. Firstly as the longest distance between two annotated edge points, and as the median of the 25 longest distances between annotated edge points. The latter method makes sense because it is more stable to the small variations that happens when manually annotating structures and better handle the cases where the myelin unevenly cover the node near the node end points.

Further, I noted the presence of other features such as mitochondrial activity at the node as well as the presence of synapses. I also found a curious feature which appeared to be either a recycling of myelin, recycling of cell-membrane or encapsulation of unwanted contents at numerous node. This structure was observed both either in the process of being expelled from the node or already separate from the node. The presence of this *messy contents* was also noted.

It is important to document features, not only to map out disease pathology, but also to contribute to built to the existing knowledge of morphological and quantitative features of important neuronal structures. This work is slated for later publication.

The distribution of branching patterns can be seen in Figure 6.5. Interestingly, the there is both a significant number of nodes displaying unmyelinated branches as well as a low but highly myelinated branch patterns. Similarly, we found there to be a small number of nodes with a high 4 myelinated branches. The distribution of node lengths can be seen in Figure 6.6. The curve mostly resemble something like a Poisson distribution except that the tail seems a bit heavy with a number of very long nodes. If this is not a statistical fluke, this could indicate either a spontaneous and simultaneous myelination of a process, or some dynamic not yet understood.

6.2 Case: Measuring Astrocytic changes in Huntington's disease model mice

In this chapter, I present contributions to work of which I am not the main author. This work is neither submitted nor published at the time of writing. The work was done during my stay at the Center for Translational Neuromedicin (CTN) at the University of Copenhagen. In this work, we are assessing whether there are geometrical or morphological changes to astrocytes around synapses in Huntingtons disease model mice. Due to the work not yet being publish, I will not be able to discuss the final results. I will here describe the parts where I contributed.

Assessing geometric and morphological changes in one object, the astrocyte, in relation to another, the synapse, falls squarely within the capabilities of my developed statistical shape measuring method described in Chapter 3, and as such, we have used this method for the analysis. Further, we assessed how to determine statistical significance of differences observed between the groups.

Biologically, the astrocytes have a supportive role for the glutamatergic synapses with a key role in the replenishment of neurotransmitters and maintaining the proper ion balance in the environment after an action potential has taken place. Astrocytes can often be seen seemingly reaching in towards the synaptic cleft. As such, the location and shape of the astrocyte is deeply connected to the synapse itself. Using the measurement functions described in Chapter 3, setting the synapse as the reference object, we are able to assess if there is reason to suggest that the volume or surface area of the astrocytes and neuronal processes are morphologically coupled to the disease condition at any particular distance from the synapse. In other words, does the disease condition shrink, enlarge, bend or change the curvature or other geometrical feature, of the astrocyte and neuronal process near the synapse?

The Data Pipeline

To accomplish the analysis, I designed the following pipeline in collaboration with the main author.

1. Biologists (Main author et al.) decide on regions of interest (ROI) across their datasets of disease and control population for where to perform the analysis.

2. Biologists (Main author et al.) annotates a training set of the astrocyte ROIs choosing visually varied set of samples among the population of ROIs. The synapses were manually annotated for all ROI because accuracy was particularly important for these, and because they are smaller making it feasible to do so.

3. The astrocyte data was pre-processed and the amount increased using data augmentation strategies for training and prediction by a U-Net neural network model.

4. The astrocyte predictions were cleaned up manually by the main author.

5. Each astrocyte was turned into a mesh representation using the marching cubes algorithm.

6. Each mesh was smoothed slightly in section-plane (x-y) to prevent jaggedness due to noisy pixel classification near the astrocyte edge. The mesh was smoothed signifi-

cantly in the section-plane (z) because imaging artifacts gave edges in the z-direction which were biologically inconsistent.

7. The distance from the synapse to each astrocyte mesh vertex was interpolated using distance transform on the underlying pixel grid.

8. The Astrocytes in each synaptic area was measured using the method described in Chapter 3.

9. The Biologists (Main author et al.) decides on a distance interval of interest to which we will do a statistical analysis.

10. The significance of the difference between the measurements from the disease and the control groups respectively where assess by p-value test calculated using the Monte Carlo Permutation test.

6.3 Neural Network Segmentation

Model Selection

As model selection we chose to use a U-Net neural network structure. This structure was chosen because it is tried and tested, efficient, and being a fully convolutional model, is fast on large datasets. Likewise, it was easy to modify with the addition of a weight input for controlling which parts of the images needed extra focus by the network. Since the original U-Net paper [41], improvements have been made for slightly better results. As such, a network variant including batch-normalization and strided convolution rather than max-pooling was chosen.

Data augmentation

Training data is always in short supply, a remedy is data augmentation which is the process of modifying or recombining the data you have to make more data is almost always beneficial. The standard approaches to data augmentation which we used are

- **Rotation:** Each image is rotated randomly between 0 and 360 degrees. This is a faithful simulation of the data since our images do not have a preferred direction in contrast to typical images of dogs and vases.

- **Flipping axis direction:** Changing the axis direction, i.e., flipping the image across an axis, yields a new valid image. Again, our images do not have a preferred direction so this is a faithful simulation.

- **Gaussian Noise Addition:** Adding noise to an image can seem counter-intuitive. However, the added noise seems to make fitted models more robust and generalize better. While adding noise raises the existing noise level slightly, it also represents a different realization of the noise pattern increasing the amount of information due to noise.

- **Elastic deformation:** The input image and labels are stretched and deformed equally to make a new image. Cellular and sub-cellular structures are already elastic, and a deformation simply leads to another plausible configuration of the

material and thus prepares the network for new realization that are slight deformations of those in the unaugmented dataset. To deform the images, the images are reinterpolated using a deformation field corresponding to the elastic deformation. The deformation field is created by first initializing source pixel coordinates to the original coordinates of the image. Each pixel is then assigned an independent and identically distributed random translation. The field of random translations are then smoothed such that the changes does not differ significantly on the smaller scale which seems unlikely.

Finally, I added a, to my knowledge, new form of data augmentation. For each label, I choose a random constant α and modified each pixel by the addition of α. This is a faithful simulation of the data because the exact color intensity in the images vary due a number of factors such as: protein contents, staining absorption, imaging parameters and material density. This is unlike the addition of noise because α is constant across a label.

Pixel-wise Weights

When predicting biological structures in neuronal tissue imaged using an electron microscope, the following observations were made.

- For typical structures, the surface area to volume ratio is small.
- Often, the structure of interest accounts for a comparatively low number of pixels in the full image giving rise to highly unbalanced classes for prediction.
- Some pixels are not labeled.

The first two points is likely to lead to lower quality of predictions using a standard loss function. This is because the loss function value the difference between all predicted and true label pixels equally, and since there is fewer pixels representing the object, the cost of the network to predict the object incorrectly, is low. This problem is even larger for the boundary.

A neural network loss can be modified by adding a weight to each pixel. These weights are passed in alongside the images, but only used in the calculation of the loss which in turn affects the back-propagation weight updates. We therefore modified the categorical loss function found in Keras, by adding a weight input which was multiplied to each pixel difference before the summation to a single loss value. Since we only need the weights in the loss calculation we only need them in the training phase.

For the weights I want the following to be true.

- The weights are 0 if and only if there is no labels present.
- The weight is higher the closer you are to a label edge.
- The weights converge quickly to a constant far from an edge.
- the sum of weights for each label is equal.

For the two projects, I developed two weight generation strategies. In the first strategy, I first defined a thick boundary strip B around each object. Let L denote a set representing the labeled object. The boundary strip was calculated by

$$B = (L \oplus D_r)/(L \ominus D_r) \ , \tag{6.1}$$

where $\oplus, \ominus$ represents Minkowski addition and subtraction, / represents set difference and D_r is a disc or radius r. The result is a set around the boundary with width $2r$. In effect we now have 3 regions in the image, namely $L \ominus D_r, (L \oplus D_r)^c$ and B where c denotes the set complement. Each of these regions was given a constant weight normalized such that the weight sum of each of the regions were the same.

These labels can easily be calculated using morphological dilation and erosion to replace the Minkowski operations. The result can be seen in Figure 6.7.

A more fine-grained weight generation strategy is: calculate a distance function from the edge between the two label regions, the inner and outer compartments of the object labels. Denote this distance function by $d(x, y)$. The weights are combined by an edge term in the form of

$$w_i = (d(x, y) + \beta)^{-\alpha_i} \ , \tag{6.2}$$

where i is 0 or 1, correspoding to whether the pixel is inside or outside the label region. $\beta > 0$ controls how steep the decrease in weights from the edge is. For $\alpha_i > 0$, I use a simple bijection search to determine the values of α_i such that

$$\int_{\text{inside}} w_0 = \int_{\text{outside}} w_1 \ . \tag{6.3}$$

This step ensures the weight contribution to the neural network learning is equally contributed from each group on average. Finally, I zero the weights which are in regions without labeling. Without any weight, the contribution to the network weight update becomes zero, meaning the network is indifferent to the distance between these pixel intensities. An example of weights generated using this method is shown in Figure 6.8.

Recombination weighting

When using a fully convolutional neural network to predict images, the prediction are done in large patches at a time. However, the prediction will usually be better in the center of the patch compared to the patch edges. A common strategy is to predict overlapping patches and then recombine them in the end forming a voting scheme.

Given a patch stride less than the width of the patch, each predicted output patch then overlaps with multiple other patches. To recombine the output patches, each patch output was added to a combined output prediction before using the argmax operation to determine a unique label. This approach is a voting scheme where the output level of certainty of each pixel is one of the determining factors. As an example, for one pixel sharing prediction by two patches, if one output patch gives the pixel 40% certainty for being class 1 and 60% for being class 2 while the other patch disagrees with a 20% to 80% certainty for class 1 and 2 respectively, the pixel is then classified as class 2. This is a common technique when predicting large images using overlapping patches in a fully convolutional network setting. However, the certainty of an output prediction always sum to 100% even though the prediction at the edges still has much less information to determine the correct label. To model this uncertainty, we weight each patch such that there's a higher weight near the center of each patch where the classification is expected to be the best.

The weights were generated by placing box of size 128×128 centered at each patch with weight 1. The box was smoothed with a Gaussian kernel with standard deviation $256/8.5 \approx 30$. The result can be seen in Figure 6.9a. The parameters were here chosen

visually such that any pixel had multiple votes from patches with the pixel near their center but with a greatly penalized weight for patches which only barely covered the pixel. These parameters can be estimated by hyper-parameter search, but we did not explore this further for this work. Since the combined output is a combination of several output patches, we can examine the ratio of which one single patch has weight within itself. In other words, if a patch was to be predicted, how much information originate from itself as opposed to other overlapping patches. In Figure 6.9b, we can see how much of a vote a patch has on the region it occupies in the final image. That is, the ratio r of which a patch i contributes to itself by its pixel weight $w_i(x, y)$ is given by the ratio

$$r(x, y) = \frac{w_i(x, y)}{\sum_{j \neq i} w_j(x, y)} \ . \tag{6.4}$$

Notice that in this example the result is more narrow than the original patch weight. Generally, a larger stride will make the ratio wider while a smaller stride will make the ratio of contribution more narrow. A 1D view of the weight overlap can be seen in Figure 6.10a showing that plenty of patches contribute to the same final prediction. The combined weights from multiple patches can be seen in Figure 6.10 showing good coverage in the resulting prediction.

6.4 Statistical Tests on Measure Functions

To assess the significance of the difference between the groups, we want to perform a null hypothesis significance test. However, we are faced with the difficulty in understanding the underlying test statistic. Therefore, we choose to directly measure the significance using a Monte Carlo Permutation Test. To accomplish this we need to define an appropriate distance measure between the groups of shape measure functions. Then, to perform the significance test, we calculate the difference τ between the groups once, and perform the following repeated action. Let N, M denote the number of measure functions in each group. We then do the following a total of L times

1. Combine all $N + M$ functions into one pool.
2. Randomly split the pool of function into groups of size N and M.
3. Calculate the distance τ_i for the two randomly sampled groups.

The p-value can now directly be estimated by the proportion of $\tau_i > \tau$. I.e

$$p \approx \frac{1}{L} \sum_{i=1}^{L} 1_{\tau_i > \tau} \ . \tag{6.5}$$

We calculate distance τ between the groups by first calculating the mean function

$$f_{\text{mean}}(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x) \ , \tag{6.6}$$

in each group where n is the number of functions in the group. We then measure the distance between the mean functions as the integral of the absolute difference between them. I.e.

$$d(f, g) = \int_{r_{\min}}^{r_{\max}} |f(r) - g(r)| dr \ . \tag{6.7}$$

Here $r_{\min}$ and $r_{\max}$ determine the interval of interest of which there is expected to be a significant difference. This step needs special thought since first looking and then choosing an interval can be used as an intentionally or unintentionally way of p-hacking.

Example statistical test

Since both articles are still being written, we cannot here show the results, but as an example I here show the result of a p-value calculation for the synthetically generated object clusters used in the article presented in Chapter 3 seen here in Figure 6.11 for convenience.

Calculating the p-value for the unnormalized groups is shown in Figure 6.11a, unsurprisingly, yields a rather high approximate p-value of 0.57. Conversely, the normalized groups, shown in Figure 6.11b, gives an approximate p-value of $4 \cdot 10^{-5}$. It can difficult to spot the characteristic difference in how these two groups was formed visually, but both the normalized graph, and the calculated p-value clearly indicate that these are very distinct groups.

6.5 Using biological Structures in Image Normalization

In both projects we had multiple datasets which each were imaged under similar conditions, but due to a number of circumstances, most importantly, that the imaging device is recalibrated before imaging each of the datasets, the datasets vary in quality, signal to noise ratio, dynamic range and contrast. It is normal to standardize the input data before feeding into a neural network. This is a problematic approach for these biological samples because we are often faced with large structures, much larger than the one in which we are interested, which has a significantly different visual characteristic which is detrimental to the stability of the standardization process (See Figure 6.3 for examples of these large structures).

To get a stable normalization of the images, we use the annotation of a few significant biological structures as reference intensities in order to perform brightness and contrast adjustment on each dataset. The goal is to map each image into a common reference frame irrespective of the varying image content, and to allow for transfer learning in the training and prediction of the datasets. We note that a linear mapping of pixel intensities correspond to a brightness and contrast adjustment.

To accomplish this, we annotate two areas with distinct and uniform intensity values and calculate the mean intensity of these. Let $s_{\min}$ and $s_{\max}$ denote the means of these two regions with $s_{\min}$ denoting the smallest and $s_{\max}$ denoting the largest. We then calculate a linear map such that these areas on average will have intensity equal to two target intensities $t_{\min}$, $t_{\max}$. The intensity map from an intensity value s to an intensity value t is thus calculated by

$$t = \frac{t_{\max} - t_{\min}}{s_{\max} - s_{\min}}(s - s_{\max}) + t_{\max} \ . \tag{6.8}$$

For training the neural network, we chose a few areas consisting primarily of cytosol and a few strips of myelin. These structures approximately represent the brightest and darkest objects in the images. We mapped these to 1 and -1 respectively for each dataset.

This method can also be extended to a non-linear intensity mapping, by annotating more than two distinct biological structures and fitting a mapping to the mean intensity of these. This was not explored further here.

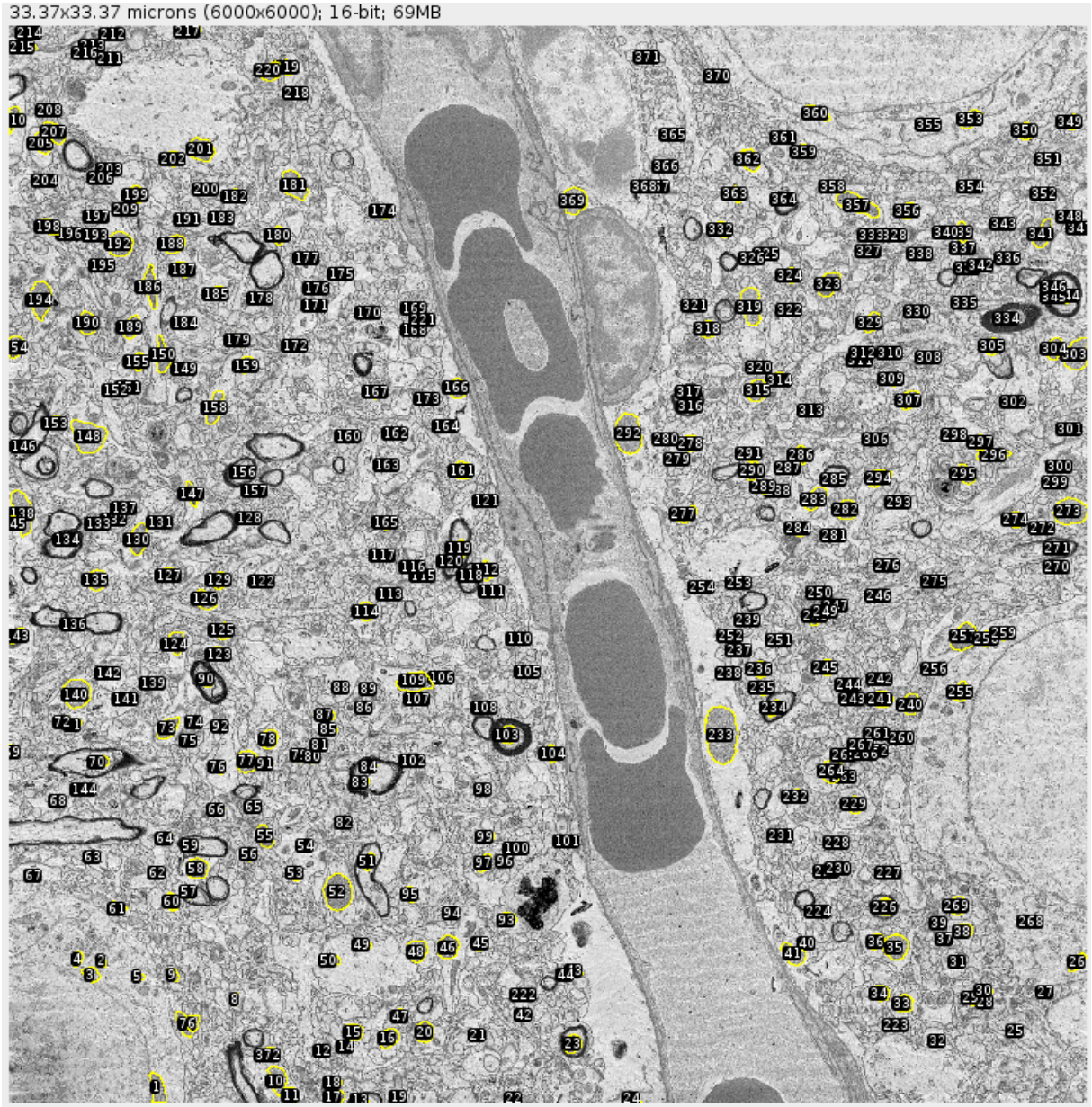


Figure 6.3: Example of one of the sections where mitochondria were annotated. Mitochondria annotations are shown in yellow.

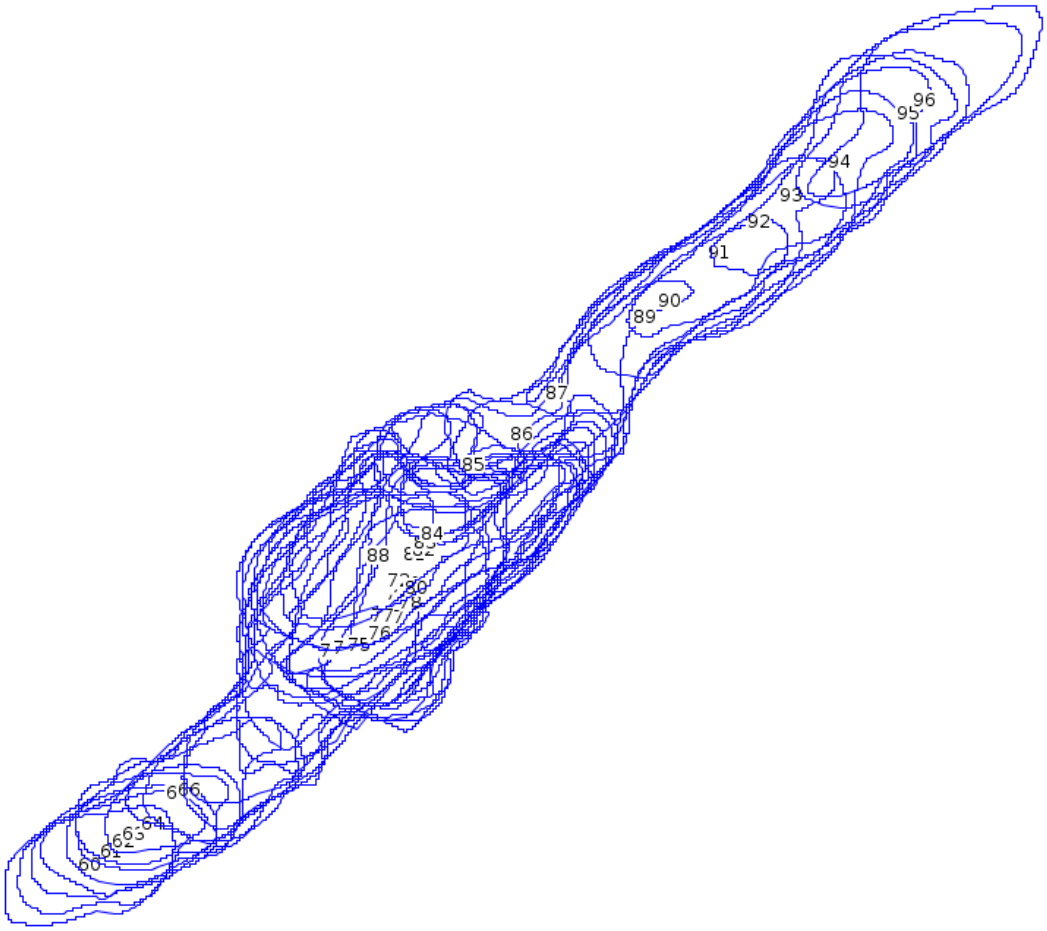


Figure 6.4: The set of curves annotated for a single Node of Ranvier.

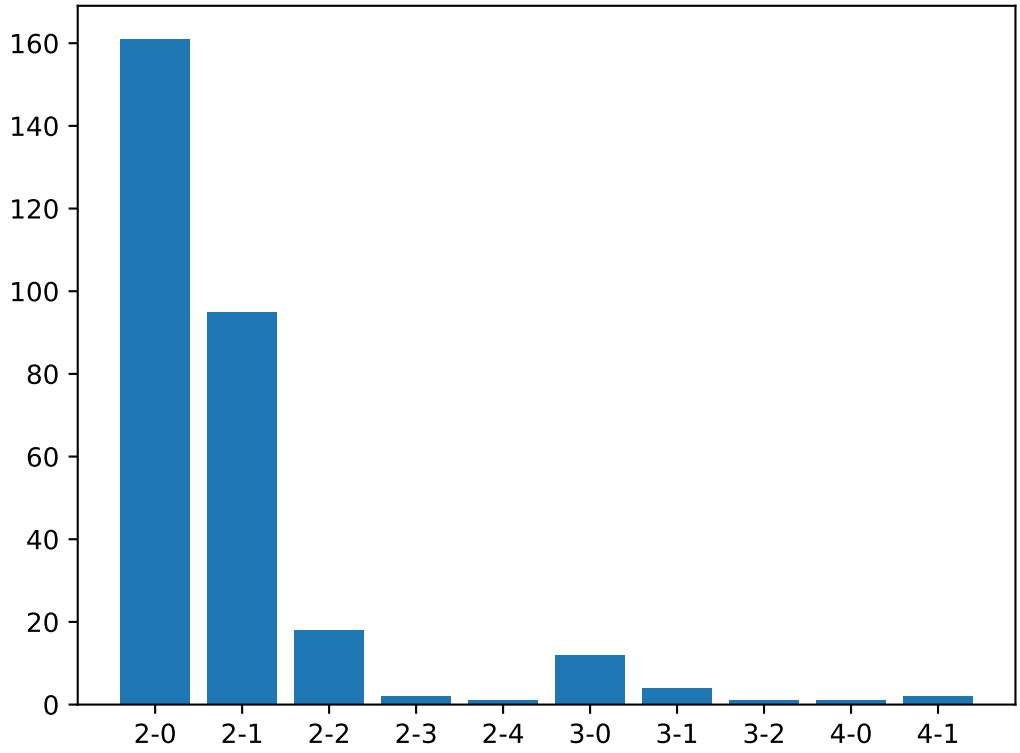


Figure 6.5: Distribution of Node of Ranvier branching types. The types are labeled as m-c where m denotes the number of myelinated branches and c denotes the number of unmyelinated branches.

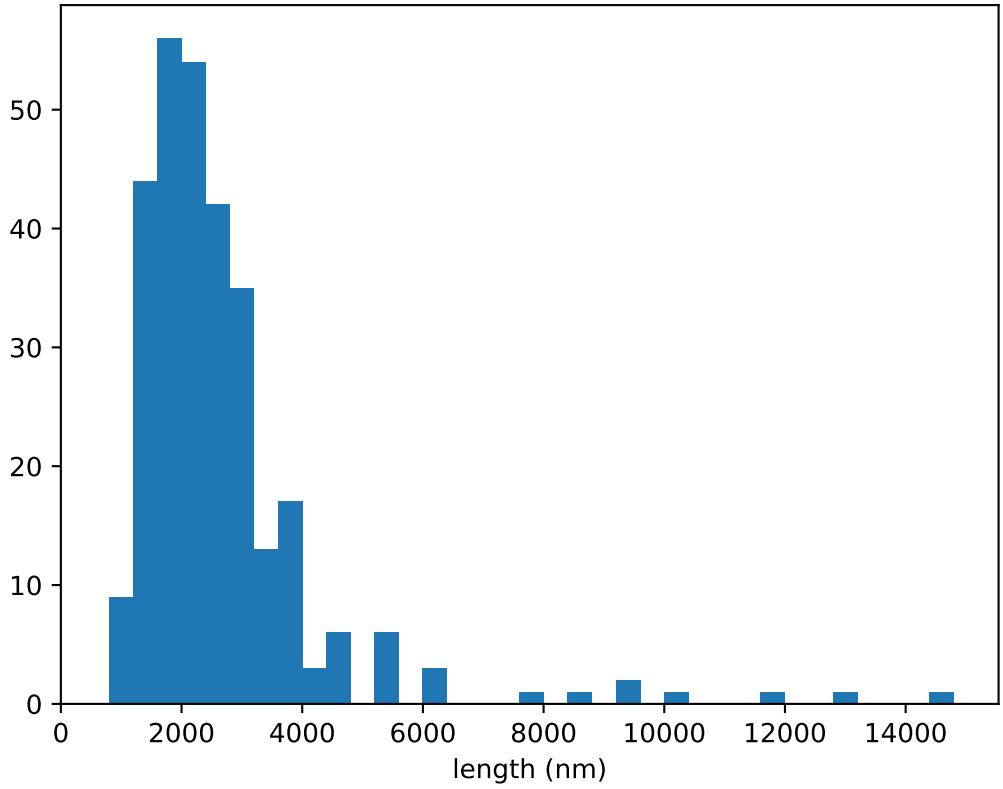
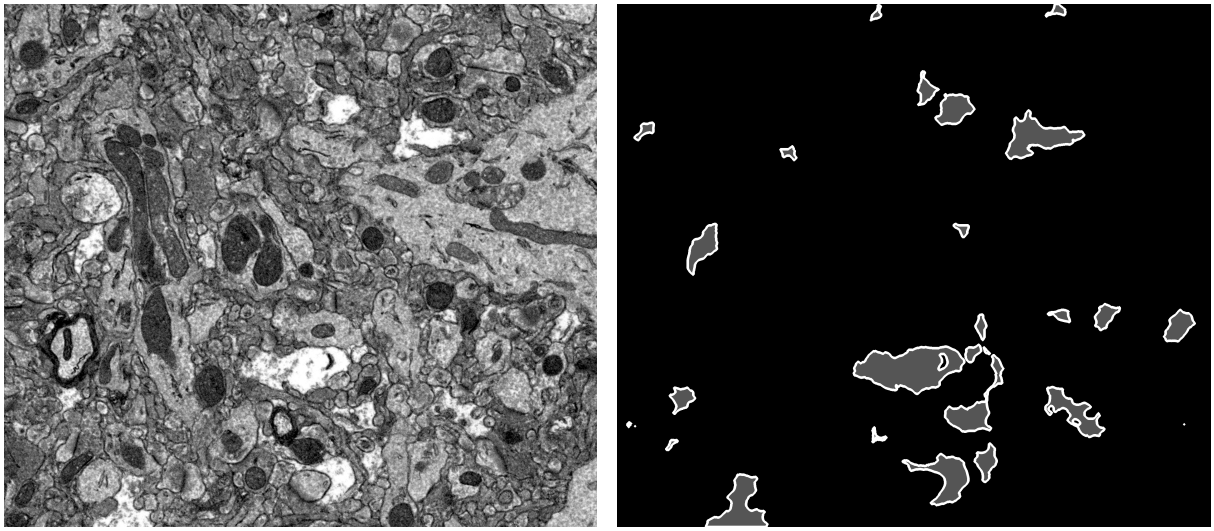


Figure 6.6: Distribution of Node of Ranvier lengths.



(a) SS-TEM image

(b) Pixel weights

Figure 6.7: text

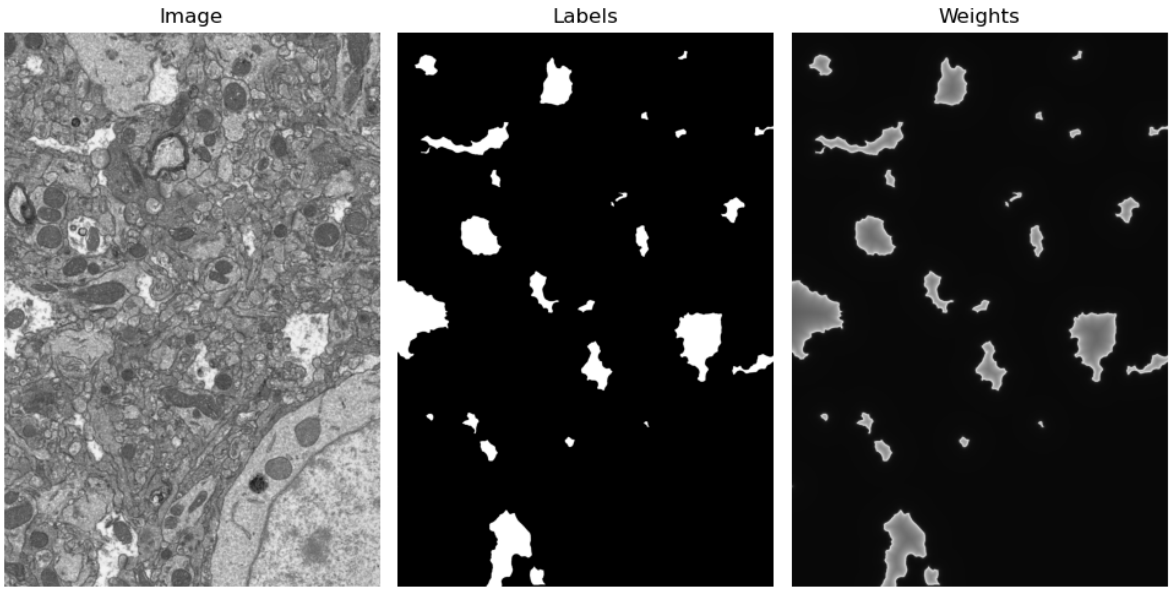
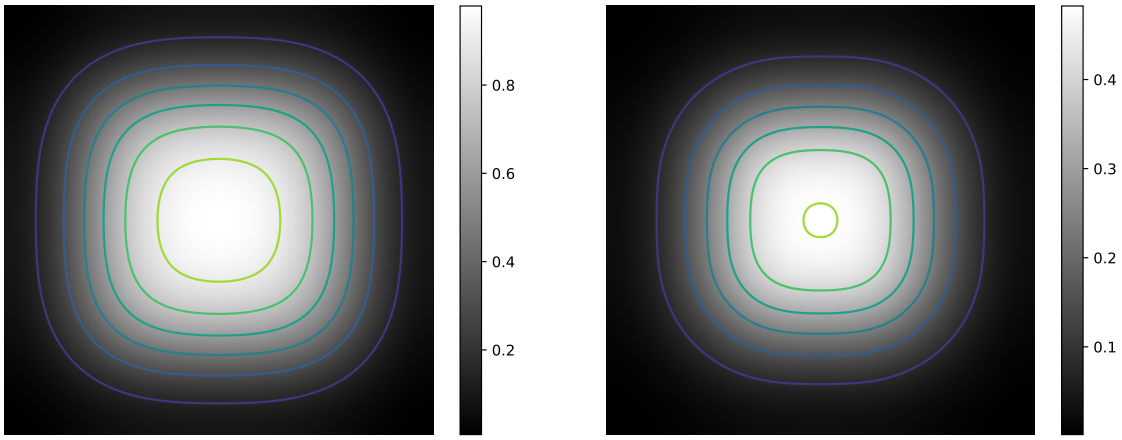


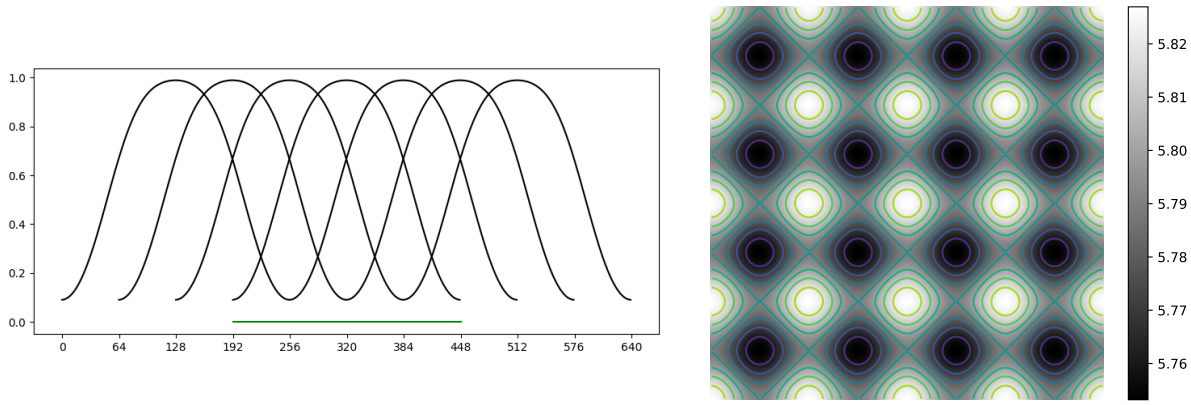
Figure 6.8: text



(a) patch weights

(b) weight ratio

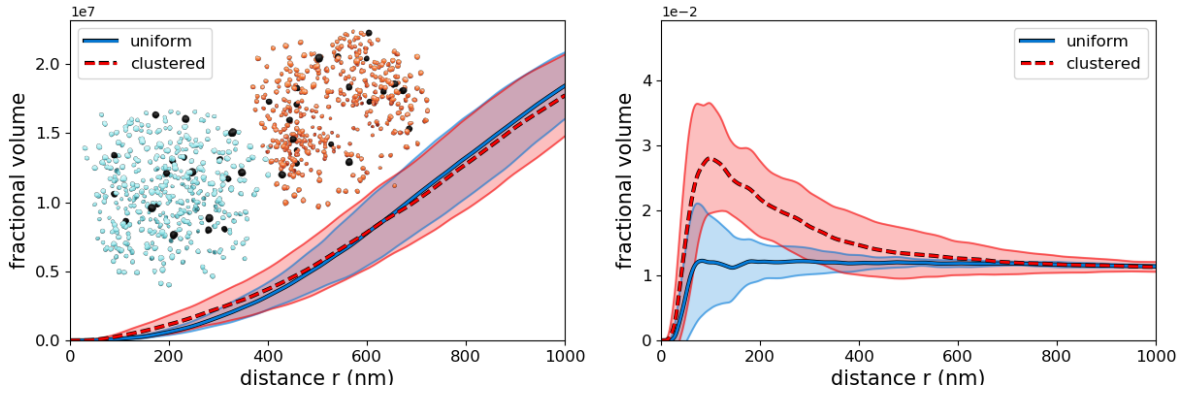
Figure 6.9: (left) Visualization of how weights contribute in the images in a 2D representation. (right) ratio of weight contribution compared to the weight from other patches. The ratio of weight contributions is more narrow here due to the prediction patch stride. Higher stride will make the ratio wider and more focused for smaller stride values.



(a) patch overlap in 1D

(b) weight total

Figure 6.10: a) 1D representation of the weight overlap. The size of a single patch is shown as a green line. (b) The combined patch weights for part of the resulting prediction shown in the size of an image patch.



(a) $\mu_{00}(X, Y^r)$ volume measure.

(b) $\nu_{00}(X, Y^r)$ normed volume.

Figure 6.11: Example of the volume measures on uniformly distributed and clustered spheres. (a) A slight difference can be seen directly in the $\mu_{00}(X, Y^r)$ graph, (b) but the clustering is clearly visible in $\nu_{00}(X, Y^r)$.

Chapter 7

Conclusion

This dissertation has worked to advance the interdisciplinary field of computer science and biology focusing on geometric problems that arise in this intersection. Throughout the work, we have specifically addressed how to handle a number of limitations that often arise in biological data.

In Chapter 2, we have shown how a model-based approach using known biological properties can be used for image registration. This work shows how to handle the problem theoretically and shows its superior accuracy. Follow-up work should be focused on reducing the time needed for annotation of the structures and on a formulation that can can calculate the drift more locally, as a section-by-section estimate not affected by the drift in nearby sections.

In Chapter 3 we present a shape-measuring method that takes into account the distance to a reference object and takes into account its morphology with clear uses presented in Chapter 6 on ALS and Huntington model mice respectively. This method in particular presents a perspective on object relations that has not been explored fully and could be the foundation of a new field of study. Future work could extend the theoretical formulation to boundary limited measures and to better understand the equivalence under the measure in a more general setting.

In Chapter 4 we show how to do point statistics when the points move from a common source point and we only have sparse knowledge about the point process realization. This method, as well as the relational shape-measure from Chapter 3, are similar in that they ask the question of what happens when you measure objects or points in a distance parameterized manor. This is a general topic with a plethora of open questions which will be exciting to see further developed. We hope that this work can be a stepping stone to such an excursion.

Finally, in Chapter 5, we present a summary statistic combining the currents distance measure on curves and the K-function from spatial statistics. We show how the measure performs on synthetic data and myelin sheath data.

As the fields of bioimaging, biology, and pathology evolve, so has the computational and statistical methods evolved through the work presented in this dissertation, if just by a couple of steps. It is with the utmost anticipation for the future progress in science, and in particular, the above topics, that I close off this final chapter.

Bibliography

[1] Brian B Avants, Charles L Epstein, Murray Grossman, and James C Gee. Symmetric diffeomorphic image registration with cross-correlation: evaluating automated labeling of elderly and neurodegenerative brain. *Medical image analysis*, 12(1):26–41, 2008.

[2] Adrian Baddeley, Ege Rubak, and Rolf Turner. *Spatial Point Patterns: Methodology and Applications with R*. CRC Press, November 2015. Google-Books-ID: rGbmCgAAQBAJ.

[3] Martin Bauer, Martins Bruveris, and Peter W. Michor. Overview of the Geometries of Shape Spaces and Diffeomorphism Groups. *Journal of Mathematical Imaging and Vision*, 50(1-2):60–97, September 2014.

[4] Carles Bosch, Albert Martínez, Nuria Masachs, Cátia M Teixeira, Isabel Fernaud, Fausto Ulloa, Esther Pérez-Martínez, Carlos Lois, Joan X Comella, Javier DeFelipe, et al. Fib/sem technology and high-throughput 3d reconstruction of dendritic spines and synapses in gfp-labeled adult-generated neurons. *Frontiers in neuroanatomy*, 9:60, 2015.

[5] Corrado Calì, Jumana Baghabra, Daniya J Boges, Glendon R Holst, Anna Kreshuk, Fred A Hamprecht, Madhusudhanan Srinivasan, Heikki Lehväslaiho, and Pierre J Magistretti. Three-dimensional immersive virtual reality for studying cellular compartments in 3d models from em preparations of neural tissues. *Journal of Comparative Neurology*, 524(1):23–38, 2016.

[6] Peter B Canham. The minimum energy of bending as a possible explanation of the biconcave shape of the human red blood cell. *Journal of theoretical biology*, 26(1):61–81, 1970.

[7] Guangxiang Chen, Baiwan Zhou, Hongyan Zhu, Weihong Kuang, Feng Bi, Hua Ai, Zhongwei Gu, Xiaoqi Huang, Su Lui, and Qiyong Gong. White matter volume loss in amyotrophic lateral sclerosis: a meta-analysis of voxel-based morphometry studies. *Progress in Neuro-Psychopharmacology and Biological Psychiatry*, 83:110–117, 2018.

[8] Sung Nok Chiu, Dietrich Stoyan, Wilfrid S. Kendall, and Joseph Mecke. *Stochastic Geometry and Its Applications*. John Wiley & Sons, June 2013.

[9] André Collignon, Frederik Maes, Dominique Delaere, Dirk Vandermeulen, Paul Suetens, and Guy Marchal. Automated multi-modality image registration based on information theory. In *Information processing in medical imaging*, volume 3, pages 263–274, 1995.

[10] Sune Darkner and Jon Sporring. Locally orderless registration. *IEEE transactions on pattern analysis and machine intelligence*, 35(6):1437–1450, 2012.

[11] Philip M Dixon. Ripley's k function. *Wiley StatsRef: Statistics Reference Online*, 2014.

[12] Stanley Durrleman, Xavier Pennec, Alain Trouvé, and Nicholas Ayache. Statistical models of sets of curves and surfaces based on currents. *Medical Image Analysis*, 13(5):793 – 808, 2009.

[13] A Einstein. On the movement of small particles suspended in stationary liquids required by the molecularkinetic theory of heat. *Ann. d. Phys*, 17(549-560):1, 1905.

[14] Moses Ender, Jochen Joos, Thomas Carraro, and Ellen Ivers-Tiffée. Quantitative characterization of lifepo4 cathodes reconstructed by fib/sem tomography. *Journal of the electrochemical society*, 159(7):A972, 2012.

[15] Riccardo Fesce, Fabio Grohovaz, Flavia Valtorta, and Jacopo Meldolesi. Neurotransmitter release: fusion or 'kiss-and-run'? *Trends in cell biology*, 4(1):1–4, 1994.

[16] R Douglas Fields. Myelin—more than insulation. *Science*, 344(6181):264–266, 2014.

[17] Nikolay Gavrilov, Inna Golyagina, Alexey Brazhe, Annalisa Scimemi, Vadim Turlapov, and Alexey Semyanov. Astrocytic coverage of dendritic spines, dendritic shafts, and axonal boutons in hippocampal neuropil. *Frontiers in cellular neuroscience*, 12:248, 2018.

[18] Ute Hahn. A studentized permutation test for the comparison of spatial point patterns. *Journal of the American Statistical Association*, 107(498):754–764, 2012.

[19] Pernille EH Hansen, Rasmus Waagepetersen, Anne Marie Svane, Jon Sporring, Hans JT Stephensen, Stine Hasselholt, and Stefan Sommer. Currents and k-functions for fiber point processes. *arXiv preprint arXiv:2102.05329*, 2021.

[20] Wolfgang Helfrich. Elastic properties of lipid bilayers: theory and possible experiments. *Zeitschrift für Naturforschung C*, 28(11-12):693–703, 1973.

[21] Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. Tetrahedral meshing in the wild. *ACM Trans. Graph.*, 37(4):60:1–60:14, July 2018.

[22] Mahdieh Khanmohammadi, Rasmus Waagepetersen, and Jon Sporring. Analysis of shape and spatial interaction of synaptic vesicles using data from focused ion beam scanning electron microscopy (fib-sem). *Frontiers in neuroanatomy*, 9:116, 2015.

[23] Mahdieh Khanmohammadi, Rasmus P. Waagepetersen, and Jon Sporring. Analysis of shape and spatial interaction of synaptic vesicles using data from focused ion beam scanning electron microscopy (FIB-SEM). *Frontiers in Neuroanatomy*, 9(september), 2015.

[24] Matthew C Kiernan, Steve Vucic, Benjamin C Cheah, Martin R Turner, Andrew Eisen, Orla Hardiman, James R Burrell, and Margaret C Zoing. Amyotrophic lateral sclerosis. *The lancet*, 377(9769):942–955, 2011.

[25] JH Koenig and Kazuo Ikeda. Synaptic vesicles have two distinct recycling pathways. *The Journal of cell biology*, 135(3):797–808, 1996.

[26] Patrick Long, Guoqiang Wan, Michael T Roberts, and Gabriel Corfas. Myelin development, plasticity, and pathology in the auditory system. *Developmental neurobiology*, 78(2):80–92, 2018.

[27] Aurelien Lucchi, Yunpeng Li, Carlos Becker, and Pascal Fua. Electron microscopy dataset. `https://cvlab.epfl.ch/data/data-em/`. Accessed: 2020-03-14.

[28] Bohumil Maco, Anthony Holtmaat, Marco Cantoni, Anna Kreshuk, Christoph N Straehle, Fred A Hamprecht, and Graham W Knott. Correlative in vivo 2 photon and focused ion beam scanning electron microscopy of cortical neurons. *PloS one*, 8(2):e57405, 2013.

[29] Benoît Marin, Giancarlo Logroscino, Farid Boumédiene, Anaïs Labrunie, Philippe Couratier, Marie-Claude Babron, Anne Louise Leutenegger, Pierre Marie Preux, and Ettore Beghi. Clinical and demographic factors and outcome of amyotrophic lateral sclerosis in relation to population ancestral origin. *European journal of epidemiology*, 31(3):229–245, 2016.

[30] Joselene Marques, Harry K. Genant, Martin Lillholm, and Erik B. Dam. Diagnosis of osteoarthritis and prognosis of tibial cartilage loss by quantification of tibia trabecular bone from MRI. *Magnetic Resonance in Medicine*, 70(2):568–575, 2013.

[31] Nikolai Medvedev, Victor Popov, Christian Henneberger, Igor Kraev, Dmitri A Rusakov, and Michael G Stewart. Glia selectively approach synapses on thin dendritic spines. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1654):20140047, 2014.

[32] Angel Merchan-Perez, José-Rodrigo Rodriguez, Lidia AlonsoNanclares, Andreas Schertel, and Javier DeFelipe. Counting synapses using fib/sem microscopy: a true revolution for ultrastructural volume reconstruction. *Frontiers in neuroanatomy*, 3:18, 2009.

[33] Ling Miao, Udo Seifert, Michael Wortis, and Hans-Günther Döbereiner. Budding transitions of fluid-bilayer vesicles: the effect of area-difference elasticity. *Physical Review E*, 49(6):5389, 1994.

[34] Peter W. Michor, David Mumford, Jayant Shah, and Laurent Younes. A Metric on Shape Space with Explicit Geodesics. *arXiv:0706.4299 [math]*, May 2008. arXiv: 0706.4299.

[35] Juan Morales, Lidia Alonso-Nanclares, José-Rodrigo Rodríguez, Javier DeFelipe, Ángel Rodríguez, and Ángel Merchán-Pérez. Espina: a tool for the automated segmentation and counting of synapses in large stacks of electron microscopy images. *Frontiers in neuroanatomy*, 5:18, 2011.

[36] Andrei D Polyanin and Alexander V Manzhirov. *Handbook of mathematics for engineers and scientists*. CRC Press, 2006.

[37] DA Richards, C Guatimosim, and WJ Betz. Two endocytic recycling routes selectively fill two vesicle pools in frog motor nerve terminals. *Neuron*, 27(3):551–559, 2000.

[38] B. D. Ripley. The Second-Order Analysis of Stationary Point Processes. *Journal of Applied Probability*, 13(2):255–266, 1976. Publisher: Applied Probability Trust.

[39] B. D. Ripley. Tests of 'Randomness' for Spatial Point Patterns. *Journal of the Royal Statistical Society: Series B (Methodological)*, 41(3):368–374, 1979.

[40] B. D. Ripley. *Statistical inference for spatial processes*. Cambridge University Press, 1988.

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[42] Rolf Schneider and Wolfgang Weil. *Stochastic and Integral Geometry*. Springer, Heidelberg, 2008.

[43] Thermo Fisher Scientific. Amira-avizo software, 2020.

[44] Udo Seifert, Karin Berndl, and Reinhard Lipowsky. Shape transformations of vesicles: Phase diagram for spontaneous-curvature and bilayer-coupling models. *Physical review A*, 44(2):1182, 1991.

[45] Andrew W Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander WR Nelson, Alex Bridgland, et al. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792):706–710, 2020.

[46] James A Sethian. Fast marching methods. *SIAM review*, 41(2):199–235, 1999.

[47] Hang Si. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software (TOMS)*, 41(2):1–36, 2015.

[48] Jon Sporring, Rasmus Waagepetersen, and Stefan Sommer. Generalizations of Ripley's K-function with Application to Space Curves. In Albert C. S. Chung, James C. Gee, Paul A. Yushkevich, and Siqi Bao, editors, *Information Processing in Medical Imaging*, pages 731–742, Cham, 2019. Springer International Publishing.

[49] Hans Jacob Teglbjærg Stephensen, Sune Darkner, and Jon Sporring. Drift corrector software.

[50] Hans Jacob Teglbjærg Stephensen, Sune Darkner, and Jon Sporring. Restoring drifted electron microscope volumes using synaptic vesicles at sub-pixel accuracy. *Communications Biology*, 3(1):1–7, 2020.

[51] Hans Jacob Teglbjærg Stephensen and Jon Sporring. Rodent neuronal volume annotations and segmentations, 2020. https://www.doi.org/10.17894/ucph.33bd30d2-5796-48f4-a0a8-96fcc0ce6af5.

[52] Hans JT Stephensen. Biophysical parameter estimation using image analysis. Master's thesis, University of Copenhagen, 2017.

[53] Colin Studholme, Derek LG Hill, and David J Hawkes. An overlap invariant entropy measure of 3d medical image alignment. *Pattern recognition*, 32(1):71–86, 1999.

[54] Anne Marie Svane. Valuations in image analysis. In *Tensor Valuations and Their Applications in Stochastic Geometry and Imaging*, pages 435–454. Springer, 2017.

[55] Philippe Thevenaz, Urs E Ruttimann, and Michael Unser. A pyramid approach to subpixel registration based on intensity. *IEEE transactions on image processing*, 7(1):27–41, 1998.

[56] Hui-Hsin Tsai, Jianqin Niu, Roeben Munji, Dimitrios Davalos, Junlei Chang, Haijing Zhang, An-Chi Tien, Calvin J Kuo, Jonah R Chan, Richard Daneman, et al. Oligodendrocyte precursors migrate along vasculature in the developing nervous system. *Science*, 351(6271):379–384, 2016.

[57] DA Turner, IJ Anderson, JC Mason, and MG Cox. An algorithm for fitting an ellipsoid to data. *National Physical Laboratory, UK*, 1999.

[58] George E Uhlenbeck and Leonard S Ornstein. On the theory of the brownian motion. *Physical review*, 36(5):823, 1930.

[59] Marc Vaillant and Joan Glaunès. Surface matching via currents. In *Biennial International Conference on Information Processing in Medical Imaging*, pages 381–392. Springer, 2005.

[60] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[61] Yumei Wu, Christina Whiteus, C Shan Xu, Kenneth J Hayworth, Richard J Weinberg, Harald F Hess, and Pietro De Camilli. Contacts between the endoplasmic reticulum and other membranes in neurons. *Proceedings of the National Academy of Sciences*, 114(24):E4859–E4867, 2017.

[62] Dengsheng Zhang and Guojun Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1):1–19, 2004.

[63] Xuebao Zhang, Clement Y Chow, Zarife Sahenk, Michael E Shy, Miriam H Meisler, and Jun Li. Mutation of fig4 causes a rapidly progressive, asymmetric neuronal degeneration. *Brain*, 131(8):1990–2001, 2008.