Improving Super-Resolution Microscopy Data Reconstruction and Evaluation by Developing Advanced Processing Algorithms and Artificial Neuronal Networks

DISSERTATION ZUR ERLANGUNG DES
NATURWISSENSCHAFTLICHEN DOKTORGRADES
DER JULIUS-MAXIMILIANS-UNIVERSITÄT
WÜRZBURG

Vorgelegt von

Sebastian Reinhard

geboren in Aschaffenburg

Würzburg 2023

Eingereicht am: .....

Mitglieder der Promotionskommission:

Vorsitzender: .....
Gutachter: Prof. Dr. Markus Sauer
Gutachter: Prof. Dr. Philip Kollmannsberger

Tag des Promotionskolloquiums: .....

Doktorurkunde ausgehändigt am: .....

"Stand on the shoulders of those who came before you
and challenge yourself to go further"
To my beloved parents
Heide and Günther

Zusammenfassung

Für die Weiterentwicklung der Wissenschaft wird es immer wichtiger, Methoden aus verschiedenen Gebieten zu kombinieren. Die künstliche Intelligenz beruht beispielsweise auf dem Prinzip biologischer neuronaler Netze. Hier wird die Natur als Vorlage für unsere technische Entwicklung genutzt. Diese Innovationen können dazu eingesetzt werden, die verbliebenen Rätsel der Biologie zu lösen. Dazu gehören insbesondere Prozesse, die sich auf mikroskopischer Ebene abspielen und nur mit hochentwickelten Techniken untersucht werden können. Die *direkte* Stochastisch Optische Rekonstruktionsmikroskopie kombiniert Methoden der Chemie, Physik und Informatik, um biologische Prozesse auf molekularer Ebene sichtbar zu machen. Eine der Schlüsselkomponenten ist die computergestützte Rekonstruktion von hochaufgelösten Bildern. Die Verbesserung der zugrunde liegenden Algorithmen erhöht die Qualität der erzeugten Daten und ermöglicht weitere Einblicke in unsere Biologie. Es muss jedoch sichergestellt werden, dass die künstlich erstellten Bilder immer noch ein Abbild der Realität sind und nicht auf zufälligen Artefakten beruhen.
Expansionsmikroskopie vergrößert die Probe durch Einbettung in ein Hydrogel. Die Methode kann mit anderen hochauflösenden Techniken kombiniert werden, um die Auflösung noch weiter zu verbessern. Dieser Ansatz wurde an Mikrotubuli, einer bekannten filamentösen Referenzstruktur, verwendet, um verschiedene Protokolle und Markierungstechniken zu testen.
Mit `LineProfiler` wurde ein objektives Werkzeug zur Datenerfassung entwickelt. Anstatt Linienprofile in kleinen Bereichen zu erfassen, wertet die Software das gesamte Bild aus. Dies verbessert die Datenmenge und Datenqualität und verhindert eine voreingenommene Auswahl der ausgewerteten Regionen. Auf Grundlage der gesammelten Daten wurden theoretische Modelle für die erwartete Intensitätsverteilung über die Filamente erstellt. Daraus konnte geschlossen werden, dass die Markierung nach der Expansion den Markierungsfehler erheblich reduziert und somit die Qualität der Daten verbessert. Die Software wurde außerdem zur Bestimmung des Expansionsfaktors und der Anordnung der Daten des synaptonemalen Komplexes verwendet.
`Automated Simple Elastix` verwendet modernste Bildregistrierung, um Bilder vor und nach der Expansion zu vergleichen. Lineare Verzerrungen, die bei isotroper Expansion auftreten, werden korrigiert. Der strukturelle Expansionsfaktor wird berechnet und strukturelle Unstimmigkeiten werden in einer Verzerrungskarte hervorgehoben. Die Software wurde zur Bewertung expandierter Pilze und NK-Zellen eingesetzt. Dabei wurde festgestellt, dass der Expansionsfaktor für die beiden Strukturen unterschiedlich ist und unter der Gesamtexpansion des Hydrogels liegt.
Die Auswertung der Fluoreszenzlebensdauer von Emittern, die für die *direkte* Stochastische Optische Rekonstruktionsmikroskopie eingesetzt werden, kann zusätzliche Informationen über die molekulare Umgebung liefern oder Farbstoffe unterscheiden, die

eine ähnliche Lichtwellenlänge emittieren. Die entsprechenden Messungen erfordern eine konfokale Abtastung der Probe in Kombination mit dem fluoreszenten Schalten der zugrunde liegenden Emitter. Dies führt zu nichtlinearen, unterbrochenen Punktspreizfunktionen. Die Software `ReCSAI` löst dieses Problem, indem sie den klassischen Algorithmus des Compressed Sensing mit modernen Methoden der künstlichen Intelligenz kombiniert. Es wurden verschiedene Ansätze zur Kombination der Komponenten ausgewertet und festgestellt, dass die Integration von Compressed Sensing in die Netzwerkarchitektur die beste Performance in Bezug auf Rekonstruktionsgeschwindigkeit und -genauigkeit bringt. Neben einem tiefen Einblick in die Funktionsweise und das Lernen von künstlicher Intelligenz in Kombination mit klassischen Algorithmen konnten die beschriebenen Nichtlinearitäten mit einer deutlich verbesserten Auflösung im Vergleich zu anderen modernen Architekturen rekonstruiert werden.

Abstract

The fusion of methods from several disciplines is a crucial component of scientific development. Artificial Neural Networks, based on the principle of biological neuronal networks, demonstrate how nature provides the best templates for technological advancement. These innovations can then be employed to solve the remaining mysteries of biology, including, in particular, processes that take place on microscopic scales and can only be studied with sophisticated techniques. For instance, *direct* Stochastic Optical Reconstruction Microscopy combines tools from chemistry, physics, and computer science to visualize biological processes at the molecular level. One of the key components is the computer-aided reconstruction of super-resolved images. Improving the corresponding algorithms increases the quality of the generated data, providing further insights into our biology. It is important, however, to ensure that the heavily processed images are still a reflection of reality and do not originate in random artefacts.
Expansion microscopy is expanding the sample by embedding it in a swellable hydrogel. The method can be combined with other super-resolution techniques to gain additional resolution. We tested this approach on microtubules, a well-known filamentous reference structure, to evaluate the performance of different protocols and labelling techniques.
We developed `LineProfiler` an objective tool for data collection. Instead of collecting perpendicular profiles in small areas, the software gathers line profiles from filamentous structures of the entire image. This improves data quantity, quality and prevents a biased choice of the evaluated regions. On the basis of the collected data, we deployed theoretical models of the expected intensity distribution across the filaments. This led to the conclusion that post-expansion labelling significantly reduces the labelling error and thus, improves the data quality. The software was further used to determine the expansion factor and arrangement of synaptonemal complex data.
`Automated Simple Elastix` uses state-of-the-art image alignment to compare pre- and post-expansion images. It corrects linear distortions occurring under isotropic expansion, calculates a structural expansion factor and highlights structural mismatches in a distortion map. We used the software to evaluate expanded fungi and NK cells. We found that the expansion factor differs for the two structures and is lower than the overall expansion of the hydrogel.
Assessing the fluorescence lifetime of emitters used for *direct* Stochastic Optical Reconstruction Microscopy can reveal additional information about the molecular environment or distinguish dyes emitting with a similar wavelength. The corresponding measurements require a confocal scanning of the sample in combination with the fluorescent switching of the underlying emitters. This leads to non-linear, interrupted Point Spread Functions. The software `ReCSAI` targets this problem by combining the classical algorithm of compressed sensing with modern methods of artificial intelligence.

We evaluated several different approaches to combine these components and found, that unrolling compressed sensing into the network architecture yields the best performance in terms of reconstruction speed and accuracy. In addition to a deep insight into the functioning and learning of artificial intelligence in combination with classical algorithms, we were able to reconstruct the described non-linearities with significantly improved resolution, in comparison to other state-of-the-art architectures.

Contents

List of Figures

List of Tables

1 Introduction

Modern science is inconceivable without interdisciplinary research. Physics, chemistry, biology and computer science are closely linked and benefit from complementary findings [1][2]. While the research on the brain enabled us to develop the foundations of artificial intelligence, computational reconstructions and evaluations are an irreplaceable compartment of modern biology [3]. To gain insights into protein interactions, one has to resolve structures beyond the diffraction limit. Achieving this requires sophisticated microscopy techniques and reconstruction algorithms [4][5][6]. Fluorescent microscopy labels structures of interest with fluorescent dyes. These dyes are irradiated with an excitation laser and emit photons with a longer wavelength. Excitation and emission light can thus be separated, revealing specific information about the underlying structure. *direct* Stochastic Optical Reconstruction Microscopy (*d*STORM) [7] uses this technique to temporally separate emitters, by reversibly switching them into a non-fluorescent OFF state. Using a suitable algorithm, the stack of images with blurred points can be used to exactly determine the spatial position of the underlying emitters. With ten-thousands of images this allows the reconstruction of a super-resolved image [8][9]. Despite improving the resolution by one order of magnitude processes on the smallest scale still remain hidden. Reaching physical limits defined by the size of the used label or the number of collected photons, another solution has to be found. A recent promising approach is Expansion Microscopy [10]. Here, the sample is embedded into a hydrogel matrix, which is expanded by adding water. The underlying proteins expand with the hydrogel, increasing their distance, while the overall structure remains intact. Under the microscope, the increased sample size yields an increased resolution, proportional to the expansion factor of the gel. Thus, in combination with other super-resolution microscopy techniques like *d*STORM molecular resolution seems achievable [11]. However, the question remains, whether the resulting images still describe reality. Distortions can be introduced at several points. In the first place, already the choice of labelling can have an impact on the fluorescence image. The antibodies have a size and orientation that can not be neglected for resolutions in that order of magnitude. Furthermore, it must be ensured that the expansion process occurs isotropically and that only minor distortions are introduced during this processing step. Finally, the processing can lead to artefacts if the algorithm is not constructed for difficult or dense data.

Thus, the method has to be validated. This can be done by imaging well-known reference samples. If the molecular structure is validated by e.g. electron microscopy, it can be checked whether the processed images are valid. Significant evaluations require, however, adequate software. For this application, we developed `LineProfiler` [12]. `LineProfiler` is specialized in the evaluation of filamentous structures. Instead of locally measuring biased samples of those filaments, `LineProfiler` collects profiles

from filamentous structures all over the image. The software contains a graphical user interface and is easily portable. The latest executable, as well as the source code, can be found on GitHub under:
https://github.com/super-resolution/lineprofiler
To ensure isotropic expansion pre-expansion images can be compared with the images taken post-expansion. `Automated Simple Elastix` [13] is a script, that determines the expansion factor and detects and displays occurring distortions. The software is available under:
https://github.com/acecross/automated_elastix
While current reconstruction algorithms for super-resolution techniques like *d*STORM acquire precisions close to the theoretical limit for good signal-to-noise ratios and Point Spread Function (PSF)s, the reality is often more difficult. Confocal *d*STORM measurements yield non-linear PSFs because fluorescent emitters can switch into a non-fluorescent off-state during the scanning of the sample. The corresponding evaluation requires a high degree of flexibility in the fit function which can not be provided by classical algorithms. Artificial Neural Networks (ANN)s on the other hand are perfectly suited to compensate for these non-linearities since they can cover a variety of inputs by a huge amount of trainable variables. `ReCSAI`
https://github.com/super-resolution/ReCSAI is a package for the fitting of non-linear PSFs with high accuracy. It provides an accurate data simulation as well as several state-of-the-art network architectures.
To grasp the concepts of the described projects it is beneficial to first look at the underlying systematic. Chapter 2 and 3 of this thesis introduce the concepts of super-resolution microscopy and signal processing respectively. A majority of the corresponding figures are rendered with python code which can be found on
https://github.com/acecross/ai-and-image-processing. The code is published under MIT license and can be reused to render similar figures or gain a deeper insight into the underlying systematic.
The results part of this thesis (cf. chapter 4) incorporates state-of-the-art algorithms into the biological evaluation workflow.

2 Theory of super-resolution microscopy

The first question that comes to mind when talking about super-resolution microscopy is: What is resolution in the first place? Abbe [14] and Rayleigh [15] described resolution in the context of light microscopy as the capability to discern two close light points from each other. In classical optics, this metric is coupled to the Full Width at Half Maximum (FWHM) of the PSF, the distribution of photons over the detector. With the rise of techniques surpassing that limit, the resolution depends on a large set of parameters. Some of them, like photon count and signal-to-noise ratios, are commonly

respected and can be clearly quantified [16], but others like the distance of the emitter to its target structure are often neglected [11]. So how far can we possibly go?
This chapter will provide an introduction to modern super-resolution microscopy. It will start out with the foundation for most techniques: The principle of fluorescent dyes and their application in biological samples. State-of-the-art methods surpassing the resolution limit will be introduced. The chapter will be concluded with a recent breakthrough: Expansion microscopy, a technique inverting the problem by enlarging the sample instead of enhancing the resolution.

2.1 Why classical light microscopy is limited

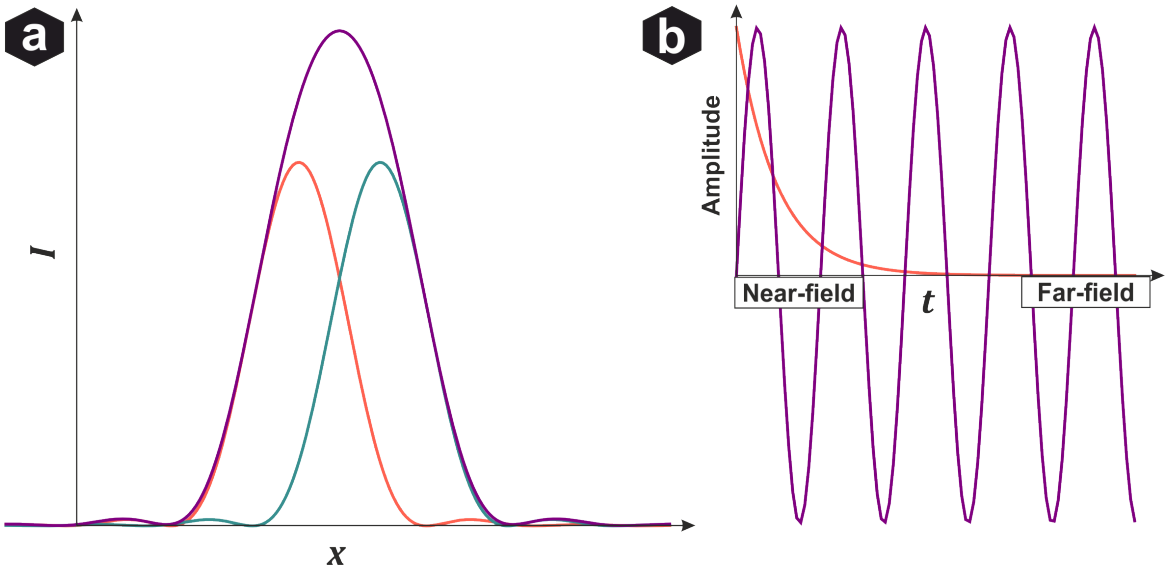

Figure 1: **Resolution limit** a) Rayleigh criterion: if the maximum of the PSF (orange, cyan) of one emitter overlap for more than the first minimum of the other, they can not be distinguished anymore and appear as one emitter (purple). b) Propagation of information into the far field. If $k_x^2 + k_y^2 > |\boldsymbol{k}|^2$ the information decays exponentially with the distance from the sample (orange) and can not be detected anymore. Smaller k_x and k_y propagate as sine waves (purple) and are, therefore, detectable in the far field.

Ernst Abbe quantifies the optical resolution as "die physikalische Unterscheidungsgrenze dagegen hängt allein vom Oeffnungswinkel ab und ist dem Sinus seines halben Betrages proportional" [14]. Freely translated this means that the minimum distance d between two emitters required to distinguish them is:

$$d = \frac{\lambda}{2n \sin \alpha}, \tag{1}$$

where λ is the wavelength of the light, α denotes the half of the aperture angle and n the refraction index of the medium. Rayleigh [15] describes the problem similarly with a slightly different, but more visually appealing explanation: Two emitters can

be separated until the maximum of the diffraction pattern (PSF) of the first emitter crosses the first minimum of the PSF of the second emitter (cf. Figure 1 a). If the emitters are closer together, the overlap of their central maxima is larger than two times half of the maximum, so they appear as one emitter with a higher maximum. This makes the two original functions much harder to distinguish. For conventional light microscopy with an excitation wavelength of 400 nm this leads to a diffraction limit of ~ 200 nm. But why are photons from point emitters distributed over the PSF in the first place?
For this concept, it is beneficial to first grasp the concept of the Fourier space, described in section 3.2.1.

High resolution resides on high frequencies since fast changes add detail to the signal. Performing a Fourier decomposition these frequencies get decomposed into spatial positions. In Fourier space an electromagnetic wave can be described by the wavevector $\mathbf{k}$ for which holds:

$$|\boldsymbol{k}|^2 = k_x^2 + k_y^2 + k_z^2. \tag{2}$$

The Optical Transfer Function H describes the propagation of light in the system by $H \propto e^{ik_z}$. Therefore, large k_x and k_y components decay exponentially if their sum exceeds the limited magnitude of:

$$|\boldsymbol{k}| = \frac{2\pi NA}{\lambda}. \tag{3}$$

In other words: For $|\boldsymbol{k}|^2 < k_x^2 + k_y^2$ the k_z component gets imaginary leading to a negative argument in the exponent of the propagator. The amplitudes of the larger k_x and k_y wave components are therefore lost in the far field [17]. These missing details let the point source appear blurry under a microscope. The two cases are shown in Figure 1 b): for small k_x and k_y components the information propagates as a wave whereas the propagated information decays exponentially if k_z gets imaginary.

2.2 Fluorescence microscopy

A crucial step in surpassing the resolution limit was the discovery of organic dyes, able to emit light in the visible spectrum. Since it was discovered that these dyes can be attached to biological structures or proteins, fluorescence microscopy has become an important tool in biological and biomedical imaging [18]. This method remains diffraction limited, but it exclusively illuminates structures of interest with high contrast and is therefore able to deliver important insights into cell-protein interactions. To help understand the underlying physical processes of fluorescence, we will give a shallow introduction to the concepts of quantum mechanics.
According to the Pauli principle [19, chapter 5.1] all electrons of an atom must differ in at least one quantum number. This results in atomic shells (referring to the first quan-

tum number) with different shapes (second quantum number) and orientation (third quantum number) being filled up with alternating spin up ($\frac{1}{2}$) and spin down ($-\frac{1}{2}$) electrons. If energetically sufficient, two or more atoms add up to a molecule, sharing the electrons of their outmost shells.

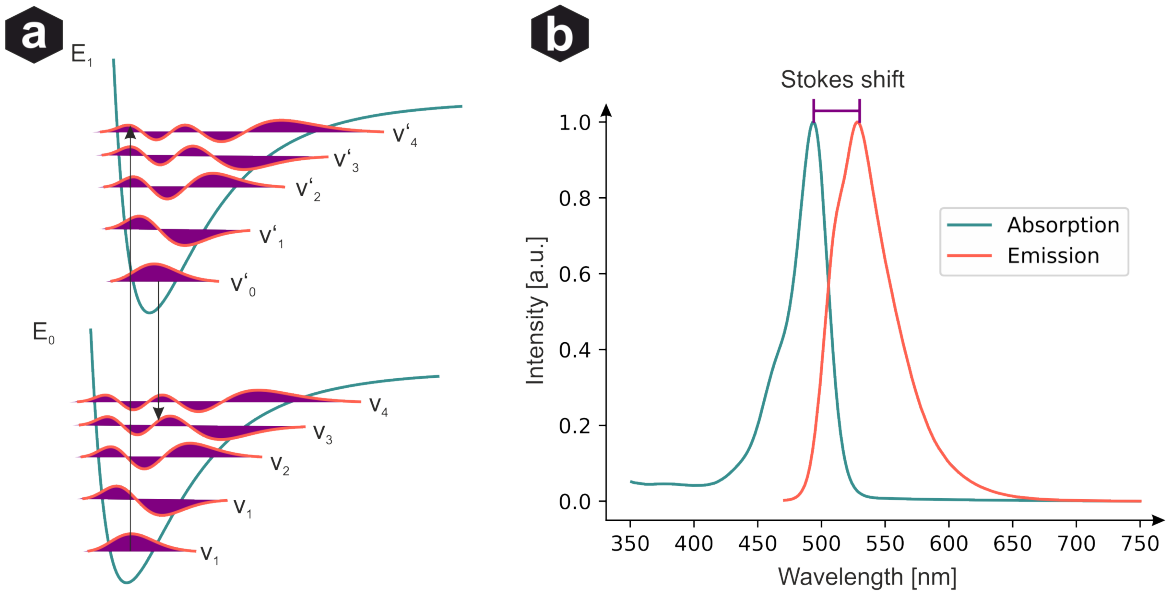

Figure 2: **Frank Condon principle** a) Energetic ground E_0 and excited E_1 state with its vibronical levels v, v'. The transition probabilities are given by the overlap of the wave functions (magenta). While the absorption generally excites from the v_0 level of E_0 to v'_n of the excited state (E_1), the electron rapidly decays non-radiative to the v'_0 level before dropping to the v_n level of the ground state (E_0) under emission of a photon. Absorption and emission energies are mirrored if the potential wells referring to the vibronical levels v and v' of ground and excited state, respectively, are identical. b) Emission (red) and absorption (cyan) of fluorescein in borax (measured by Dominic Helmerich.

Molecules can be excited by absorbing photons. If a photon with the right amount of energy hits a molecule, one electron can rise to an energetically higher state. Dropping back to the ground state, a photon with equal or higher wavelength, i.e. lower energy, is re-emitted. The dynamic of these electronic transitions is defined by the Frank Condon Principle [20] (Figure 2). The basic assumption is that these electron transitions happen too fast (10^{-15} s) to be influenced by nuclei movements (10^{-13} s). Therefore, the decisive factors are the vibronical levels (v_n, v'_n) occurring due to vibrations of the molecule's underlying atoms relative to each other. The according wave functions of the ground and excited state yield the probability p of a transition. Visually, the probability equals the square of the overlap area under the two wave functions Ψ_1, Ψ_2. Mathematically, it is described by their squared integral:

$$p = |\Psi_1 \cdot \mu \cdot \Psi_2|^2, \tag{4}$$

where μ denotes the dipole operator, which is determined by the location and charge of the electrons. Following the Born-Oppenheimer approximation [21], the component contributed by the nuclei is neglected due to the previously mentioned orders of magnitude regarding the dynamics. The wavefunctions of a molecule can be approximated by solving the stationary Schrödinger equation for a particle of mass m at location r in a potential $U(r)$ [22]:

$$\frac{\hbar^2}{2m}\Delta\Psi(r) + (E - U(r))\Psi(r) = 0 \tag{5}$$

for its Eigenvectors. The Eigenvalues are the possible energy levels E.
The excitation and the emission process according to the Frank Condon principle [20] is shown in Figure 2 a). An electron in the ground state gets excited to a state of a higher vibronic level v'_n. Against common intuition the excitation from v_0 to v'_0 is not the one with the highest probability, as can be seen in Figure 2 a). Taking a closer look, one can see that the potential wells of E_0 and E_1 are shifted. This shift arises due to different optimal distances between the nuclei depending on the state. It changes the overlap of the wavefunctions and therefore the transition probability. From the v'_n vibronic level the excited electron rapidly relaxes non-radiative to the lowest vibronic level v'_0. The energy is transferred to the thermal energy of the molecule. During emission, the electron decays from v'_0 of the E_1 state to any v_n of the ground state E_0, leading to less released energy compared to the energy needed for absorption. This leads to the shift of absorption and emission spectra to longer wavelengths visible in Figure 2 b). The phenomenon is called Stokes shift [23, chapter 1.9] and is common for most fluorescent dyes. From the vibronic excited v_n level the electron also relaxes rapidly to the lowest vibronic level v_0 of the ground state E_0. If the potential wells of ground and excited state are similar, the transitions of $v_0 \rightarrow v'_n$ and $v'_0 \rightarrow v_n$ have similar probabilities. Therefore, absorption and emission spectra are approximately mirrored.
Understanding the fundamentals of the energy transitions in a molecule, we can take a look at the possible excited states (Figure 3). The luminescence of molecules is mainly composed of two state transitions: fluorescence and phosphorescence. Fluorescence is the transition between the singlet excited state S_1 and the singlet ground state S_0. In the singlet state, the spins of electrons in the molecule add up to zero. The excitet state has a high overlap with the ground state. Lifetimes are, therefore, relatively short ($\sim 1\,\text{ns}$) and transitions have a high probability. The fluorescent emission stands in concurrence with Internal Conversion (IC). Internal Conversion denotes the process of an electron transferring to a high vibronic level of an energetic lower state. From here, it relaxes non-radiative into the ground state. The proportion of absorbed photons N_A to radiative emission N_E of a molecule is called the quantum yield:

$$Q_E = \frac{N_E}{N_A}. \tag{6}$$

The Q_E of a dye is an important quantity since its lifetime is limited and the number of emitted photons directly correlates with the quality of the image.
Transitions with higher singlet state S_n and S_0 play only a minor role. $S_{n>1}$ states have a high overlap of higher vibronical levels of S_1. Internal Conversion is therefore the dominant process leading to Kasha's rule [24], that all higher excited states quickly relax non-radiative to the S_1, v_0' level. The fluorescent emission spectrum is therefore independent of excitation wavelength. Phosphorescence occurs during the transition from triplet-state T_1 to ground state. A triplet-state has a resulting spin of one. Excitation to the triplet state happens over an excitation to the singlet state and a subsequent spin flip. Singlet and triplet states share a very small overlap, that only exists due to spin-orbit coupling. Electrons, therefore, have a very small probability to transfer to a triplet-state via Inter System Crossing (ISC). The lifetime of the triplet-state is in comparison very long (up to seconds). However, the energetic level of the triplet state is below the excited singlet state. Thus, emitted photons have a different wavelength, often in the infrared regime, and can be easily distinguished from fluorescent photons. State transitions are often depicted in a Jablonski diagram as shown in Figure 3.

2.2.1 Microscopy modalities

Widefield fluorescence microscopes use a laser of a specific wavelength to excite a large sample volume. The beam is projected onto the sample with an objective. The fluorescent light can be collected with the same lens. Excitation and emission are separated with a dichromatic mirror. The signal is detected with a multipixel detector camera (cf. Section 2.3). The excited volume can be reduced by operating the widefield microscope in Total Internal Reflection Fluorescence (TIRF) mode [25]. Figure 4 a) shows schematically the corresponding beam path. As can be seen, the light does not penetrate into the sample but gets completely reflected at the coverslip. This results in an evanescent wave with exponentially decaying intensity in the sample. Therefore, only fluorophores close to the coverslip are excited.
Confocal microscopes [17, chapter 5.1] as shown in Figure 4 b) use a laser beam which is focused on the sample for excitation. The beam is spatially filtered by a pinhole to generate a Gaussian beam profile. The fluorescence is collected by the same objective lens and separated from the excitation by a dichroic mirror. Here, the Stokes shift is fundamentally important, to separate emission and excitation through the difference in wavelength. The fluorescence is focused on a second pinhole in front of the detector. The pinhole filters light from different focal planes. The technique can therefore be used to realise 3D resolution and enhance contrast. Filtering the fluorescent signal by a pinhole increases the lateral resolution by a factor of 1.3 by narrowing the FWHM of the PSF. To record a full image, the confocal setup has to scan the sample, as one position only collects information for one pixel.

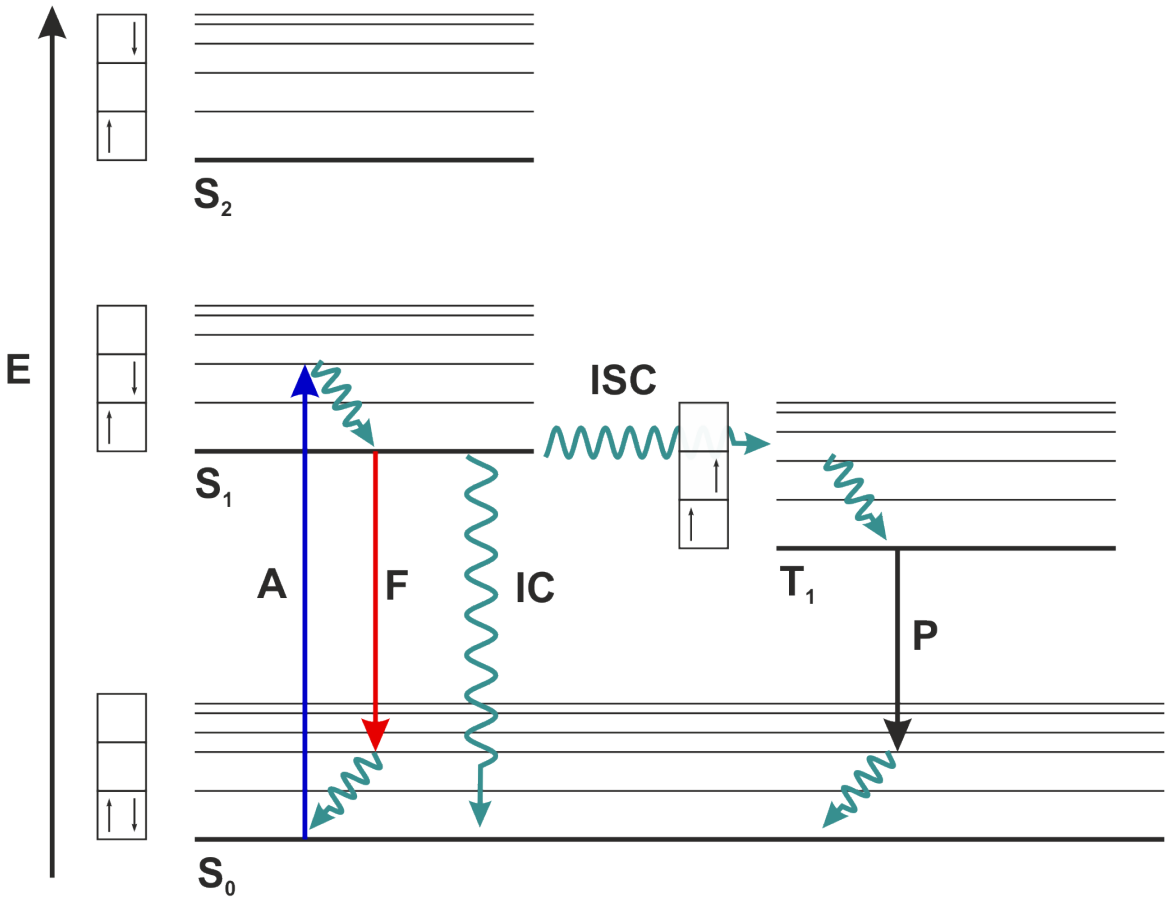

Figure 3: **Jablonski diagram** Electrons in the singlet ground-state S_0 can be excited into the S_1 state by absorbing a photon. The electron rapidly decays to the lowest vibronic level of the S_1 state and can relax to the ground state via Fluorescence (F) or Internal Conversion (IC). There also exists a small probability to enter a long-lived triplet state T_1 by Inter-System Crossing (ISC). From there the electron can relax under phosphorescence (P) into the ground state.

Since only the fluorescence is detected, the triplet state can be used to switch an emitter into an OFF state. As will be shown in Section 2.4, this feature can be used to gain additional information about the spatial location of fluorophores.

2.2.2 Fluorescence-Lifetime Imaging Microscopy

Even though the fluorescence lifetimes of various singlet-to-singlet transitions are on the same order of magnitude, there are still differences. Measuring these differences can reveal further insights into the biological context. Two dyes with highly overlapping emission spectra can not be distinguished by their wavelength, but they can be properly identified by their lifetime. It is even possible to deduct information about adjacent molecules. The corresponding method, Fluorescence-Lifetime Imaging Microscopy (FLIM) [26], measures the decay time of the fluorescent singlet-singlet transition. The concept can be divided into two different approaches. Figure 5 shows the excitation and signal principles of both methods.

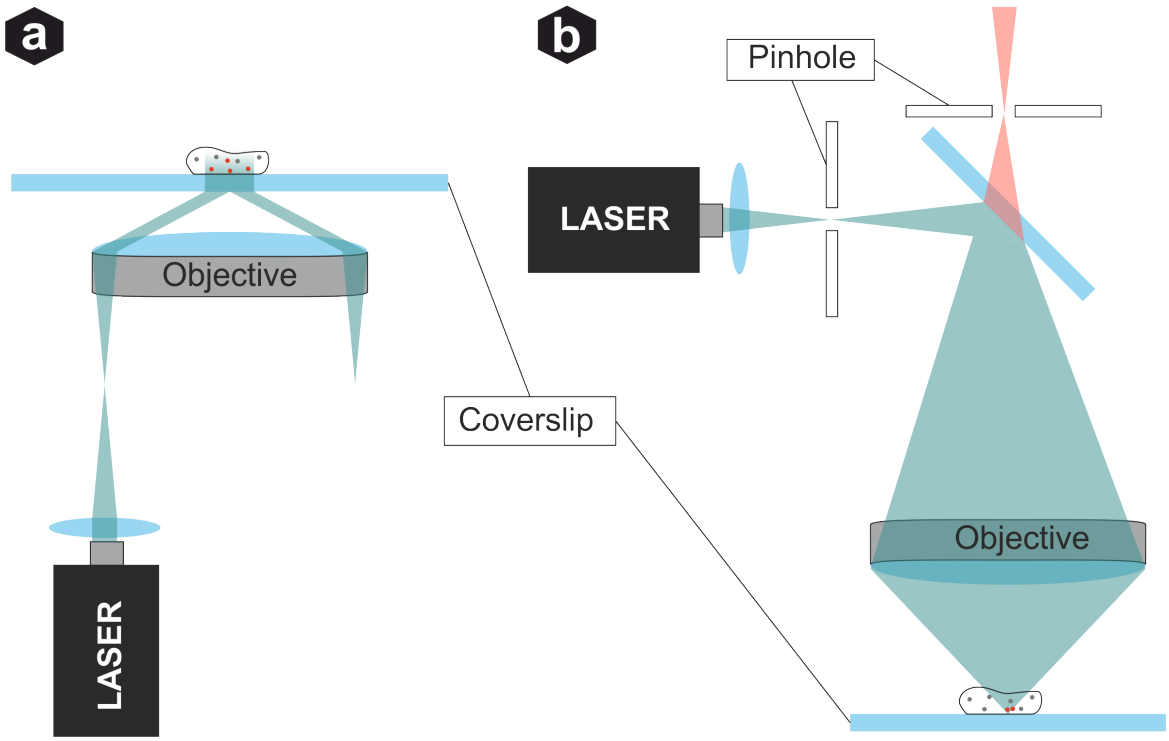

Figure 4: **Microscopy modalities** a) TIRF: The excitation beam is completely reflected at the coverslip. This results in an evanescent wave that only excites emitters very close to the coverslip. b) Confocal microscopy: Excitation and emission are filtered by a pinhole. Out-of-focus signals are significantly reduced.

The time-domain method [23, chapter 4] illuminates the sample with a very short laser pulse (5 a). Electrons are lifted into an excited state with an average lifetime τ. Note here, that the fluorescent emission is a statistical process, so the number of emitted photons decreases over time. The resulting exponential decay of the intensity I at time t is expressed by

$$I(t) = I_0 e^{-\frac{t}{\tau}}, \tag{7}$$

where the intensity at the start is I_0. Fitting this formula to the histogram of arriving photons, yields the lifetime of the emitter. A common method to measure this said histogram is by Time-Correlated Single-Photon Counting (TCSPC). Here, the conditions are designed in such a way, that less than one photon per laser pulse is detected. The photon arrives at the detector with a delay time that is used for the corresponding histogram. Exciting multiple molecules per pulse would lead to a shift to lower lifetimes since suitable detectors like the Avalanche Photo Diode (APD) have downtimes of several nanoseconds. A second photon would fall with a high probability into the downtime and would, therefore, not be collected.

The frequency-domain method [23, chapter 5] illuminates the sample with a laser beam of varying intensity (Figure 5 b). A common modulation is a sine wave with a frequency around 100 MHz. The intensity modulation which is in the order of magnitude of the

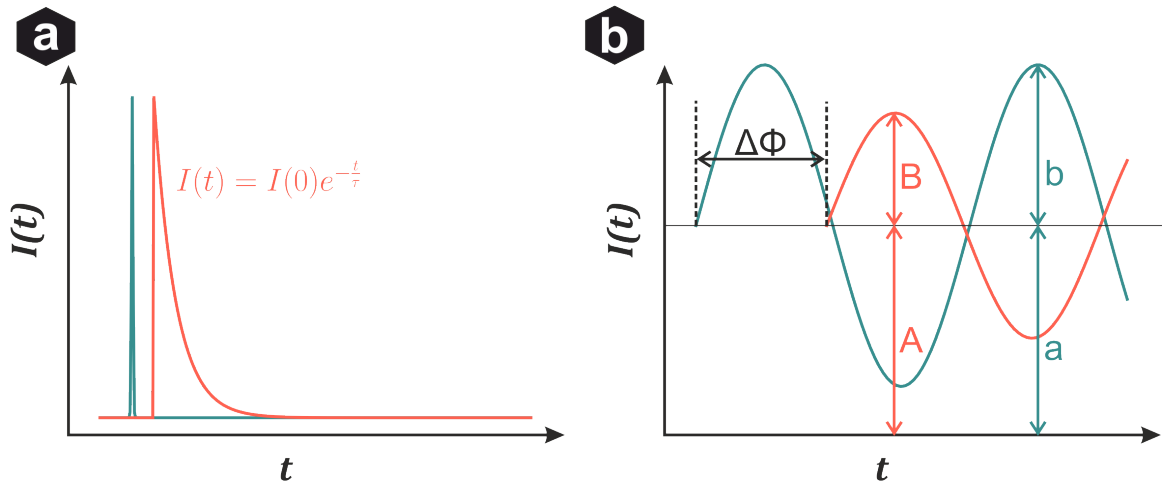

Figure 5: **Fluorescence-Lifetime Imaging Microscopy** a) Time-domain FLIM: The sample is excited by a short laser pulse (cyan). The lifetime can be calculated over the exponential decay of the signal intensity (orange). b) Frequency-domain FLIM: The excitation intensity (cyan) is modulated with a frequency close to the fluorescence lifetime. The lifetime can be calculated over the amplitudes $A, B,\ a, b$ and the phase shift $\Delta\phi$ of the signal intensity (orange).

lifetime also changes the emission over time. Plotting the excitation pulse together with the emission signal yields a phase shift, which can be used to calculate the lifetimes:

$$\tau_\phi = \omega^{-1} \tan(\Delta\phi) \tag{8}$$

$$\tau_m = \omega^{-1} \left(\frac{1}{m^2} - 1 \right)^{\frac{1}{2}}. \tag{9}$$

m denotes a factor that can be calculated over the excitation and emission signal strength by $m = \frac{aB}{Ab}$ as shown in Figure 5 b), ω denotes the modulation frequency in radians. τ_ϕ and τ_m are independently calculated from the phase angle ϕ and the modulation m. They yield differently weighted averages of the decay time and are only equal if they consist of one exponential.

2.3 Noise sources in microscopy

As will be explained in section 2.4.2 the localisation precision in Single Molecule Localisation Microscopy (SMLM) strongly depends on the noise level of the image. Elastic Rayleigh scattering [27] has the same wavelength as the excitation beam and can be filtered from the emission by using a dichroic mirror. Inelastic Raman scattering [28] is problematic, but mainly depends on the illuminated volume. Keeping this volume low reduces the scattering to an acceptable level. Hence, next to various distortions that can occur in the sample, like unspecific labelling or autofluorescence, noise is mainly defined by the camera [29]. Most photon detectors have the following relevant noise sources in common.

- **Dark noise** is the current that flows in the detector in the absence of photons. It occurs due to the semiconducting properties of the photo-diode. Depending on the temperature, electrons have a certain probability to jump from the valence band into the conduction band. The amount of noise is defined by the number of events D_s that happen during the exposure time t. The counted photons S contain the true number of photons S_0 and the noise:

$$S = S_0 + D_s * t. \tag{10}$$

- **Photon shot noise** is characterized by the light source. Since the number of emitted photons is discrete, the illumination of the sensor can not be assumed as continuous. This results in a (noise) statistic. The number of arriving photons S converted into electrons with a detection rate E_Q follow a Poisson statistic [30]. Thus, the probability of a photon under the condition of detection is described as:

$$P(S|E_Q) = \frac{E_Q^S e^{E_Q}}{S!}. \tag{11}$$

- **Amplification noise** describes the probability of the input electrons S_i to produce S_o output electrons in the amplification register with a gain factor g. The process is described by the Gamma distribution:

$$P(S_o|S_i) = S_o^{S_i-1} \frac{e^{-S_o/g}}{\Gamma(S_i) g^{S_i}}, \tag{12}$$

  where $\Gamma(S_i) = (S_i - 1)!$ denotes the Gamma function.

- **Readout noise** occurs during the conversion of electrons into an electronic signal and is normally distributed. This normal distribution N of the electric current I with mean I_0 and standard deviation σ_{read} follows

$$N(I|I_0, \sigma_{\text{read}}) = \frac{1}{\sqrt{2\pi}\sigma_{\text{read}}} e^{-\frac{(I-I_0)^2}{2\sigma_{\text{read}}^2}}. \tag{13}$$

To predict and simulate an accurate model for SMLM data, it is, therefore, crucial to understand the principles of common camera models and identify noise sources. One of the most popular camera models is the Electron-Multiplying Charge-Coupled Device (EMCCD). The EMCCD acquisition process, as shown in Figure 6 a), can be divided in three steps. Each one adds specific noise to the image. The detection step collects photons and converts them to electrons. Electrons are either stored in a detection array before being passed into the readout register or directly passed on. The row of pixels is then serially shifted to the multiplying register amplifying the signal by a factor g. This step increases the signal in comparison to the readout noise σ_{read} and the offset

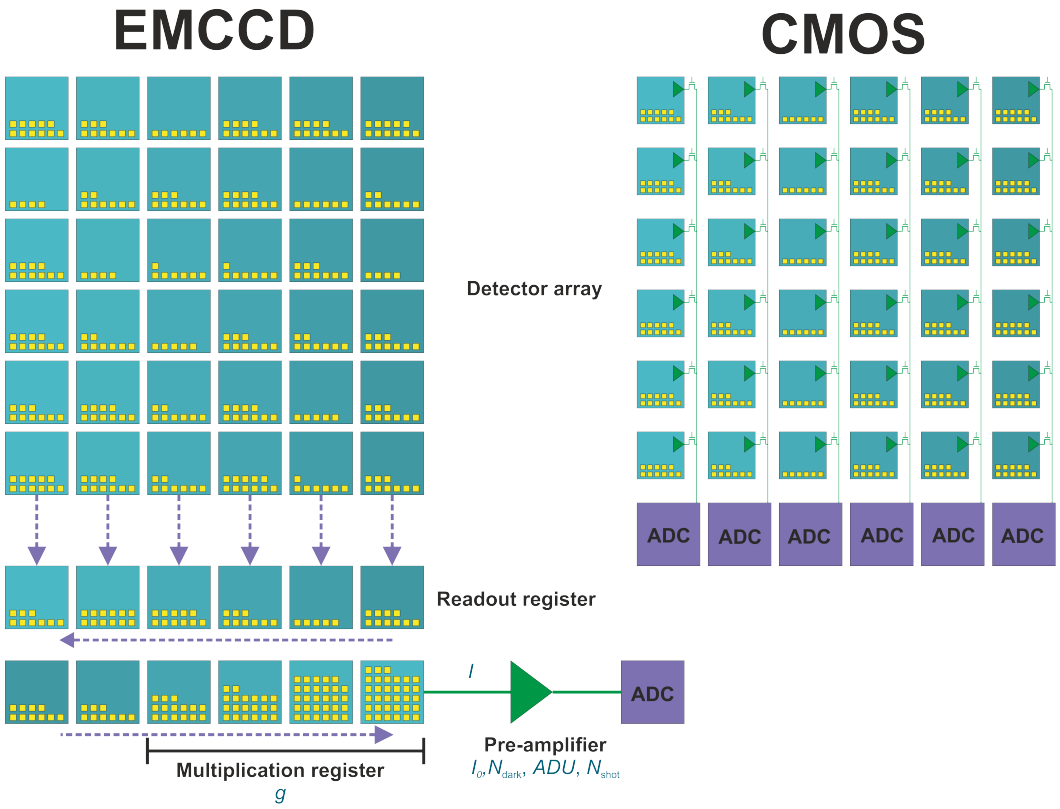

Figure 6: **EMCCD and CMOS photon detector.** The EMCCD chip collects electrons in a photo-diode during the exposure time. A typical pixel can store up to 2^{14} electrons, resulting in a true range of 14 bits. During the read-out process, electrons are shifted row-wise into the readout register. The next step involves a pixel-wise shift through a multiplication register where electrons are multiplied in each step with a gain factor g. The signal is converted into the current in a pre-amplifier before it is converted into digital units in the Analog Digital Converter (ADC). The CMOS chip focuses photons on the photodiode with a microlens, since the pre-amplifier is built on the chip, decreasing the photosensitive region. Here, the electrons are directly converted into current and pre-amplified. An ADC manages the readout process which can be controlled pixel-wise.

I_0 at the amplifier, allowing measurements with high sensitivity [31].

The Complementary Metal-Oxide-Semiconductor (CMOS) sensor has a transistor which performs the electron-to-voltage conversion and amplification, for each pixel. The transistor is placed next to the photo-diode, reducing the active region of each pixel. Light is typically focused on the photoactive zone using a microlens. Variations in the individual transistors and amplifiers introduce the so-called **Fixed-pattern Noise**. This noise describes the pixel-dependent signal response for the same number of impinging photons [32]. In comparison to EMCCD, CMOS cameras are cheaper to produce and have no blooming, i.e. the overflowing of electrons to the next pixel.

2.4 Breaking the resolution limit

"For the development of super-resolved fluorescence microscopy" the Nobel prize in chemistry 2014 was awarded to E. Betzig, S. W. Hell and W. E. Moerner [33]. They developed methods that surpass the diffraction limit by determining the position of the underlying emitters in fluorescence microscopy. Betzig's method, Photoactivated Localisation Microscopy (PALM) [34], works by activating and bleaching fluorescent proteins over time, resulting in less than one active emitter per diffraction-limited area. Samples are imaged with TIRF microscopy. The signal is collected on an EMCCD chip. The initial frames consist of a sparse subset of spatially separated emitters since the majority of the fluorescent proteins are in an inactive OFF state. As the laser destroys the chromophores, they bleach out after some time. Thus, the sample is illuminated with a second laser of a different wavelength, switching new emitters into the fluorescent ON state. This process is repeated until all fluorescent molecules are bleached. All images captured during this process are saved as a stack of frames. The centre of the time and spatial separated PSFs can be estimated by a statistical fit. With thousands of frames collected over time, a super-resolved image can be reconstructed. This principle is shown in Figure 7. It is the basis for a whole class of microscopy methods, the so-called SMLM.

Hell's approach, Stimulated Emission Depletion (STED) [35], uses a confocal microscopy setup with a two-laser system. While one laser performs a common excitation of fluorophores, the other one induces stimulated emission to induce a decay of the singlet state with a lower wavelength. This laser is called a depletion laser. Using a set of phase modulation plates, the beam of the depletion laser is modulated to a hollow focus, a ring-like intensity distribution with a minimum in the centre. As a result, fluorescence only happens in the centre of the excitation beam, which is not limited by resolution, but by the intensity of the depletion laser. Rastering over the sample with the two laser system allows reconstruction with enhanced resolution.

Another method worth mentioning is Structured Illumination Microscopy (SIM) [36]. Here, a low-frequency stripe pattern is used for the excitation of the sample. By changing the direction and phase of the pattern, additional high-frequency information about the underlying structure is gained and an image with approximately doubled resolution (for linear excitation patterns) can be reconstructed.

Dertinger et al. developed Super-resolution Optical Fluctuation Microscopy (SOFI) [37]. SOFI uses the temporal fluctuations of emitters to increase resolution.

SMLM was developed further by the introduction of Stochastic Optical Reconstruction Microscopy (STORM) [38]. STORM uses a Cy5-Cy3 cyanine dye pair to switch emitters between a fluorescent ON and a non-fluorescent OFF-state. A red laser is used to excite the Cy5 emitter and turn it into the OFF state. A green laser is used to control the recovery rate into the ON state.

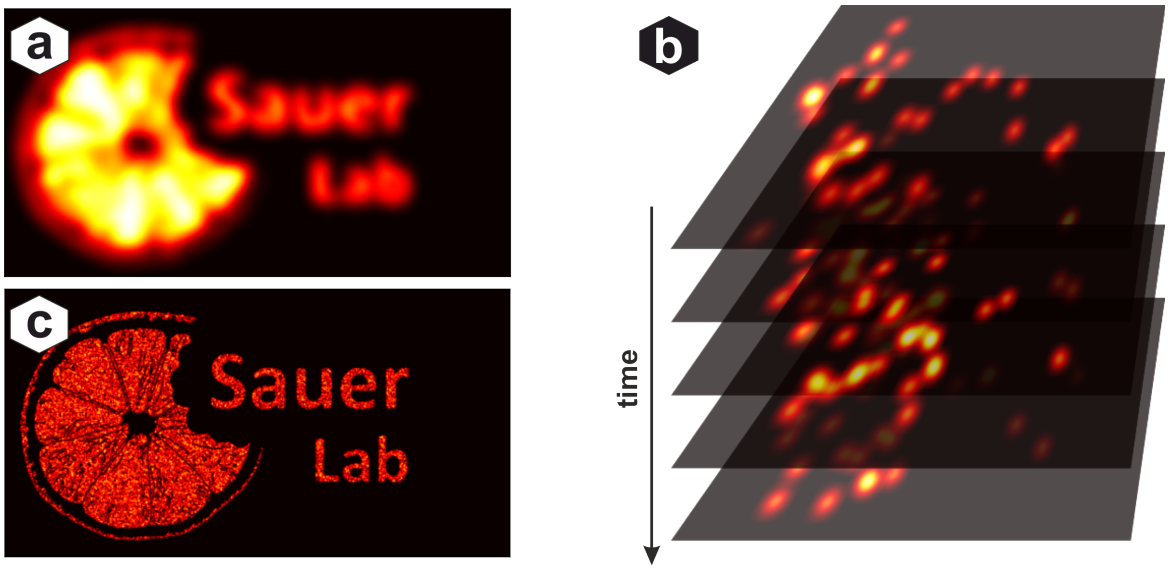

Figure 7: **Principle of SMLM** a) Resolution limited image b) Temporal separation of emitters allows determining their position with high precision. c) Rendering the position of all emitters results in the reconstruction of a highly resolved image.

Heilemann et al. [7] showed that this photoswitching between the ON and the OFF state is also possible without the use of an activator fluorophore. Due to the direct switching of the emitters, their method is called *d*STORM. The advantage of both methods is the reversible switching of fluorophores, which leads to a higher photon number per emitter and thus to a higher localisation precision. In [39] the nuclear pore complex was imaged with a resolution of $\sim 15\,\text{nm}$ with this method.
DeoxyriboNucleic Acid (DNA)-Paint (Points accumulation for imaging in nanoscale topography) uses a solvent including emitters attached to single DNA strands (imager strands). Unbound imager strands flow freely in the solvent and do not emit in discrete spots. Additionally, using a TIRF setup only a small subset is excited. Unbound emitters are, therefore, considered to be in an OFF state. The target is labelled with the complementary single DNA strand, the docking strand. By docking on the target, the imager strands get unfolded and emit continuously from the same position. The density of active emitters depends on the number of strands within the solvent and can therefore be easily adjusted to the required rates. Another advantage is the lag of photo-bleaching since emitters are switched in every ON-OFF cycle [40].
To understand how localisation precision and reconstruction quality are limited, it is crucial to have a fundamental understanding of the PSF and perturbations occurring during the acquisition process.

2.4.1 The Point Spread Function

As examined in section 2.1, even under optimal conditions, photons of a point-shaped emitter do not arrive as a sharp peak in the detector. The PSF is the distribution of photons from an ideal point source by an imaging system. This process can mathemat-

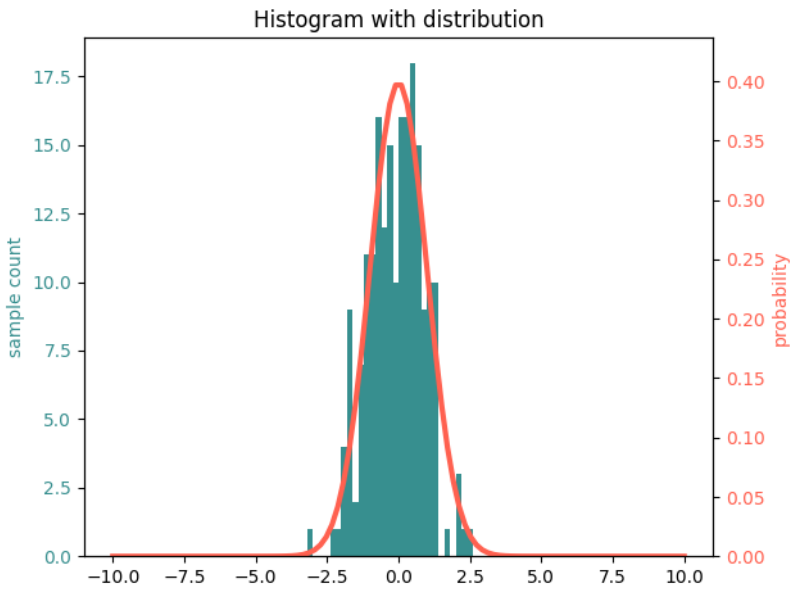

Figure 8: One-dimensional Gaussian photon distribution (orange) and randomly sampled photons for $N = 200$ counts (cyan).

ically be described as the 2D convolution of the "true" image $B^{K,L}$ which has the shape of KxL pixel, and the PSF P. One pixel of the measured image I reads:

$$I_{n,m} = \sum_{k=k-K/2,\, l=l-L/2}^{K/2,\, L/2} P_{k,l} B_{n+k,\, m+l} + s, \tag{14}$$

where n, m denotes the coordinates of the pixel of the measured image and s is the occurring noise.
The simplest estimation for a PSF is a Gaussian distribution:

$$\mathrm{PSF}_{\mathrm{gauss1D}}(x) = \frac{N}{\sigma_0\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma_0^2}, \tag{15}$$

where μ denotes the center of the emitter, σ_0 the standard deviation and N the number of emitted photons. Figure 8 shows such a distribution and possible underlying detection of photons. Expanding the concept to multiple dimensions the multivariate normal distribution is given by:

$$\mathrm{PSF}_{\mathrm{gauss2D}}(x) = \frac{N}{|\Sigma|^{\frac{1}{2}} 2\pi^{\frac{d}{2}}} e^{-\left((\mathbf{x}-\boldsymbol{\mu})^T \Sigma (\mathbf{x}-\boldsymbol{\mu})\right)}, \tag{16}$$

where Σ denotes the covariance matrix, the bold letters are the corresponding vectors and d denotes the dimension. Despite being inaccurate, this approximation performs pretty well for reconstruction purposes since its mean value $\boldsymbol{\mu}$ is the centre of mass of the distributed values [9] [8]. A plot of the distribution is shown in Figure 9 a. More

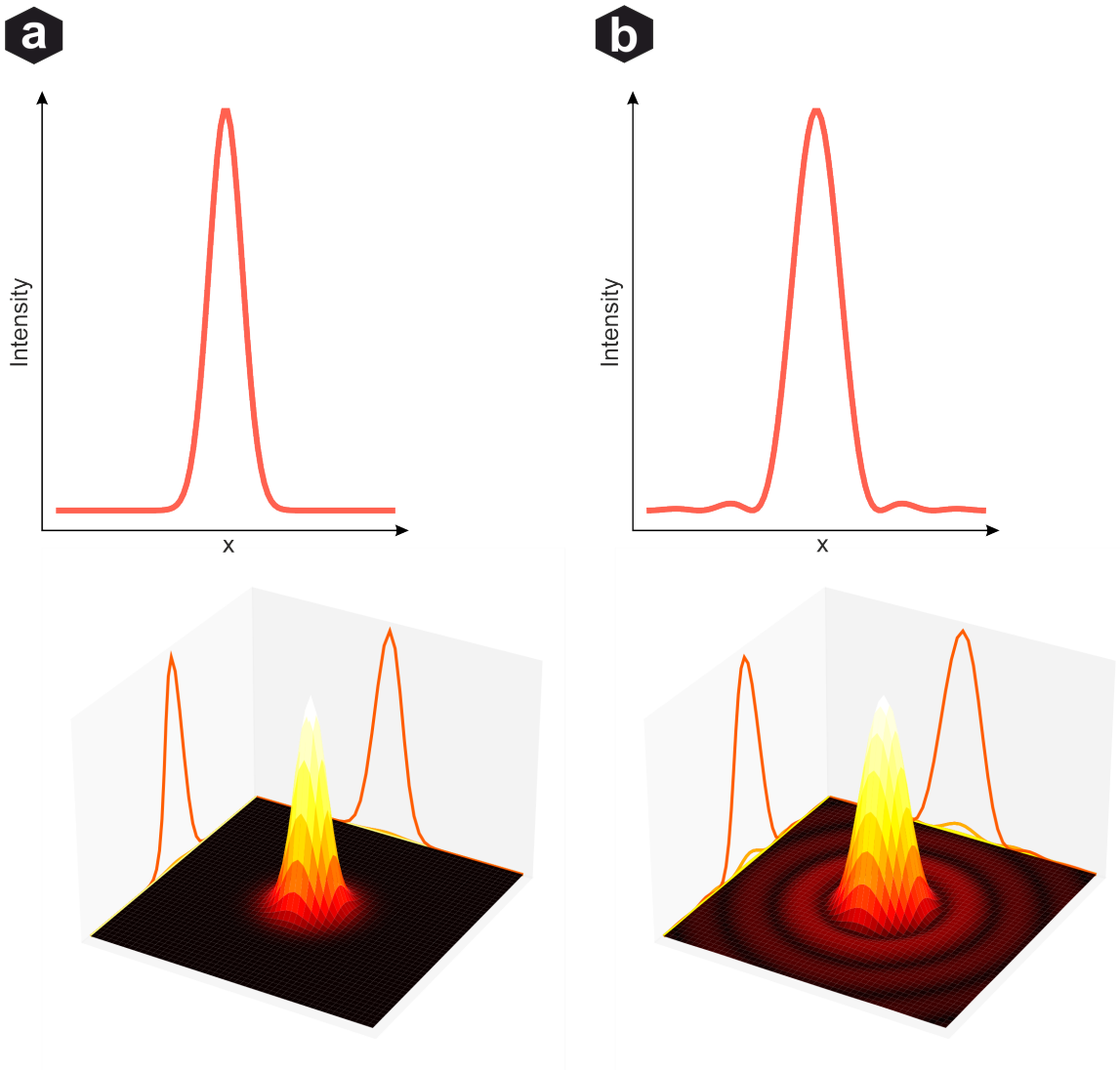

Figure 9: **Gaussian distribution (a) and Airy disc (b)** Despite being the better approximation for the actual PSF the Airy disc is rarely used for fitting. Side maxima often reside under the noise level and the Gaussian approximation provides similar results with a simpler model.

complex estimations like the airy disc shown in Figure 9 b, take the diffraction along a lens or a pinhole into account:

$$\mathrm{PSF}_{\mathrm{airy}}(r) = \left(\frac{J(r)_1}{r}\right)^2. \tag{17}$$

$J(r)_1 = \sum_{m=0}^{\infty} \frac{(-1)^m}{m!\Gamma(m+2)} \frac{x}{2}^{(2m+1)}$ is the Bessel function of the first kind and first order, with Γ denoting the Gamma function and r the radius in polar coordinates. The additional level of complexity in the airy disc, however, only yields little benefit for practical purposes since the resulting distribution is also radial symmetric and side maxima are often under the noise level. Better accuracy can be achieved using a measured PSF which can include microscope-specific non-linearities arising e.g. due to optical errors in the lenses [41].

Engineering the PSF through the optical pathway can encode additional information about the axial location of the emitter. Huang et al. [42] determined the axial position of an emitter by installing a cylindrical lens, leading to astigmatism depending on z-position. Pavani et al. [43] used a double helix PSF shape to encode highly resolved axial information. However, introducing a new degree of freedom makes accurate fitting even more difficult.
With the rising popularity of SMLM the number of available reconstruction algorithms also skyrocketed. To evaluate the performance of these software packages, theoretical limitations were assessed and quality metrics were introduced.

2.4.2 Statistical uncertainty: The Cramér-Rao Lower Bound

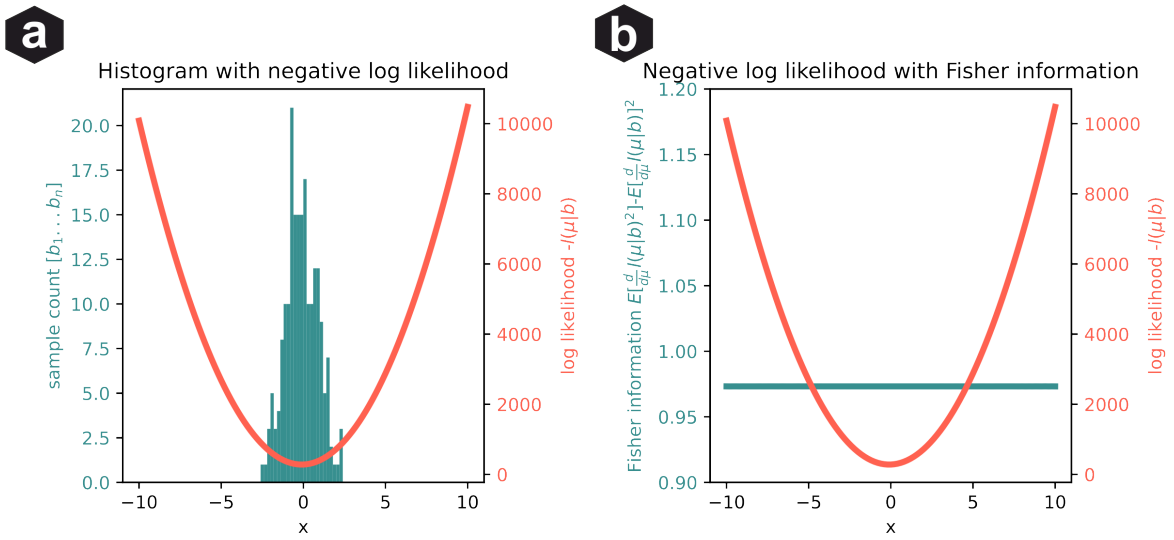

Figure 10: **CRLB** a) Gaussian distributed photons ($N = 200$ counts, $\mathcal{N}(0, 1)$, cyan) together with the negative log-likelihood (orange) for the underlying mean value μ. It can be seen, that the negative log-likelihood for the Gaussian probability density function (cf. Figure 8) given the shown data minimizes for $\mu \approx 0$. Minimizing the negative log-likelihood yields the maximum likelihood estimation. b) The Fisher information (cyan), i.e. the uncertainty of μ, is derived with the derivative of the negative log-likelihood (orange) (cf. Equation 19). The Fisher information is constant, i.e. independent of the estimated μ of the distribution. Its value is close to the theoretically derived value of $\frac{1}{\sigma_0} = 1$. Increasing the sample count decreases the deviation from the theoretical value.

Fitting a distribution to a data set, the underlying variables can only be determined up to a certain precision. The best precision a reconstruction algorithm can achieve for a variable θ is defined by the Cramér-Rao Lower Bound (CRLB) which describes a lower limit for the variance of the variable σ_θ^2:

$$\sigma_\theta^2 \geq \frac{1}{nF(\theta)}. \tag{18}$$

n denotes the number of evaluated samples and $F(\theta)$ the Fisher information of the evaluated variable, a measurement for the information on the said variable. In localisation microscopy, the variable of interest is in most cases μ, the centre of the Gaussian distribution. To deduce a simple approximation for the localisation precision σ_μ, θ is set to $\theta = \mu$. The corresponding Fisher information can be computed by the derivative of the likelihood function $l(\mathbf{b}, \mu) = \log(\mathrm{PSF}(\mathbf{b}, \mu))$:

$$F(\mu) = \mathrm{var}\left(\frac{\delta l(\mathbf{b},\mu)}{\delta \mu}\right) = E\left(\left[\frac{\delta l(\mathbf{b},\mu)}{\delta \mu}\right]^2\right) - \left[E\left(\frac{\delta l(\mathbf{b},\mu)}{\delta \mu}\right)\right]^2, \tag{19}$$

where E denotes the expected value and $\mathbf{b}$ the data points, i.e. the measured pixel values. If μ is the mean value of the true distribution it can be shown that $E\left(\frac{\delta l(\mathbf{b},\mu)}{\delta \mu}\right) = 0$. Because the fitted μ is a maximum in the likelihood function, its derivative, the gradient at that position, is zero. Figure 10 shows a Gaussian distributed sample count, the logarithmic probability for the mean value being at this point and the derivative of the log probability. The second term of the Fisher information can be simplified to $E\left(\left[\frac{\delta l(\mathbf{b},\mu)}{\delta \mu}\right]^2\right) = -E\left(\left[\frac{\delta^2 l(\mathbf{b},\mu)}{\delta \mu^2}\right]\right)$. Since Fisher information is additive, the result for all measured values $b_1, ..., b_n$ can be added up. Using equation 15, the simple Gaussian distribution, as approximation for the PSF, the expression leads up to the common estimation [5] for the uncertainty of the localisation's center μ:

$$\sigma_\mu \geq \frac{\sigma_0}{\sqrt{N}}, \tag{20}$$

where σ_0 denotes the standard deviation of the Gaussian, N the number of measured photons, i.e. the number of taken samples. Since the PSFs sigma has a lower limit, the number of photons is the only variable factor in the equation and, therefore, it is crucial to collect as much as possible. This also holds for more complex CRLB approximations like [44]:

$$\sigma_\mu \geq \sqrt{\frac{\sigma_a^2}{N}\left(\frac{16}{9} + \frac{8\pi\sigma_a^2\xi^2}{Na^2}\right)}, \tag{21}$$

with $\sigma_a = \sigma_0^2 + a^2/12$, taking the pixel size a and background noise ξ into account. Modern reconstruction algorithms achieve precision (l_2-norm, will be eq. 28) close to this limit under suitable measurement conditions.

2.4.3 Emitter uncertainty: The Jaccard index

Fitting under high emitter densities or bad signal-to-noise ratios, remains, however, pretty challenging. If emitters are missed (false negative F_n) or predicted into noise (false positive F_p) additional quality metrics are needed. The Jaccard Index (JI)

$$I_J = \frac{T_p}{F_p + T_p + F_n} \tag{22}$$

computes the ratio of correctly predicted emitters T_p to the sum of all present T_p and falsely detected $F_p + F_n$ emitters. It ranges $I_J \in [0, 1]$, with 1 indicating a perfect reconstruction and 0 indicating no correctly predicted localisations.

2.4.4 State of the art

To provide a simple overview of the strengths and weaknesses of algorithms in certain conditions, the super-resolution fight club was founded [45]. The algorithms were tested for their performance on 2D and 3D data, as well as difficult signal-to-noise ratios and high emitter densities. Results are sorted in a scoreboard together with the corresponding computation time. Algorithms can be divided into three classes:

Classical fitters use a combination of denoising and or spot detection to identify possible regions of interest. These are subsequently further analysed. A popular choice for a fit-function that approximates the regions as well as possible, is the minimization of the squared difference (c.f. l_k-norm eq. 28) between function f and data $\mathbf{x}$: $\min||f(\mathbf{x})||_2^2$. A suitable method for this is the Levenberg-Marquardt algorithm that is e.g. used in RapidSTORM [8] or ThunderSTORM [9]. It combines the Gauss-Newton algorithm with gradient descent (cf. Section 3.2.7). Maximum likelihood estimations [44] aim to find the optimal positions (x, y) for potential emitters to maximize the probability $P(I, x, y)$ to measure the given image I. With good noise models and signal-to-noise ratios, this method approaches the CRLB. The metric can be conveniently computed by taking the inverse of the second derivative of the logarithm of the likelihood [46]. The diagonal of the resulting Fisher Information matrix is the CRLB, directly giving an uncertainty for the estimated (x, y) position.

Compressed sensing (CS) algorithms originate in signal processing. They were developed for highly resolved data from poorly sampled signals under the constraint that the underlying ground truth is sparse. The obtained reconstruction can even exceed the quality predicted by the Nyquist-Shannon sampling theorem (c.f. Section 3.2.1). Since localisation microscopy raw images only contain a very sparse subset of active emitters, compressed sensing algorithms are well suited for this problem. Applied to SMLM, the reconstruction problem would formulate as finding the sparsest possible solution that describes a measurement $\mathbf{b}$ under a noise-induced error rate λ and a matrix A describing the measurement process on the ground truth $\mathbf{x}$. The matrix A is here, however, rectangular since the ground truth resides on a finer grid than the measurement, leading to an under-determined system. Mathematically, the problem is described by the equation

$$||A\boldsymbol{x} - \boldsymbol{b}||_1 \leq \lambda, \tag{23}$$

where the l_1-norm (Eq. 28) is used. Despite not being directly solvable, a solution can be efficiently approximated by greedy algorithms [47]. Popular software packages of this class are Falcon [48], Faster STORM [49] or ADCG [50]. In comparison to other approaches, CS algorithms achieve outstanding results under high-density conditions allowing faster acquisition times. However, the evaluation demands a huge computational effort. The computational cost of most greedy algorithms rises with $\mathcal{O}(N^2M^2)$, where N, M denote the image size. Most software packages are therefore either relatively slow or difficult to set up due to hardware-specific optimizations.

Artificial intelligence (AI) is in essence a tool for problem inversion. While classic approaches determine a solution by applying a theoretical model to measured data, AI takes data and solutions to estimate an appropriate model. Since the understanding of noise in the optical pathway and the underlying photophysics recently got a lot better, highly accurate datasets can be simulated to train reconstruction networks. Nehme et al. developed DeepSTORM [51], a network that reconstructs a super-resolved image by processing the stack of single-molecule data. Another approach is the work of Speiser et al.: DECODE [52]. DECODE encodes the estimated position as well as other important parameters for SMLM into pixels resulting in a classical localisation file. The advantage of this kind of reconstruction algorithm is its capability to adapt to a variety of different PSFs. Taking the temporal context of previous and subsequent frames into account, AI fitters are able to distinguish high emitter densities or compensate for occurring non-linearities. After the training procedure, reconstruction is comparably fast. Possible disadvantages arise if the training data differs from the measured images. Overfitting can lead to outstanding performances on artificial samples that can not be applied to realistic problems. Another approach worth mentioning is ANNA-PALM [53]. As has been previously shown, the shape giving information of a point cloud is mainly defined by a few control points [54]. With a full dataset, this feature can be used to train a network on predicting a full super-resolved image from a small number of localisations. However, the structure used for training has to be the same as the predicted one. The algorithm also does not compensate for unforeseen features and, therefore, sufficient quality control is a must.

2.5 Expansion microscopy

A relatively new approach to enhance resolution in the biological context is Expansion Microscopy (ExM) [10]. ExM works by linking labels into a swellable polymer grid. Biological structures are digested, e.g. chemical bonds are broken to prevent the cell from sticking together during the subsequent expansion process. The hydrogel with the linked fluorescent markers can then be grown under the addition of water. Thus, Diffraction-limited spots get physically separated, allowing for higher-resolution imag-

ing. The feasible growth ranges from 3x [55] up to 20x [56]. However, higher rates often come with more complex protocols or signal loss. Labelling with fluorescent dyes can happen before or after expansion. The corresponding protocols are called pre- and post-labelling. If enough epitopes survive the expansion process to allow for post-labelling, this method also reduces the linkage error. The linkage error describes the distance between the labelled protein and emitter and depends on the labelling method [11].

3 Algorithmic Theory

With the gained insights into modern microscopy methods, it is obvious that a computational evaluation is indispensable. In fact, algorithms are a key factor for the quality of reconstructed images and help to quantify relationships that can hardly be seen by the eye. This chapter will introduce the theoretical concepts and algorithms for computational image processing used for this thesis. The chapter starts with a formulation of the notations used throughout this thesis. The basic introduction to image processing will include the Fourier transform, filters, correlation indices, wavelet and image transforms. Digging deeper into the material, we will examine the concept of optimization and Compressed Sensing. The second part of the chapter will cover Artificial Neural Networks. We will deduce the concept from the neurons that can be found in the human brain. The artificial neurons will be brought into context with loss functions, regularizations and activations, which form the fundamental building blocks of expressive neural networks. Finally, it will be shown how the flow of information can be tuned to achieve the outstanding performance of state-of-the-art neural networks.

3.1 Notations

Notation is a tool to describe complex problems in a short and precise way for a person that is familiar with it. It expresses operations in a logical form to create an ambiguity-free environment. Therefore, this section will be used to establish a strong and consistent notation throughout this thesis. In parts, the applied notation is adapted from [57]. Overall it was tried to use common or simple ways to describe data formats and operations. For high dimensional data we include a complex description, where additional information is necessary, and a simplified version if this information is arbitrary.

3.1.1 Data

A **scalar value** is described by a lowercase letter, i.e. $x = 0$. If this value is part of a high-dimensional data structure, the index within the structure is denoted with a subscript ($x_i = 0$). **Vectors** are denoted with bold lowercase letters:

$$\mathbf{v} = \{v_i\}^I. \tag{24}$$

The superscript indicates the number of components. A **matrix** is represented by an uppercase letter:

$$M = \{M_{i,j}\}^{I,J}. \tag{25}$$

Tensors are denoted as:

$$\mathbf{T} = \{T_{i,j,k,\dots}\}^{I,J,K,\dots}. \tag{26}$$

3.1.2 Operations

Functions are written as lowercase letters in combination with a function call, depicted by round brackets:

$$f(x) = x^2 \tag{27}$$

The **k-norm** of a higher dimensional data structure is described by:

$$||\{v_i\}^I||_k = \sqrt[k]{\sum_{i=0}^{I} v_i^k} \tag{28}$$

for a $k \in \mathbb{R}$. Different norms have fundamentally different properties. In general, they provide a measurement of similarity and are often used in optimization processes like artificial intelligence, Compressed Sensing or fitting. The **Hadamard product** is the elementwise product of a vector, matrix or tensor:

$$A \circ B = \{A_{i,j} B_{i,j}\}^{I,J} \tag{29}$$

The Hadamard product preserves the dimension of the input. The **tensor product**:

$$\boldsymbol{v} \otimes \boldsymbol{u} = \{v_i u_j\}^{I,J} \tag{30}$$

expands the dimension of the input types. Using it on two vectors results for example in a matrix. The **dot product** of two vectors results in a scalar:

$$c = \boldsymbol{v} \cdot \boldsymbol{u} = \sum_{i}^{I} v_i u_i \tag{31}$$

The **floor division** describes a division which is rounded down to the next integer value:

$$\left\lfloor \frac{5}{3} \right\rfloor = 1 \tag{32}$$

The **modulo operation** returns the remainder of a division:

$$5 \mod 3 = 2 \tag{33}$$

3.2 Image processing basics

An image is defined by a matrix of picture elements. Picture elements or in short pixels contain a value of a certain range defining the brightness of the element. These pixels are discrete samples of a spatial function with constant spacing. The spacing defines the spatial resolution of an image which will be further explained in the Fourier transform chapter 3.2.1. Detectors for microscopy modalities often count photons without distinguishing wavelengths or rather colours. The resulting data are so-called grayscale

images, which contain only one value per pixel. These values are displayed as shades of black and white. The range of the present values is defined by the pixel's bit count and is usually a multiple of eight. Note that even if the detector has a physical range of 13 bits the resulting image is projected on 16 bits. An unsigned pixel with 8-bit has a range of $[0, 255]$ and is composed of eight binary values. The binary values represent a two to the power of their position. For example, 00001001 would yield 2^3 for the one at the fourth position (from right to left) and 2^0 for the one at the first position, adding up to $2^3 + 2^0 = 9$. The signed version covers half of the (value) range since the first value is used to indicate a "+" or a "-". Resulting in a range of $[-128, 127]$.
Multiple values can be used to display colours. The most popular format RGB composes the colours **R**ed, **G**reen and **B**lue to a wide variety of shades. Other common formats are the subtractive CMYK colour space or HSV. Since coloured images do not play a major role in this thesis, the interested reader can find further literature on the topic in [58, chapter 2.2].

3.2.1 Fourier transform

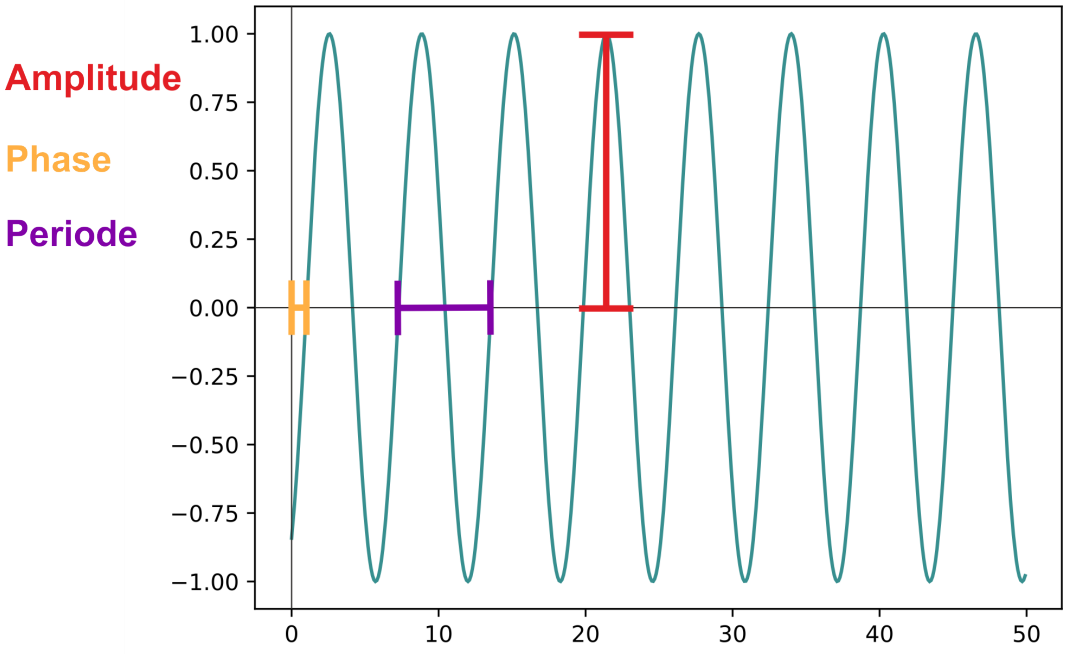

Figure 11: **Sine signal** a) A sine signal is described by an amplitude a, a phase ϕ and a period p. The period is often replaced by the frequency $k = \frac{1}{p}$. b) Sine signal with decreasing sampling rate. If the sampling rate drops too low, oscillations are skipped and the frequency can't be recognised anymore.

Breaking it down to a set of data points, an image can be considered a two-dimensional discrete signal. Apart from the spatial domain, it is often beneficial to analyse signals

for their periodic components, the underlying frequencies. A suitable tool for the task is the Fourier transform. To get a better grasp of the concept, we will start off with one-dimensional signals before expanding the method to two-dimensional images. Figure 11 shows a periodic sine wave. A signal f is called periodic, if there exists a number p for which holds: $f(x+p) = f(x)$. p is called the period and is coupled to the frequency $k = 1/p$. Overall, a periodic signal can be described by three components: An **amplitude** a determining the height of the signal, a frequency k determining the periodicity and a **phase** ϕ determining the offset of the periodic function in x to zero. This adds up to:

$$y(x) = a\sin(2\pi kx + \phi) \tag{34}$$

Getting back to the concept of analysing signals, the Fourier transform states that any continuous signal can be represented by a linear combination of sine functions [59, S.177].

$$f(x) \approx \sum_{k=1}^{\infty} \gamma_k \sin(kx + \phi_k) = \frac{\alpha_0}{2} + \sum_{k=1}^{\infty} (\alpha_k \cos(kx) + \beta_k \sin(kx)) \tag{35}$$

where the Fourier coefficients γ_k, which can be decomposed into α_k and β_k. These can be calculated by the Fourier transform:

$$\tilde{f}(k) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi kx} dx, \tag{36}$$

solving the above integral leads to α_k as the real part of $\tilde{f}(k)$ and β_k as the imaginary part of $\tilde{f}(k)$.

Getting back to discrete data points, the quality of the signal directly depends on the number of samples taken from the continuous signal. Looking at Figure 11 b one can see that the information about higher frequencies deteriorates with a decreased sampling rate. The limit of equally spaced sampling points b that have to be taken to resolve a certain frequency k_{max}, is defined by the Nyquist-Shannon [60] sampling theorem:

$$b \geq 2k_{max}. \tag{37}$$

Fewer b's can lead to aliasing. An effect that describes the detection of a non-present signal by under-sampling of the ground truth. Therefore, the number of coefficients needed to reconstruct a discrete sample is finite and equals the number of points that were sampled. The transformation to receive these coefficients is called the discrete Fourier transform and differs from the continuous Fourier transformation mainly by changing the integral to a sum:

$$\gamma_k = \sum_{x=0}^{X-1} a_x e^{\frac{-2\pi ikx}{X}}. \tag{38}$$

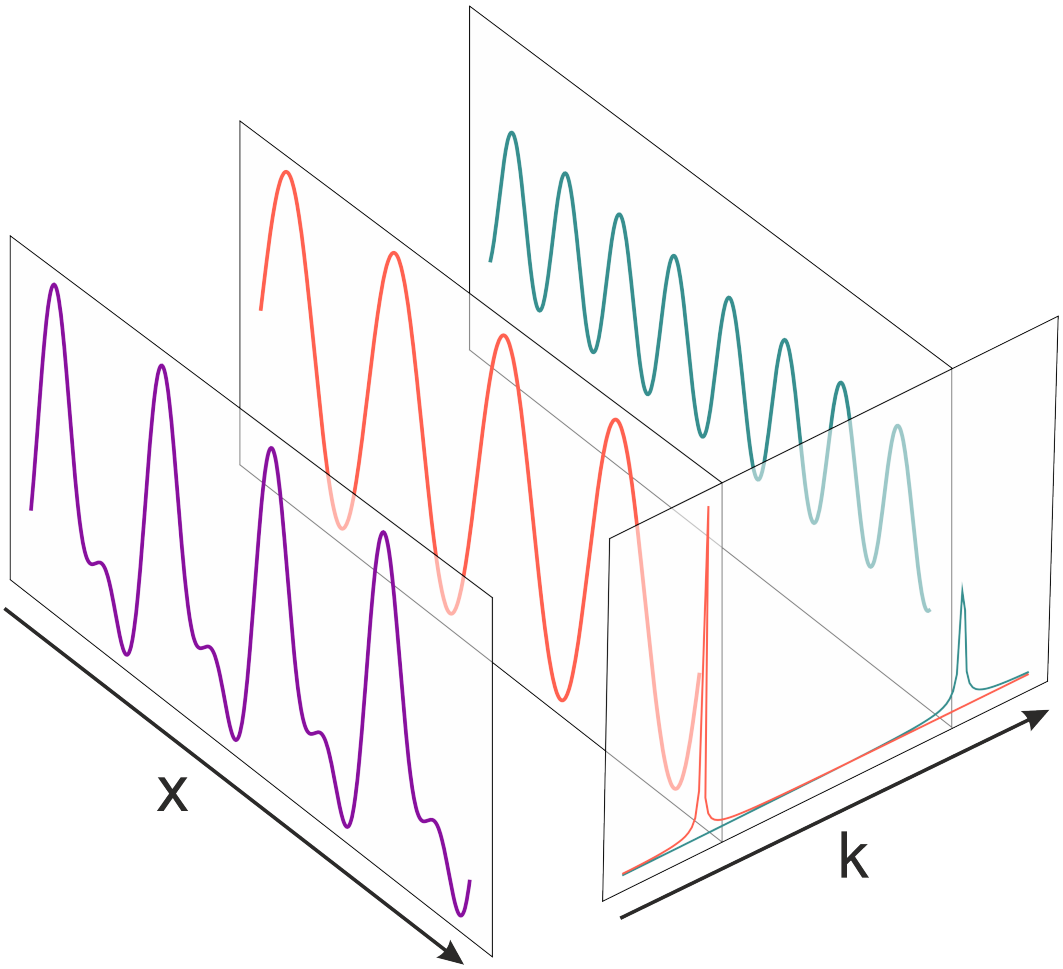

Figure 12: **The Fourier transform** decomposes a signal $f(x)$ of superimposed sine waves in a room of complex numbers $\tilde{f}(k)$. The real part of $\tilde{f}(k)$ contains the amplitude of a frequency, while the imaginary part contains the phase. Superimposing two signals with the same frequency but different phases and amplitudes results in a new signal with identical frequency but changed phases and amplitudes.

Since programming languages like python are not able to work with imaginary numbers, it is beneficial to decompose γ_k into a real part α_k and an imaginary part β_k. Using Euler's formula $e^{i\phi} = \cos(\phi) + i\sin(\phi)$ we receive:

$$\alpha_k = \sum_{x=0}^{X-1} a_x \cos(-\frac{2\pi kx}{X}) \tag{39}$$

$$\beta_k = \sum_{x=0}^{X-1} b_x \sin(-\frac{2\pi kx}{X}) \tag{40}$$

With the frequency decomposition of the signal, we can perform operations that are not possible or computationally very expensive in real space. Common examples are signal compression, auto-correlation functions, fast convolutions with large kernels and signal analytics. For some of these operations, the signal has to be transformed back

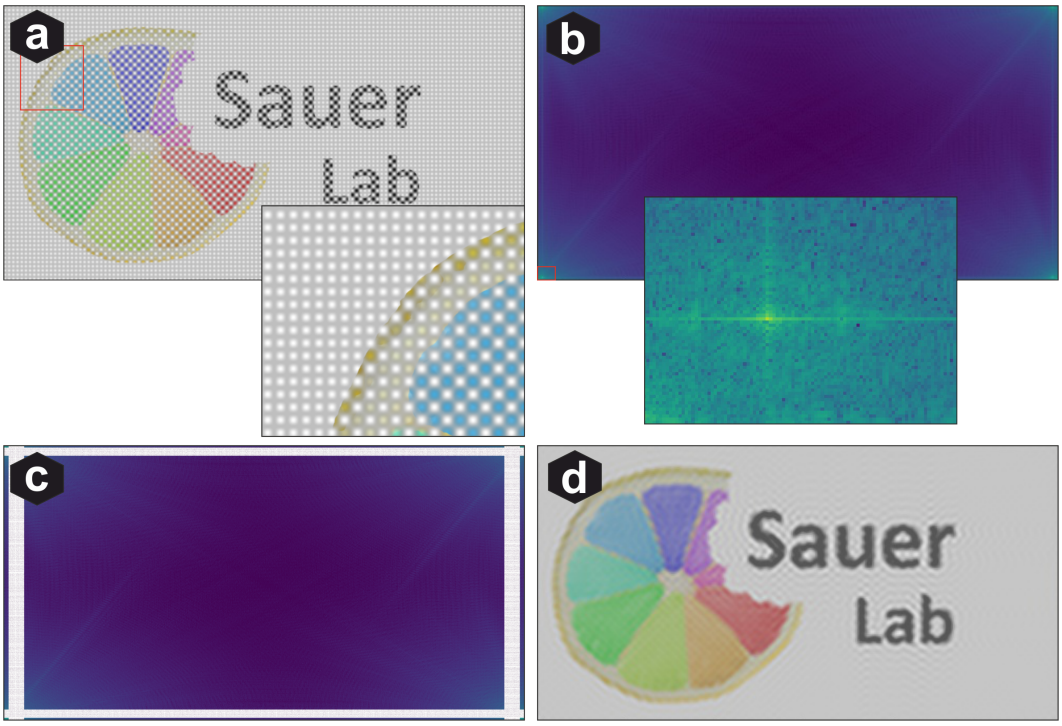

Figure 13: **The Fourier transform of images** a) Image that is afflicted by pattern noise. b) In Fourier space these patterns peak at a certain frequency. c) Cutting out the corresponding frequency bands removes the patterns from the image. (d) Removing a broader frequency band also decreases the quality of the image.

from the Fourier space into real space. This can be performed with the inverse Fourier transform:

$$b_x = \sum_{k=0}^{K-1} \beta_k e^{\frac{2\pi i k x}{K}}. \tag{41}$$

With the frequency decomposition of discrete, one-dimensional signals we can further expand the concept to two-dimensional images Γ:

$$\Gamma_{k,l} = \sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} A_{x,y} e^{-i2\pi(\frac{kx}{X-1}+\frac{ly}{Y-1})}, \tag{42}$$

where y denotes the second spatial dimension and l the second frequency dimension. An example of a common application can be seen in Figure 13. The original image is afflicted by a linear noise pattern. In microscopy, this is a common artefact that occurs for example when the sample was excited with structured illumination. Transforming the image into Fourier space, the patterns are clearly visible as bright dots at the corresponding frequencies. Setting these frequencies to zero corrects the noise pattern in real space. The result can be optimized by narrowing the band of the discarded frequencies since these contain important information about the underlying structure.

3.2.2 Linear filters

Another closely related operation for image processing is the so-called filtering. Filters are distinguished into two categories: Linear and nonlinear. Linear filters can be described by a convolution, which is analytically defined by the integral [61, chapter 13]:

$$h(x) = \int_{-\infty}^{\infty} k(n) f(x-n) dn \tag{43}$$

of two functions $f(x)$ and $k(x)$. As shown in Figure 14 the integral describes the overlapping area of the function and the moving kernel. The area linearly increases until both functions lie on top of each other and then start decreasing again. The convolution of two rectangular functions thus results in a triangular function.

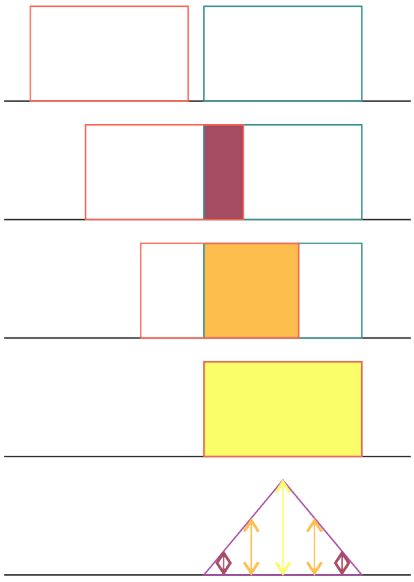

Figure 14: **Convolution of two rectangular functions** The result of the convolution is defined by the overlap of the two rectangular functions at a certain time point. The red, orange and yellow rectangles mark the overlapping area. The corresponding value in the resulting function is indicated with an arrow of the same colour.

For large images and filter kernels, the convolution integral is computationally very expensive. A neat trick is to apply the convolution theorem [62, chapter 3.1] in these cases. In Fourier space, the convolutional operation translates into a simple multiplication $\tilde{h} = \tilde{k}\tilde{f}$. A look at the frequency response of a kernel can therefore provide a better understanding of its impact on the original signal. Suitable kernels can be used as high-pass for the elimination of low frequencies, low-pass for a cutoff of high frequencies and band-pass for an interval of frequencies.

The discrete convolution (Figure 15) can be pinned down to three operations: 1) Take a sliding window with the size of the filter kernel at position i of the signal. 2) Compute the Hadamard product, the element-wise multiplication of the inverted filter kernel and the window of the signal. 3) Sum the resulting matrix to receive the new entry at position i. The operation adds up to the following equation [58, chapter 3]:

$$h_i = \sum_{n=0}^{n=N} k_n f_{i+N/2-n}, \tag{44}$$

where k_n is the n-th value of the filter kernel and f_i is the i-th value of the signal. One can immediately see that this leads to problems for $i - N/2 < 0$ and $i + N/2 > I$ at the edge of the signal, where negative entries will be indexed. Possible workarounds are to either reduce the size of the output signal by

$(N-1)/2$ only including **valid** values into the computation or to create the missing values. Common solutions for estimating values are to **mirror** entries at the edge of the signal or to assign a **constant** value. Convolutions often include a so-called stride. The stride describes the operation of either down or up-sampling, depending on the type of convolution. A down-sampling convolution by a factor of two ($\downarrow 2$), is shown in Figure 15. Here, every second value of the input signal is skipped:

$$h_i = (\downarrow 2) f_i = f_{2i}. \tag{45}$$

If included in a discrete convolution the operation is given by:

$$h_i = \sum_{n=0}^{n=N} k_n f_{2i+N/2-n}. \tag{46}$$

Notice, that the discarded values f_{2i+1} are still included in the kernel operation, as indicated by the shadow in Figure 15 b. However, the centre of the kernel skips one value during the process, as indicated by the cyan values. The up-sampling convolution, on the other hand, increases the image size. A stride of two as shown in Figure 15 c pads zeros (white panels) between the original signal pixels (magenta). The operation formulates as:

$$h_i = (\uparrow 2) f_i = \begin{cases} f_{i/2} & \text{if} \quad i/2 \in \mathbb{N} \\ 0 & \text{else} \end{cases} \tag{47}$$

During the upward convolution, no values are skipped. Hence, it is simply applied before the convolution process and not included in the operation.

If a linear filter k_n is separable the operation can be applied successively over multiple

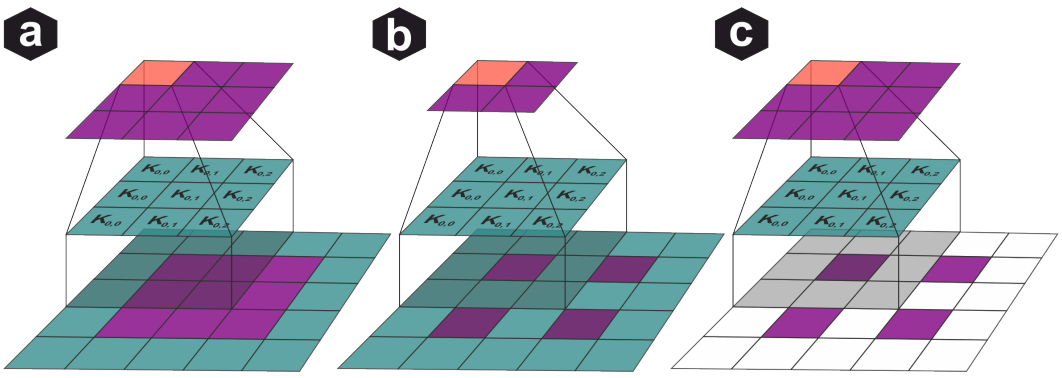

Figure 15: **Convolution** The values of the filter kernel $K = k_{n,m}^{N,M}$ are multiplied with the corresponding values of the image. This operation, the Hadamard product, is indicated by the shadow over the image. The sum of the computed values divided by the kernel size gives the new entry in the filtered image (red). In "valid" mode the image shrinks by $(N-1)/2$ and $(M-1)/2$ pixels. a) is a convolution with stride 1. b) is a downward convolution with stride 2 shrinking the image. c) is an upward convolution with stride 2. White pixels are empty. The operation therefore extends the image size from (2,2) to (3,3).

dimensions. Commonly used examples for this kind of filter are:

- **Gaussian filter:**

$$K_{gaus} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{48}$$

- **Box filter:**

$$K_{box} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{49}$$

- **Sobel filter:**

$$K_{sobel} = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{50}$$

A filter is separable if its kernel can be written as the tensor product $K = \mathbf{v} \otimes \mathbf{u}$. In the case of a Gaussian filter for example the operation can be written as:

$$F_{gaus} = \frac{1}{4}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \otimes \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}. \tag{51}$$

Non-separable linear filters are uncommon in classical image processing. However, they find their application in Convolutional Neural Networks, where every kernel value is an independent adaptive variable. During the learning process, this leads to value combinations that are not separable.

3.2.3 Nonlinear filters

Nonlinear filters, on the other hand, are also widely used in classical image processing algorithms. This category contains filters that can not be described by linear operations on neighbouring pixels. This kind of algorithm can for example be used to correct shot noise. The occasional extensive outliers would shift the mean value of a linear operation excessively to large values, resulting in distorted images. Using a filter that excludes outliers helps to improve image quality. The median filter [63, chapter 9] sorts the values from smallest to largest and picks the middlemost, i.e. the median of the values that are currently covered by the kernel. By excluding only a certain percentage of outliers, the image quality can be further enhanced, since the information is not pinned down to one value. The corresponding algorithm is called α-trimmed mean [58, chapter 3.1].

Nonlinear filters are often used to process binary images. A binary image contains only zeros and ones and can be produced by applying a thresholding algorithm. Commonly used filters for this type of image are morphological operations that change the

shape of the underlying features. Features are, in this case, the elements in the binary image that have a value of one. Morphological operations are defined by four elements: a **structuring element** that defines the shape of an applied filter. The **fit**, the event in which all pixels of the structuring element hit a feature pixel in the image. The **hit**, the event when one pixel of the structuring element hits a feature pixel. And the **miss**, which indicates no overlap of structuring element and feature. The corresponding operations are shown in Figure 16. Erosion yields 1 for a fit, which shrinks structures and strips away joints, extrusions and small structures.

$$h_{i,j} = \begin{cases} 1 & \in \textbf{fit} \\ 0 & \end{cases}, \tag{52}$$

where $h_{i,j}$ is the pixel in the constructed image. Dilation outputs 1 for a hit, extending structures, closing gaps and intrusions:

$$h_{i,j} = \begin{cases} 1 & \in \textbf{hit} \\ 0 & \end{cases} \tag{53}$$

Opening and closing are combinations of erosion and dilation. Both keep larger structures more or less unchanged since erosion and dilation are contrary operations that cancel each other out. However, this is not the case for smaller structures, where closing applies the properties of a dilation and opening the properties of an erosion [64, chapter 2-3]. Considering for example a small point and the opening operation, the object is first afflicted by the erosion operation, which completely erases the point. The subsequent dilation can not restore the point, since no feature pixels are left to score a hit. Small points or features in general vanish from the image, while larger structures are restored.

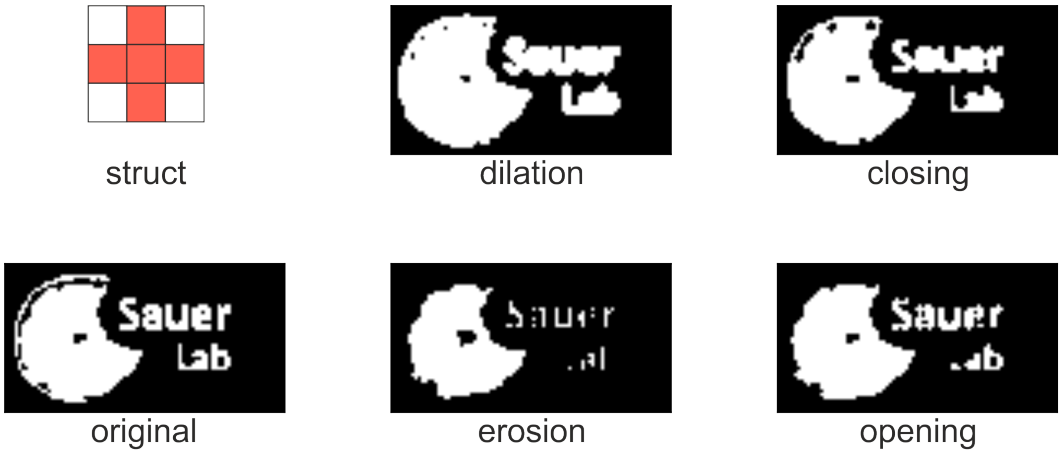

Figure 16: **Morphological operations** The Structural element for the used morphological operations. original image, erosion, dilation, closing and opening.

When working with binary structures, it is often required to bring connected pixels into context to quantify image features. The image foresting transform [65] provides the concept for a whole set of nonlinear filters, specified in this kind of analysis. The key idea is to build a graph throughout the image that starts at the feature pixels. The graph propagates like a wave and computes a recursive cost function at each iteration step. All non-feature pixels are thereby initialized with a sufficiently large value. If the cost function is smaller than the pixel's current value $v_{i,j}$, the pixel value is updated.

$$h_{i,j} = \min(v_{i,j}, \text{cost}(i,j)) \tag{54}$$

Thus, every pixel converges to its minimal possible cost. An application is the distance

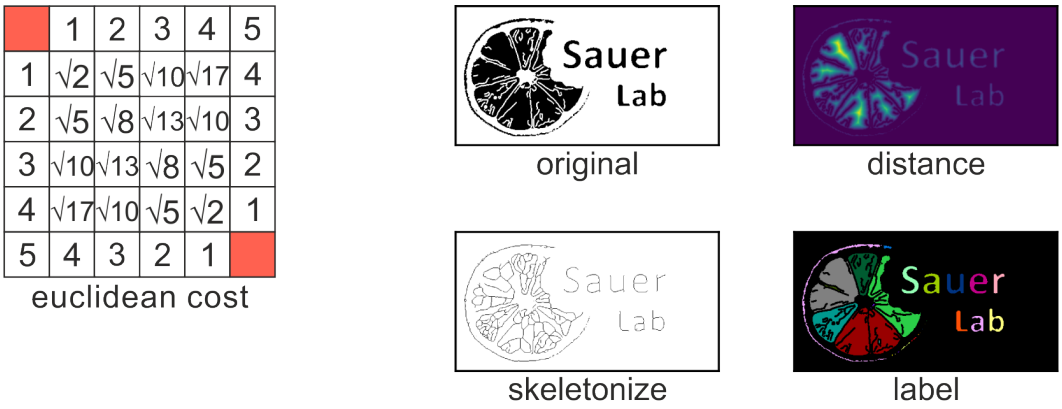

Figure 17: **Nonlinear filters** The euclidean cost of pixels within an image after the image foresting transform. A red background colour denotes feature pixels. With the binary image (original) several properties of the image can be computed. The distance transform computes the distance to the next feature pixel. The sekeletonize algorithm reduces the features to a line of one-pixel diameter. A label algorithm identifies connected components of the image.

transform, using the l_1 or l_2 distance to the next edge pixel as a cost function. It can be used as an error map in segmentation algorithms or losses/priors in AIs. The distance transform can also be used to perform multiple erosion and dilation operations in one step by including or excluding a certain distance from the current edge pixel into the feature. Skeletonize [66] captures the morphological shape of the object and can be achieved by step-wise opening and erosion. Connected component [67] works by scanning an image line by line, merging pixels of identical value into a union of labelled components.

3.2.4 Correlation Indices

As previously stated, combining different microscopy modalities can be an efficient method to bring fundamental biological processes into context. To evaluate the resulting data, it is often required to measure the similarity of images. However, optical illusions can deceive the eye into recognizing relationships that are not present [68].

Thus, a mathematical metric to measure similarity is required: the correlation index. The three most commonly used metrics are the Pearson, Manders and Spearman correlation indices. The Pearson correlation index [69] $r_p \in [-1, 1]$ ranges from perfect anti-correlation (-1) to perfect correlation (1). The deviation of the first signal $\mathbf{x}$ from the mean $\bar{\mathbf{x}}$ is multiplied by the deviation of the second signal $\mathbf{y}$ from its mean $\bar{\mathbf{y}}$. The mathematical expression formulates as:

$$r_p = \frac{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{\mathbf{x}})(y_i - \bar{\mathbf{y}})}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{\mathbf{x}})^2 \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{\mathbf{y}})^2}}, \tag{55}$$

where N is the total number of measurement points. Note that the expression is the covariance of $\boldsymbol{x}$ and $\boldsymbol{y}$ ($\mathrm{cov}(\boldsymbol{x}, \boldsymbol{y})$) divided by the standard deviation $\sigma_x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{\mathbf{x}})^2}$ and $\sigma_y = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{\mathbf{y}})^2}$ of the signals. The expression if $\boldsymbol{x}$ and $\boldsymbol{y}$ are linearly dependent. For example $\boldsymbol{x} = 2\boldsymbol{y}$ would still result in a correlation value of 1. In conclusion, the Pearson correlation index is suitable for comparing multi-colour images without preprocessing. The evaluation is independent of background and magnitude levels [68].
In contrast, the Manders correlation index r_m [70] or also called Manders overlap coefficient, takes different intensities into account. It was introduced especially to handle fluorescent microscopy images and formulates as:

$$r_m = \frac{\sum_{i=0}^{N} x_i, y_i}{\sqrt{\sum_{i=1}^{N} x_i^2 \sum_{i=1}^{N} y_i^2}}. \tag{56}$$

It ranges from mutually exclusive samples 0 to a perfect overlap 1: $r_m \in [0, 1]$. The algorithm has its advantages at comparing images of the same image modality since intensity levels and background are often camera dependent.
Another metric of similarity is the Spearman correlation index [71]:

$$r_s = \frac{\frac{1}{N}\sum_{i=1}^{N}(R(x_i) - \bar{R_x})(R(y_i) - \bar{R_y})}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(R(x_i) - \bar{R_x})^2 \frac{1}{N}\sum_{i=1}^{N}(R(y_i) - \bar{R_y})^2}}, \tag{57}$$

where $R(x_i)$ is the rank of the considered pixel x_i weighted by its intensity. Similar to Pearson's approach the Spearman rank correlation is independent of pixel intensities. It is especially useful for working with large outliers since the introduced error is limited by the number of ranks.

3.2.5 Wavelet transform

Similar to the Fourier transform, the wavelet transform decomposes a signal into frequency components. However, these frequency components still have a positional resolution. Wavelets perform a decomposition by transforming with a so-called mother

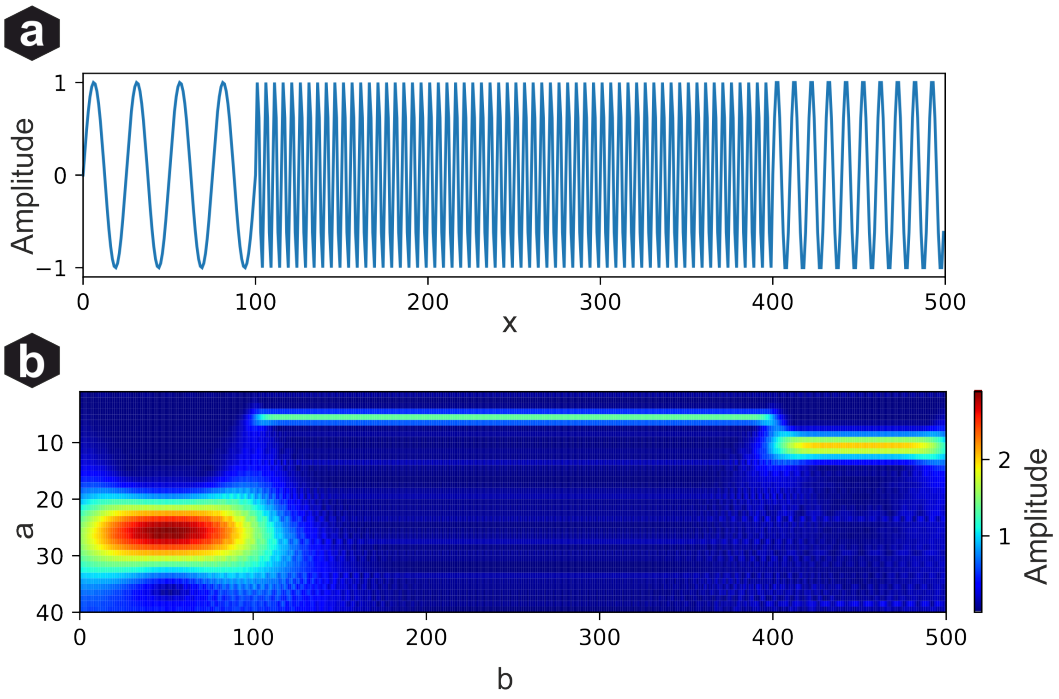

Figure 18: **Continuous wavelet transform** a) Signal with varying frequencies along x. b) Wavelet decomposition with a Morlet wavelet. The result is a function of the translation parameter b and the scaling a. It can be seen that the signal is resolved in space and frequency.

wavelet $\Psi(x)$, which can be shifted and scaled to target information at specific frequencies. The minimal constraints for this mother wavelet are:

$$\int_{-\infty}^{\infty} \Psi(x)dx = 0 \tag{58}$$

$$\int_{-\infty}^{\infty} |\Psi(x)|^2 dx = 1, \tag{59}$$

implying a zero mean and energy preservation of the input signal. For most used cases it is also useful to constrain Ψ to vanishing moments:

$$\int_{-\infty}^{\infty} x^m \Psi(x)dx = 0, \tag{60}$$

implying a convergence to zero for high positive and negative x. Having defined the mother wavelet, one can perform the continuous wavelet transform:

$$\mathbb{W}_\Psi[f(x)] = \tilde{f}(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} \Psi\left(\frac{x-b}{a}\right) f(x)dx. \tag{61}$$

Here, $f(x)$ denotes the transformed function, b is the translation parameter to sample $f(x)$ in real space and a is the scaling parameter to sample over the frequencies. Hence, the Wavelet transform extends the dimensionality of the input signal by one, display-

ing spatial and frequency information. To perform the discrete wavelet transform, the continuous Wavelet set has to be transformed into a discrete set:

$$\frac{1}{\sqrt{a}}\Psi\left(\frac{x-b}{a}\right) \rightarrow \frac{1}{\sqrt{s^j}}\Psi\left(\frac{x-ks^j}{s^j}\right). \tag{62}$$

Here, we replace the continuous variable a with a fixed scaling factor s increasing exponentially with each step j. The translation b, now denoted by k, is also given by discrete steps ($k \in \mathbb{Z}$) as well as the variable x which transforms into data points. The discrete wavelet transform for $s = 2$ is given by:

$$\mathbb{W}_\Psi[f_x] = \tilde{f_{j,k}} = \frac{1}{\sqrt{2^j}}\sum_x \Psi\left(\frac{x-k2^j}{2^j}\right) f_x. \tag{63}$$

As indicated in Figure 19 the wavelet samples half of the frequency components in every step. With increasing j the wavelet samples lower resolution frequencies, yielding a remainder with half of the sampling points. For a full wavelet decomposition, the process has to be repeated until only one pixel remains. For most tasks, this is not required. Therefore, the Wavelet transform is divided into a high pass decomposition, applying the actual Wavelet, and an orthogonal low pass decomposition, collecting the remaining frequencies in a so-called scaling function (orthogonality principle). Scaling, and wavelet decomposition are applied on the signal as strided convolutions and form together with the corresponding synthesis, a wavelet filterbank.

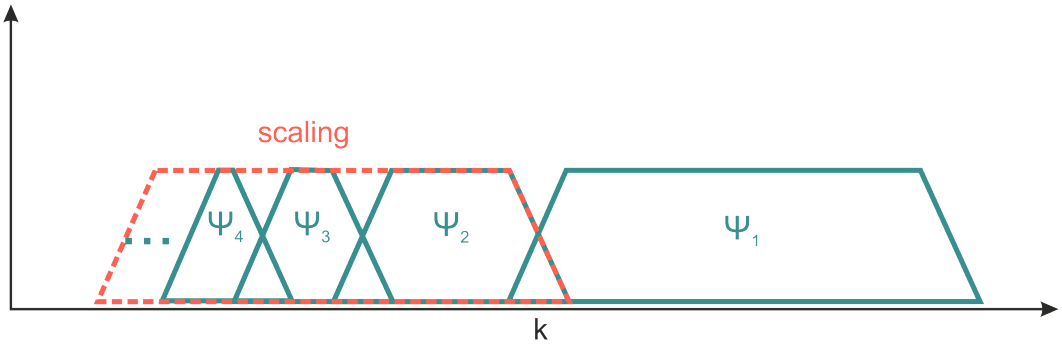

Figure 19: **Frequency enclosure of the discrete Wavelet Transform** The frequency enclosure of the wavelet is halved with every scaling step. Therefore, the filterbank decomposition is divided into a wavelet and scaling part, i.e. the rest of the signal. The decomposition is performed up to a certain level.

The simplest example of a Wavelet filterbank is the Haar-wavelet. The first order ($j = 1$) decomposition of a signal $\tilde{f}_k$ is performed by a high-pass Ψ_k and a low-pass Φ_k:

$$\Psi_k = (\downarrow 2)(\tilde{f}_k - \tilde{f}_{k+0.5}) \qquad = \tilde{f}_{2k} - \tilde{f}_k \tag{64}$$

$$\Phi_k = \frac{1}{2}(\downarrow 2)(\tilde{f}_k + \tilde{f}_{k+0.5}) \qquad = \tilde{f}_{2k} + \tilde{f}_k. \tag{65}$$

Note that the pre-factors of the filterbank can be adjusted to $\frac{1}{\sqrt{2}}$ to achieve a symmetrical distribution. The performed convolution uses a stride of two, downsampling (cf. Equation 45) the signal to two sub-signals with half of their original resolution. However, in contrast to a normal strided convolution, no information is lost. The original signal can be completely reconstructed from the two sub-signals (perfect reconstruction principle).

$$f_k = \begin{cases} (\uparrow 2)(\Psi_k + \frac{1}{2}\Phi_k), & \text{for } k \mod 2 = 0 \\ (\uparrow 2)(\Psi_{k-0.5} - \frac{1}{2}\Phi_{k-0.5}), & \text{for } k \mod 2 = 1 \end{cases} \tag{66}$$

Note here, that the two synthesis filters are combined in a way that cancels out the aliasing part of the reconstruction. In other words, the synthesis reconstructs the original signal of the two sub-sampled signals without resolution loss. This is the perfect reconstruction principle and an important property of wavelet filterbanks. Taking a

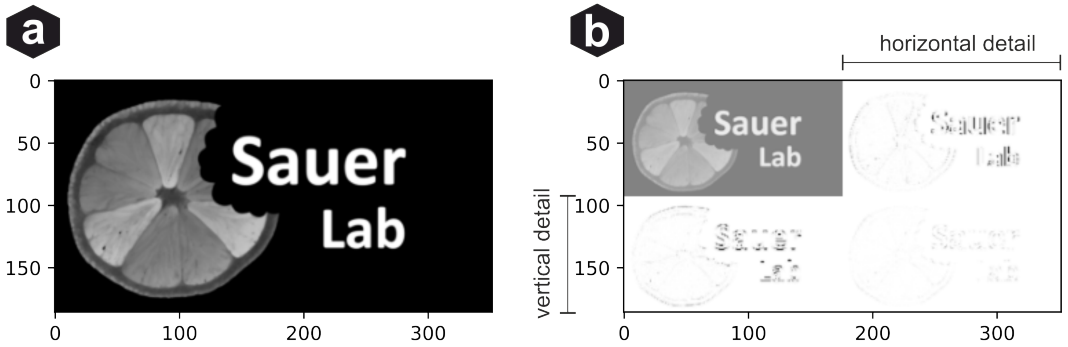

Figure 20: **2D wavelet transform** a) original image. b) Wavelet transform of a. The scaling component (low pass) on the top left contains most of the information, while sharp details are encapsulated in the wavelet component (top right, bottom right, bottom left)

look at Figure 20, it can be seen that most of the information resides in the low-pass components of the image and only a few entries in the high-pass, which significantly differ from zero. Discarding small entries in the high-pass and saving it as a sparse array is, therefore, a way to reduce the number of values needed for a close approximation of the original image. A more complex form of this approach is for example used in the common image format `jpeg`.

3.2.6 Image transformations

It is often necessary to compensate for occurring distortions, to compare image data. Comparing two microscopy modalities, a different sample position can already lead to a heavy change in the measured data [72]. Even for multi-colour images of the same modality, the chromatic error of different lenses and a different optical path leads to changes. Expanded samples often need to be compared to their unexpanded counterparts, introducing additional non-linear structural deformations [13]. A way to deal with these problems is to compute image transformations. In general, an image

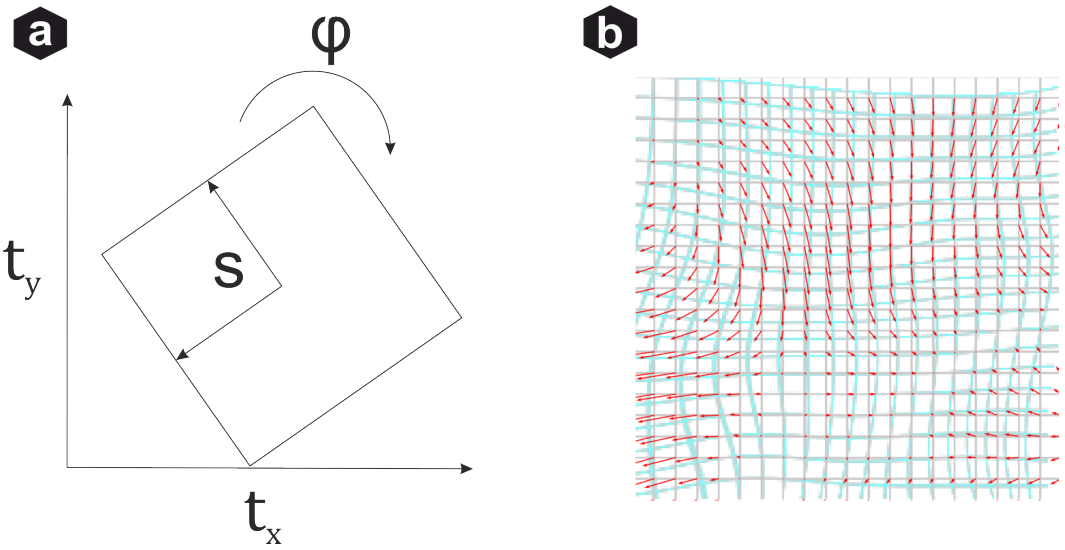

Figure 21: **Image transformations** a) The similarity transform has four DOFs: Translations in x and y direction (t_x, t_y), a rotation r and a scaling s. b) The B-spline transformation has a set of N control points of whom each can be shifted by a vector $\boldsymbol{t_k}$. This results in $2N$ DOFs.

transformation describes the coordinate mapping $T(X)$ from a source image $S(X)$ to a target image $I(X)$ such that $S[T(X)]$ is spatially aligned to $I(X)$. $T(X)$ can have various Degrees of Freedom (DOF) and the choice for an appropriate $T(X)$ should be adapted to the problem and well concerned. Choosing transformations with too high DOF can lead to overfitting, which creates correlations in the data that are not justified. The simplest $T(x)$ is the translation by a vector $\mathbf{t}$. Adding a rotation by an angle φ

$$R = \begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix} \tag{67}$$

and a scaling factor s, yields the similarity transform (Figure 21 a). The similarity transform preserves the shape of an object, adjusting its position and size. The affine transformation A scales s_x and s_y independently and introduces a shear h:

$$A = \begin{pmatrix} s_x \cos(\varphi) & -h_y \sin(\varphi) \\ h_x \sin(\varphi) & s_y \cos(\varphi) \end{pmatrix}. \tag{68}$$

While the shape is not preserved anymore, parallel lines remain parallel here. A popular nonrigid transformation is the B-spline transform (Figure 21 b). A grid of control points $\boldsymbol{t_k}$ is shifted by a vector, leading to the mapping

$$T(\boldsymbol{x}) = \boldsymbol{x} + \sum_{\boldsymbol{t_k}} p_k \beta^3 \left(\frac{\boldsymbol{x} - \boldsymbol{t_k}}{\sigma} \right) \tag{69}$$

where $\boldsymbol{t_k}$ denotes the image part of the control points, $\beta^3(x)$ the cubic B-spline polynominal [73], p_k te B-spline coefficients and σ the control point spacing.

3.2.7 Optimizers

Be it image transformation, noise estimations, or Neural Networks, we always strive to get the best possible outcome for our data. We build a model with a set of estimated parameters, evaluate the data, and test the results for plausibility. If the values are off, we adjust the model by tuning the parameters or changing the algorithm. That behaviour is called optimization, and there is a whole class of algorithms that can solve this task in a computational manner for us. To get a better grasp of the concept, consider the following problem: Given a data set with N points $\mathbf{x} = \{x_n\}^N$ and $\mathbf{y} = \{y_n\}^N$. Define a model f that yields $y_n \approx f(x_n)$. The model f contains a set of P parameters $\boldsymbol{\mu} = \{\mu_p\}^P$. Find the parameters $\boldsymbol{\mu}$ that give the best description of the relationship between $\mathbf{x}$ and $\mathbf{y}$.
In a first step we need a quantification for how well the model performs. This is called the cost function c. A suitable choice is for example least squares:

$$c(\boldsymbol{\mu}, \mathbf{x}, \mathbf{y}) = \sum_{n=1}^{N} ||f(\boldsymbol{\mu}, x_n) - y_n||_2^2. \tag{70}$$

Minimizing $c(\boldsymbol{\mu}, \mathbf{x}, \mathbf{y})$ requires an adjustment of the parameter set $\boldsymbol{\mu}$. Imagining c as a hilly landscape we want to find the deepest valley. Without a map or GPS guidance, the way would be to go downhill. Translating "downhill" to our optimization problem, we need to compute the derivative of the cost function with respect to the parameters $\boldsymbol{\mu}$. Thus, these parameters build the coordinate system of our landscape. Doing this for each datapoint $(x_n; y_n)$ yields the Jacobi matrix:

$$J = \begin{bmatrix} \frac{\delta c(\boldsymbol{\mu}, x_1, y_1)}{\delta \mu_1} & \dots & \frac{\delta c(\boldsymbol{\mu}, x_1, y_1)}{\delta \mu_P} \\ \dots & \dots & \dots \\ \frac{\delta c(\boldsymbol{\mu}, x_N, y_N)}{\delta \mu_1} & \dots & \frac{\delta c(\boldsymbol{\mu}, x_N, y_N)}{\delta \mu_P} \end{bmatrix}. \tag{71}$$

Summing over all data points gives the gradient for each parameter, i.e. the steepness in each possible direction:

$$\mathbf{g_p} = \sum_{n=1}^{N} J_{n,p}. \tag{72}$$

Taking a step in this direction and subsequently repeating the process leads to one of the simplest optimization techniques: **Gradient Descent** [74, chapter 5.1]. The iterative process for the t-th step is described by:

$$\boldsymbol{\mu_{t+1}} = \boldsymbol{\mu_t} - \alpha_t \boldsymbol{g_t}. \tag{73}$$

Note that here the subscript denotes an iteration step and $\boldsymbol{\mu_t}$ is still a vector. $\boldsymbol{g_t}$ substitutes the gradient of the cost function, $\boldsymbol{g_t} = \boldsymbol{g}(c(\boldsymbol{\mu_t}))$, and α_t is the gradient step which is optimized to hit the next valley in the gradient direction. It is subject to

$\alpha_t = \min_{\alpha} c(\boldsymbol{\mu_t} - \alpha \boldsymbol{g}(\boldsymbol{\mu_t}))$. Reaching a minimum, the gradient in the current direction

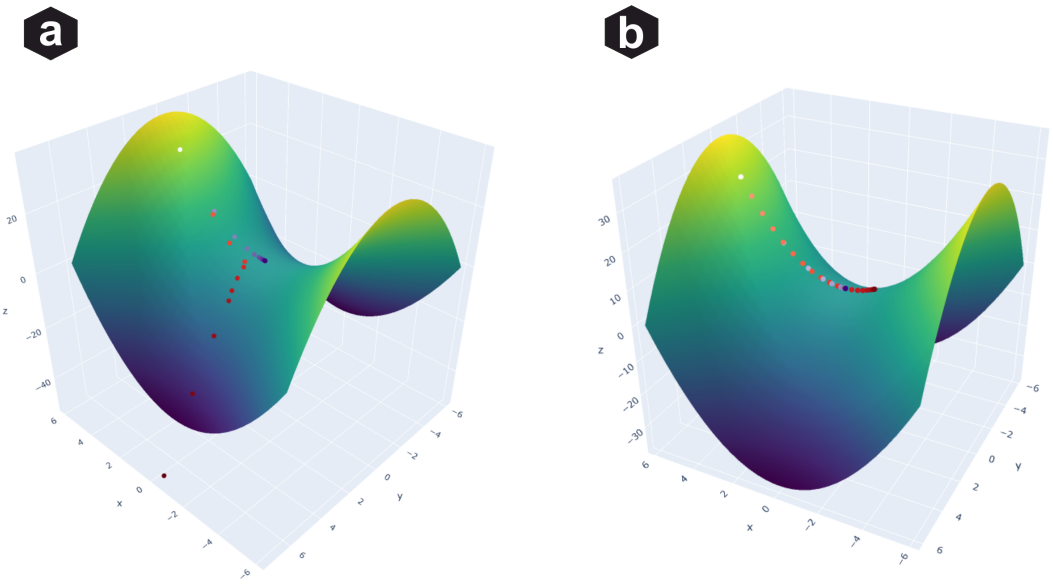

Figure 22: **First order optimization** a) Gradient Descent (purple dots) and Noisy Gradient Descent (red dots). While Gradient Descent gets stuck on the saddle point, Noisy Gradient Descent finds a way further down the feature space. b) The Adam optimizer builds momentum running downhill. It slightly ascends the other side of the parabola before converging to the saddle point.

is zero. So, if there still exists a gradient, it has to be perpendicular to the previous one, since it has to satisfy $\boldsymbol{g_t} \cdot \boldsymbol{g_{t+1}} = 0$. Gradient Descent reaches its limits when dealing with saddle points or narrow valleys. The algorithm has a very short step size in narrow valleys that do not run in the gradient direction. The progression is therefore computationally very inefficient. Hitting a saddle point, the algorithm gets stuck with no gradient left. The latter can be resolved by using **Noisy Gradient Descent** (cf. Figure 22 a) [74, chapter 8.1]. Noisy Gradient Descent adds an additional term to the update process:

$$\boldsymbol{\mu_{t+1}} = \boldsymbol{\mu_t} - \alpha_t \boldsymbol{g_t} + n(\boldsymbol{\sigma_t}), \tag{74}$$

where $n(\boldsymbol{\sigma_t})$ is Gaussian noise with zero mean and standard deviation $\boldsymbol{\sigma_t}$. Generally, the standard deviation decreases over time, decreasing the modifications towards convergence to hit the true minimum. In **Stochastic Gradient Descent (SGD)** a random subset of the data points is used, replacing the Gaussian noise term. This approach is computationally much more efficient and therefore, a good choice for neural networks.

The performance of gradient descent methods is, however, limited if feature spaces contain nearly flat surfaces. Traversing these surfaces takes a huge amount of steps since the gradient is low. This is computationally inefficient. A solution for this task is to introduce momentum. Like a ball rolling downhill, the algorithm accumulates speed which can be used to traverse flat surfaces or roll uphill to find a deeper global minimum. The most popular algorithm of this category is **Adam** [75]. Adam computes a first and a second momentum $\mathbf{m_t}$ and $\mathbf{v_t}$, the rolling average of the gradient and

squared gradient. To slow the momentum down, an exponential damping term with the constants β_1 and β_2 is introduced. Bringing everything together, the first and second momentum succumb to the following update rule:

$$\mathbf{m_{t+1}} = \beta_1 \mathbf{m_t} + (1 - \beta_1)\boldsymbol{g}_t, \tag{75}$$

$$\mathbf{v_{t+1}} = \beta_2 \mathbf{v_t} + (1 - \beta_2)\boldsymbol{g}_t^2. \tag{76}$$

The update step size α_t is adjusted according to the current step (note that β^t is to the power of t):

$$\alpha_{t+1} = \alpha \frac{\sqrt{1 - \beta_2^t}}{1 - \beta_1^t}. \tag{77}$$

Since the divisor is the higher order term, the step size decreases with increasing iterations. Finally, the update step for the parameters adds up to

$$\boldsymbol{\mu_{t+1}} = \boldsymbol{\mu_t} - \alpha_t \frac{\mathbf{m_{t+1}}}{\sqrt{\mathbf{v_{t+1}}} + \epsilon}, \tag{78}$$

where ϵ denotes another constant damping term that is typically set to 10^{-8} and prevents a division by zero. While ϵ is not discussed in detail in the original paper, its impact is well described in [76]. An illustration of the steps is shown in Figure 22 b).

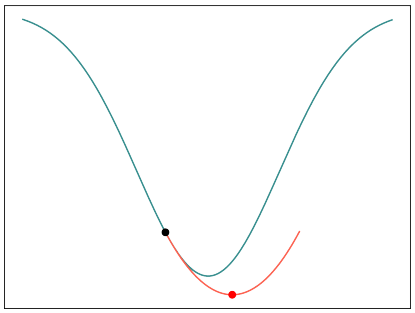

Figure 23: **Second order optimization** Starting at the black dot, a second-order Taylor approximation estimates a quadratic polynomial (orange). The minimum of that function indicates the location for the next iteration step.

The so far examined optimizers were all first-order methods as only the first derivative is used. For certain problems, like convex optimizations, it can be beneficial to include the second derivative in the optimization process.

Newton's method [74, chapter 6.1] uses a quadratic approximation to estimate the step size for the optimization process. This approximation $q(\boldsymbol{\mu})$ is computed by the second order Taylor series:

$$\begin{aligned} q(\boldsymbol{\mu}) \approx c(\boldsymbol{\mu_t}) + \boldsymbol{g_t}(\boldsymbol{\mu} - \boldsymbol{\mu_t}) \\ + H\, c(\boldsymbol{\mu_t})^T \frac{(\boldsymbol{\mu} - \boldsymbol{\mu_t})^2}{2} \end{aligned} \tag{79}$$

where H is the Hessian matrix, which contains the second derivative. Solving for $\frac{dq(\mu)}{d\boldsymbol{\mu}} = 0$ yields the minimum of the quadratic approximation and therefore, the desired point $\boldsymbol{\mu_{t+1}}$. Thus, the update rule formulates as:

$$\boldsymbol{\mu_{t+1}} = \boldsymbol{\mu_t} - \frac{\boldsymbol{g_t}}{H(c(\boldsymbol{\mu_t}))}. \tag{80}$$

While this method features quadratic convergence (cf. Figure 23) and thus, is very fast for convex problems, the second derivative is in most cases hard to compute. The **Gauss-Newton**[77] algorithm approximates the Hessian matrix for the least squares cost function with:

$$H(c(\boldsymbol{\mu_t})) = 2J(\boldsymbol{r_t})J^T(\boldsymbol{r_t}) \tag{81}$$

[78] where $\boldsymbol{r} = f(\boldsymbol{x}|\boldsymbol{\mu}) - \boldsymbol{y}$ denotes the residuals. The update rule follows:

$$\boldsymbol{\mu_{t+1}} = \boldsymbol{\mu_t} - \beta_t \frac{J^T(\boldsymbol{r_t})\boldsymbol{r_t}}{J^T(\boldsymbol{r_t})J(\boldsymbol{r_t})}. \tag{82}$$

Thus, Gauss-Newton maintains the advantages of second-order optimization, while being computationally less expensive. Adding a damping term λI with the identity matrix I to the denominator of the update term yields the **Levenberg-Marquardt** algorithm. In comparison to the Gauss-Newton algorithm, the Levenberg-Marquardt algorithm is much more robust and also works with unfavourable starting conditions.

3.3 Compressed Sensing

As mentioned in equation 37 the number of measured data has to be at least as large as the signal we aim to reconstruct. However, under certain circumstances, it is possible to reconstruct information beyond that limit. To grasp the concept we have to take a look at linear equations. Consider for example:

$$\boldsymbol{y} = A\boldsymbol{x}. \tag{83}$$

A denotes a linear measurement matrix with $\boldsymbol{x} = \{x_n\}^N$ and $\boldsymbol{y} = \{y_m\}^M$. If $M < N$ the system is under-determined, i.e. there is an infinite number of solutions for $\boldsymbol{x}$ that satisfies the equation. Picking the Fourier transform as A, the equation aims to reconstruct frequencies beyond the maximum sampling rate. There are multiple combinations of high frequencies that would result in the measured sample. However, if it is known that the underlying frequencies are sparse, a solution can be found [47] [79].
Sparsity is the property, that most components of a vector, matrix, or tensor are zero. The important information resides on the nonzero components. This property is for example used in the compression of the `jpeg` format, which discards low-intensity components in the wavelet-transformed image. Saving only the nonzero entries in combination with their indices reduces the necessary memory significantly.
Back to our underdetermined equation. If most of the entries in $\boldsymbol{x}$ are zero, the system gets overdetermined. The two missing components for a successful reconstruction are a suitable choice for A and an efficient way to minimize the equation in a sparse manner. Bringing super-resolution microscopy into the equation, the measurement process is quickly formulated. Each frame contains a sparse set of emitters at a certain position,

which we would like to reconstruct at the CRLB $\approx 10\,\text{nm}$. Estimating a pixel size of 100 nm results in $M \approx 10N$. The measurement process is defined by the PSF. Hence, A is a matrix that convolves the sparse emitter position with the PSF, while downsampling $\approx$ 10-fold.
The algorithm to maximize sparsity can be mathematically described as l_0-minimization.

$$||\boldsymbol{x}||_p = \sum_{n=0}^{N} x_n^p \xrightarrow[p\to 0]{} \sum_{n=0}^{N} 1_{x_n \neq 0} \tag{84}$$

One can see in equation 84, that $||\boldsymbol{x}||_0$ denotes the number of non zero components in $\boldsymbol{x}$. Taking the p-th root of a value results in convergence to one of every value unequal zero for $p^{-1} \to \infty$, i.e. $p \to 0$. Though, minimizing l_0 also minimizes the number of non-zero entries. The optimization of this problem is computationally very inefficient, i.e. it involves a lot of trial and error. However, a solution can be approximated by minimizing l_1.
Another important aspect of the minimization is the so-called stability of the algorithm. Considering the example of a SMLM image, $\boldsymbol{y}$ is no exact representation of the measurement process but includes statistic uncertainties (cf. chapter 2.3) which can not be included into the linear transformation A. Therefore, the minimization should work under the given error rate $||A\boldsymbol{x} - \boldsymbol{y}||_2 < \lambda$.

The desired properties can, amongst others, be found in Iterative Shrinkage Thresholding Algorithm (ISTA) [80]. It uses the following update step:

$$\boldsymbol{x_{t+1}} = S_\lambda(\boldsymbol{x_t} + \alpha A^T(\boldsymbol{y} - A\boldsymbol{x_t})) = I(\boldsymbol{x_t}), \tag{85}$$

where α is a constant and S_λ denotes the soft-thresholding operator:

$$S_\lambda(x_n) = \begin{cases} x_n - \lambda & x_n > \lambda \\ 0 & |x_n| > \lambda \\ x_n + \lambda & x_n < -\lambda \end{cases} \tag{86}$$

As explained in Section 3.2.7, tuning the step size can be beneficial in terms of faster convergence.
Fast Iterative Shrinkage Thresholding Algorithm (FISTA) [81] introduces an adjusting α:

$$\alpha_{t+1} = \frac{1 + \sqrt{1 + 4\alpha_t}}{2} \tag{87}$$

with the corresponding update step

$$\boldsymbol{x_{t+1}} = I(\boldsymbol{x_t}) + \frac{\alpha_t - 1}{\alpha_{t+1}} I(\boldsymbol{x_t}) - I(\boldsymbol{x_{t-1}}). \tag{88}$$

3.4 Artificial Intelligence

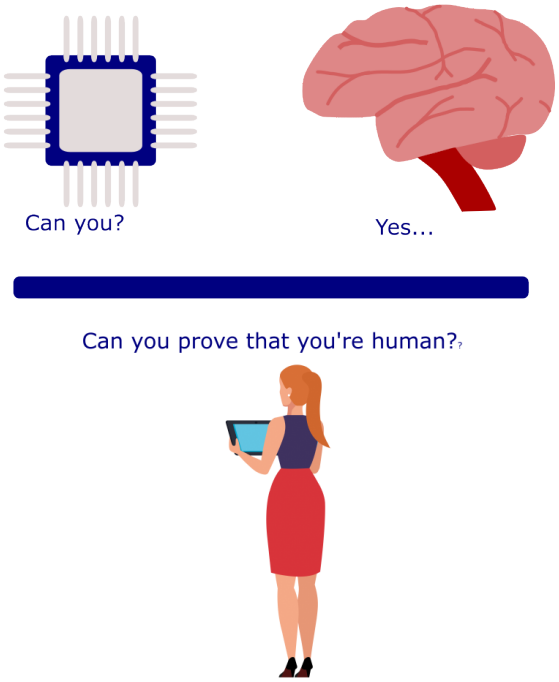

Figure 24: **Turing test** An interrogator tries to distinguish between man and machine by asking a set of questions. If the machine manages to trick the interrogator into falsely believing it is a human, it passes the test and is accepted as intelligent. [Inkscape and Vecteezy(woman)]

The introduction of a new chapter usually starts with the definition of the topic. In the case of Artificial Intelligence (AI), this is rather difficult, since intelligence is not well defined in the first place. The border where a machine can be classified as intelligent has, therefore, been the subject of scientific discussions. One of the earliest works on this topic dates back to 1950, where Turing [82] formulated the famous Turing test. The test is conducted by a professional interrogator questioning two entities behind a curtain: One is a person, the other one a tested machine (cf. Figure 24). The interrogator asks a series of questions trying to identify which entity is human and which is not. While the person tells the truth, the machine may lie to trick the interrogator into falsely believing it is human. If the machine succeeds, it passes the test and its intelligence is accepted.

Over the years the Turing test has been subject to plenty of criticism. The two major critical comments were published by Block et al. [83] and Searle et al. [84]. Block argues that a large enough database might be able to answer all questions to the interrogators liking, by simply looking them up in a table. While that may result in passing the test, one would not define that behaviour as intelligent. Searl's critique goes a little further, arguing that the processing of information by a simple rule book could not be regarded as intelligent either. While providing reasonable arguments, both fail to define what intelligence is in the first place. Current work [85], [86] tries to answer the question by distinguishing AI in weak and strong sub-classes. A weak AI includes any program that fulfils its task correctly. Stating that every kind of logic is basically intelligence, weak AI focuses on successfully solving tasks. Strong AI on the other hand mimics real-world systems, i.e. machines with consciousness and intelligence similar to human beings. While AI has seen a huge rise in popularity due to the development of

cheap high-performaning Computational Processing Unit (CPU)s and Graphics Processing Unit (GPU)s, today's systems are still far away from being a strong AI [87]. These systems have, however, already become a fundamental part of our daily life. Self-driving cars, image processing or speech recognition have taken a huge leap forward, based on the concepts of Artificial and Convolutional Neural Networks. Possible applications also extend over all fields of science. Here, AI will be used to dig deeper into the system, from which its concepts originate and what it ultimately tries to mimic: The biology of living beings.
This chapter will start out with a description of the perceptron, the artificial counterpart to the biological neuron. Reasonable functions for activating these perceptrons will be shown. It will be demonstrated how multiple perceptrons can be stacked together to ANN and how these learn through back-propagation and loss-functions. The basics will be completed by regularization techniques, that prevent vanishing or exploding gradients in very deep neural networks. The chapter will finish off with a brief history of Convolutional Neural Networks (CNN) and state-of-the-art architectures, channelling the flow of information through the introduced components.

3.4.1 Perceptrons

One of the fundamental concepts of machine learning is the perceptron. It is the artificial counterpart to the neuron, that makes up our brain. Both neurons are shown in Figure 25. The idea dates back to the work of McCulloch and Pitts in 1943 [88]. Observing the propagation of electrical signals in the nervous system, the following key components of a neuron can be identified:

- Dendrites channel the signal into the cell body (soma), which can be seen as the processing unit.
- If the electrical excitation surpasses a certain threshold, the neuron fires in an "all or nothing" manner.
- The electrical signal flows through the axon until it reaches a synapse, the contact side to another neuron.
- The signal is transferred over the synaptic gap via a conducting fluid.

The perceptron adopts these components by taking a vector $\mathbf{x} = (x_1, ..., x_n)^T$ with n denoting the number of inputs. The values are processed by a function f representing the soma. In ANNs f is called the activation function and its choice heavily affects the performance of the network. The synapse is represented by a weight vector $\mathbf{w}$ reflecting the conductivity of the synaptic gap. The axon corresponds to the output y of the activation function. All in all the formalism adds up to [87]:

$$y = f(\mathbf{x} \cdot \mathbf{w}). \tag{89}$$

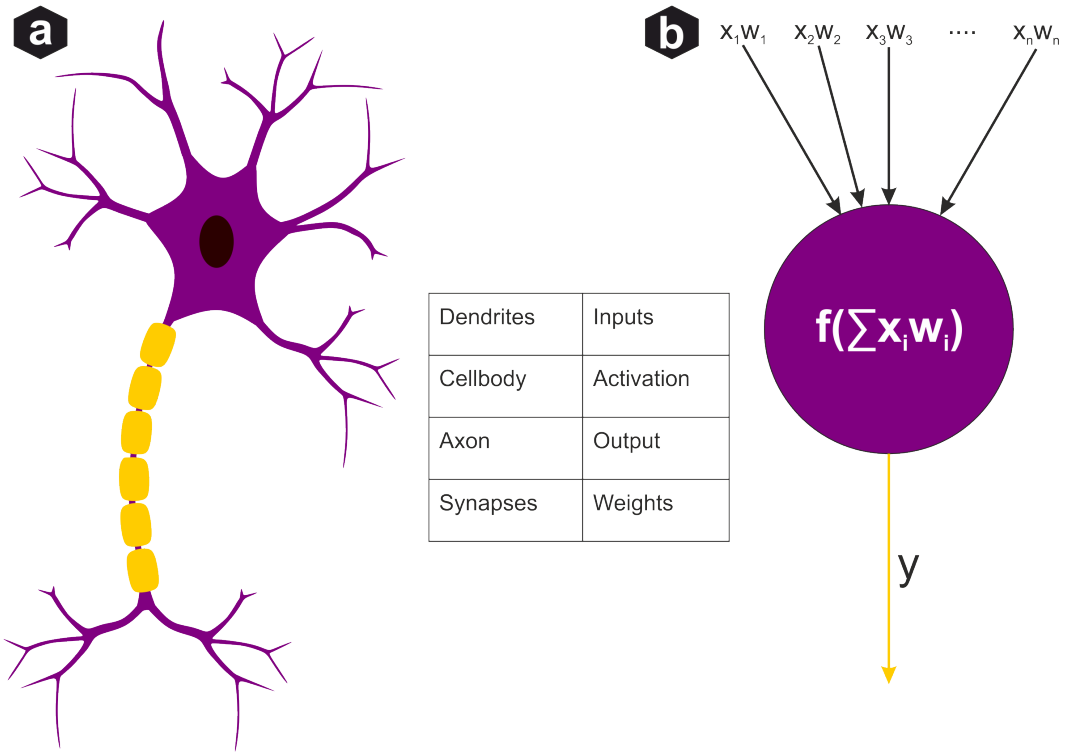

Figure 25: **Comparison of artificial and biological neurons** The Perceptron (b) receives a set of inputs, corresponding to the dendrites in a biological neuron (a). The strength of the input signal, biologically defined by the conductivity of the synaptic gap, is simulated with the multiplication of a weight variable in the artificial neuron. The cell body and the activation function define the response to the inputs respectively, passing it to the axon or output of the neuron.

Training the network with one of the optimizers described in section 3.2.7 is straightforward. The only constraint is that f has to be a differentiable function since all optimizers work with gradients.

3.4.2 Activation functions

Having a formalism for an artificial neuron, the next challenge is to find a suitable activation function f for a given problem. In general, it is a good idea to select a function that includes non-linearities. Fitting high dimensional functions, these non-linearities enable the network to perform a complex regression of the data. [89] Considering deeper ANNs, the multiplication of successive derivations can lead to vanishing gradients for values smaller one, or exploding gradients for values larger one. The task of keeping gradients within certain limits is, amongst others, also tackled by the choice of an appropriate activation function [90].
The following activation functions are commonly used in current ANNs, some of them are shown in Figure 26 as well:

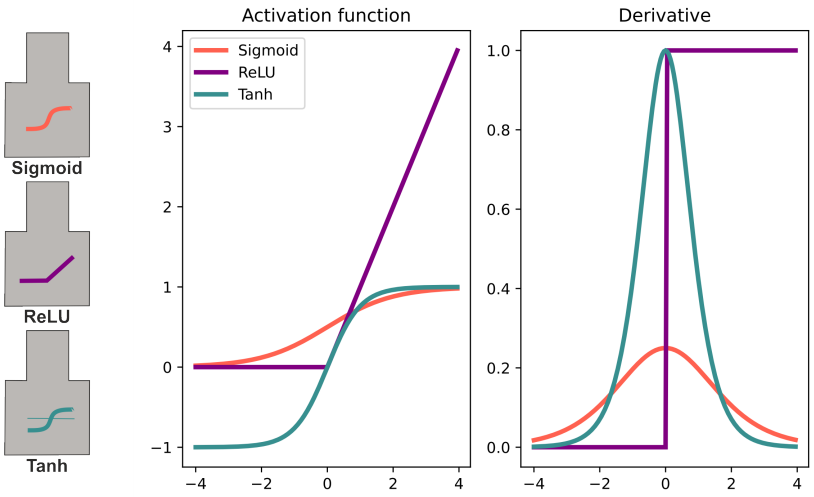

Figure 26: **Activation functions.** The commonly used activation functions are Sigmoid, Tanh and ReLU, as well as their visual representation in the following chapters (left) and their derivative (right). It can be seen that Sigmoid and Tanh activations saturate for large negative and positive values leading to vanishing gradients. ReLU activations saturate for values below zero.

- **Sigmoid:** $f(x) = \frac{1}{1+e^{-x}}$ is differentiable in $x \in \mathbb{R}$. It maps outputs from $f(x) \in [0, 1]$. For large positive or negative values, sigmoid activations can lead to a vanishing gradient.
- **ReLU:** [91] $f(x) = \max(0, x)$ is the most common activation function for an ANN. It is computationally very cheap and provides simple derivatives. The latter can be important for very deep neural networks, where a lot of functions have to be derived by the chain rule. Computationally, the derivative at 0 is set to an arbitrary constant value $\in [0, 1]$.
- **LeakyReLU:** [92] $f(x) = \max(ax, x)$ with $a < 1$ is a variation of ReLU that provides a small gradient for $x < 0$. The activation has, therefore, no saturation preventing vanishing gradients.
- **Softmax:** $f(x) = \frac{e_i^x}{\sum_{i=0}^{n} e_i^x}$, where i is the current input and n the number of inputs. Softmax is suitable for multi-class classification tasks, highlighting the output with the highest probability and ensuring that the outputs add up to one.
- **Tanh:** $f(x) = \tanh(x)$ maps outputs from $f(x) \in [-1, 1]$. In contrast to Sigmoid, it is zero-centred. Pushing the inputs of the consecutive layer closer to zero, these are more likely to fall into a regime of larger gradients, leading to improved learning. Similar to Sigmoid saturation can also lead to a vanishing gradient problem.

For hidden layers in a deep ANN, ReLU or LeakyReLU is almost always the best choice. The computation is much cheaper since no exponential functions have to be

computed. The non-saturating activation helps the network to converge much faster. ReLU provides sparse activation with a cutoff at $x < 0$ leading to efficient learning [93]. The other listed activations find their application in the output layers mapping values to a certain range. Sigmoid is optimal to model a probability. Softmax channels all values into a distribution adding up to one. Tanh maps to a range of positive and negative values, that can i.e. be used to indicate a limited positional offset.

3.4.3 Loss functions

A loss function is responsible for determining how well the output of a neural network fits the training data. It is comparable with the residuals of linear regression, but instead of taking the absolute distance l_1, other metrics are applied, which can be adjusted according to the problem. The mean squared error $\frac{1}{n}l_2$ for example results in faster convergence of the network since gradients increase/ decrease by a power law. Fitting a probability distribution $p(x)$ to a dataset, it is beneficial to minimize the negative log-likelihood: $\min\left[-\log(p(x))\right]$. Measuring the difference of two probability distributions Q, P the Kullback-Leibler divergence $\sum_{i=1}^{n} P(x_i)\,\log\left(\frac{P(x_i)}{Q(x_i)}\right)$ [94] is a good measurement.

3.4.4 Artificial Neural Networks

Artificial Neural Networks [95][96] (Fig. 27) are composed of several Perceptrons. A certain amount of these Perceptrons work in parallel, defining a structure which is called a layer. Layers are stacked above each other and their work is temporally separated. In the case of fully connected or dense layers, each Perceptron is connected to all Perceptrons of consecutive layers. The network is trained as a whole in two steps: forward pass and backpropagation. At initialization weights and biases are initialized randomly. In the forward pass, training data is channeled trough the input layer into the hidden layers. The last layer is an output layer o_i. It can be seen as the collector bringing the unordered information, which the Perceptrons computed up till now, together. The final output is fed to the loss function (cf. Section 3.4.3) comparing the output of the network with the target vector of the training data. The difference, the loss L of the forward pass, is used by the optimizer (cf. Section 3.2.7) to compute a gradient with respect to the weights w_{ji} within the network. Treating the consecutive layers of the network as a chain of functions, the chain rule can be applied to compute the corresponding gradients for each weight:

$$\frac{\delta L}{\delta w_{ji}} = \frac{\delta L}{\delta o_i}\frac{\delta o_i}{\delta w_{ji}}. \tag{90}$$

Taking a certain step size in the gradient direction, the weights are updated to minimize

the loss in the next forward pass. This optimization process is called backpropagation [97].

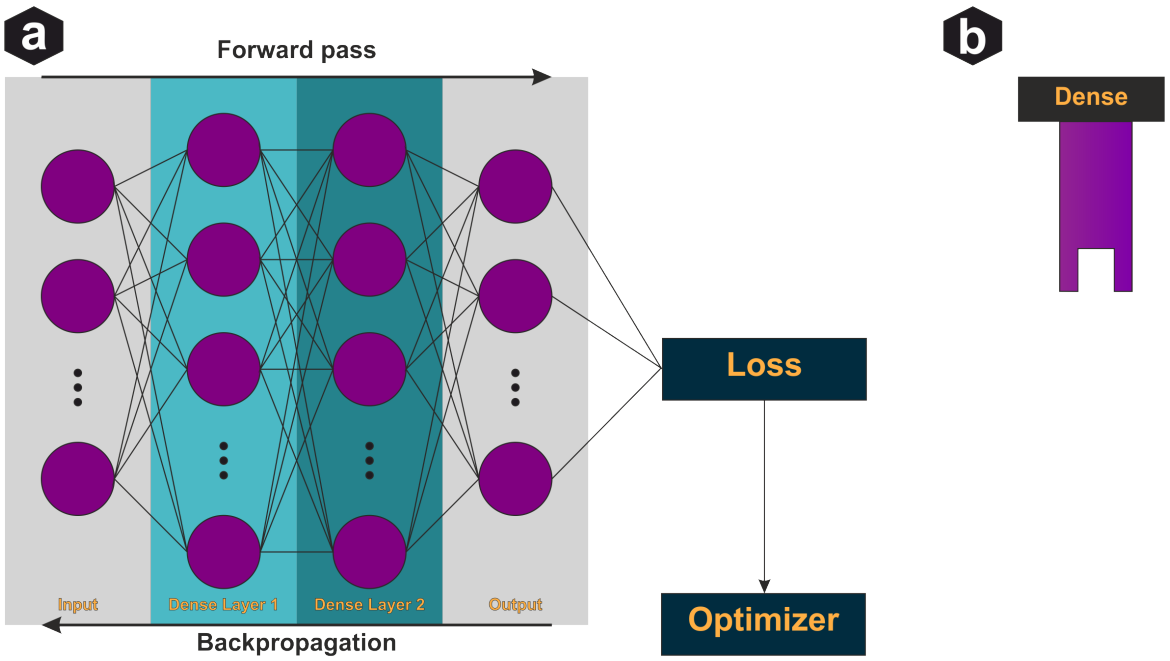

Figure 27: **Four layer artificial neural network with two hidden layers** In the forward pass each input is processed by a neuron and subsequently passed in the fully connected dense layers 1 and 2. The output layer collects the final representation from dense layer 2. The loss function compares the output to the target vector. In the backpropagation, the difference between the target vector and output is used to compute a gradient g with respect to the weights w_{ij}. The optimizer uses this gradient to update the weights for the next training iteration, minimizing the error.

3.4.5 Regularization

Creating deeper and larger ANNs, the learning process becomes more difficult. Dowat et al. [98] give a nice explanation comparing the backpropagation of the network chain to the game telephone. When whispering a message from one person to the next, slight misunderstandings propagate and add up, leading to substantial changes in the original wording. The more people in the chain, the more messed up the message gets. In ANNs this phenomenon causes, amongst others, internal covariate shift [99], the drift of output distributions of internal layers towards the saturation regime of the used activation functions. In order to stabilize the training process, it is, therefore, important to intervene at certain points. Multiple regularization techniques have been developed to target this stabilization.

Local response normalization [100] creates competition for kernel activations, reinforcing the value of the strongest, while keeping other values low. This prevents adjacent

kernels from learning similar features, i.e. it enforces locally sparse activation. The kernel update rule is given by

$$b^i_{x,y} = \frac{a^i_{x,y}}{k + \alpha \sum_{j=-n/2}^{n/2} a^j_{x,y}}. \tag{91}$$

Here, the activation of the i-th kernel $a^i_{x,y}$ at position x, y is normalized by the activations of the next $n/2$ adjacent kernels. The constants k and α provide protection against division by zero and a scaling factor, respectively.
Auxiliary loss is a technique that adds an output in the middle layers and computes the difference to the target vector. It is like getting a sneak peek into the network at a certain point and making sure the training is on track.
Another method to enforce sparse activation was developed by Hinton et al. and is called Dropout [101]. Dropout simply deactivates a stochastically selected subset of kernels, forcing neurons to act independently. Dependent activations would statistically fail more often, resulting in a higher loss.
Taking a look at the Sigmoid function's derivative in Figure 26 it is obvious why internal covariate shift leads to vanishing gradients in subsequent activations. However, it was shown that centring outputs around zero is also beneficial for the learning process of ReLU-activated networks. The corresponding regularization technique is called Batch normalization [102]. Here, the outputs of a batch of neurons are normalized with the mean value μ_B and variance σ^2_B of the batch, resulting in a distribution with a mean value of zero and variance of one:

$$x_i = \frac{x_i - \mu_B}{\sqrt{\sigma^2_B + \epsilon}}, \tag{92}$$

where ϵ is a constant to prevent division by zero. The normalized values of x_i are further processed with a learnable linear transformation with variables γ and β:

$$y_i = \gamma x_i + \beta. \tag{93}$$

The normalisation is shown in Figure 28 as well as the Symbol used for this technique.

3.4.6 Convolutional Neural Networks

Being able to work with high dimensional features and information, ANNs are a perfect candidate for image recognition tasks. Using fully connected layers on pixel data is, however, computationally very expensive. Connecting two layers for 100x100 pixels would already result in 50005000 connections with the same amount of weights. The classical approach to circumvent this exploding cost of resources is to first extract

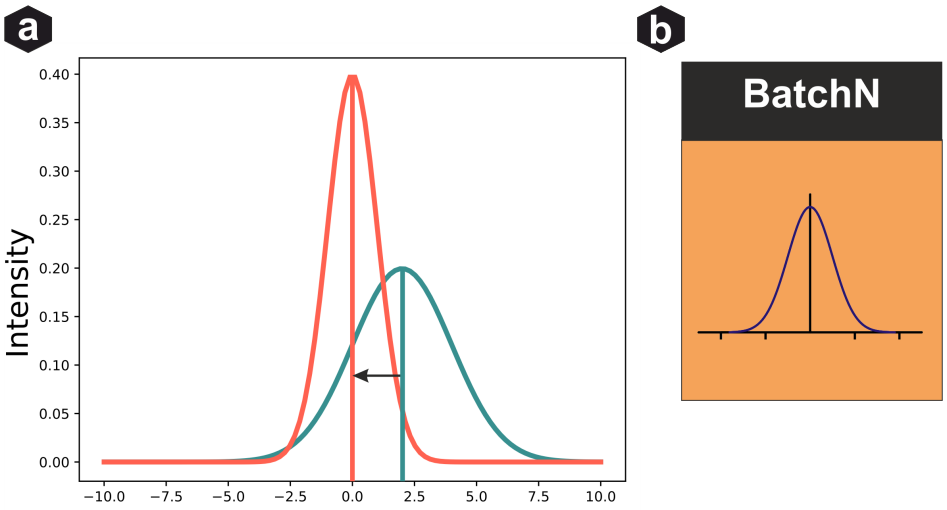

Figure 28: **Batch normalization** a) Diverging activations (cyan) are normalized to a distribution with zero mean and standard deviation 1 (red). b) Symbol used for Batch normalization in subsequent network models.

information from the underlying images with a feature extractor. It contains a set of convolutions, which is usually engineered by hand to solve a specific problem and hence not generalizable. Le Cun et al. integrated the convolutional operation into the network, making the feature extractor a trainable compartment. Their work, LeNet [103], was one of the first CNNs. It consists of two consecutive convolutional layers, where the values of the underlying filter kernels are learnable weights. These are followed by two fully connected layers and one Gaussian-connected layer. The convolution operation has the advantage that it is invariant to the spatial position of a feature, which is of little importance in classification tasks. Considering a sample of the handwritten letter T, it does not matter whether the horizontal line starts at the 5th or 8th pixel from the top. However, the network has to recognize the presence, length and relative position as well as its orientation to other features.

Achieving outstanding performances in text recognition, CNNs grew in popularity and their usage expanded to more complex applications. Detecting objects in image data was one of those. The CNNs, growing larger and deeper, suffered from vanishing gradients and overfitting. Considering that the ImageNet LSVRC-2010 challenge already provided a dataset of 10 000 000 images of more than 10 000 categories, this is hardly astonishing. Krizhevsky et al. managed, however, to resolve some of the mentioned problems with their work AlexNet [100]. AlexNet is a very deep CNN containing 650 000 neurons in five convolutional and three dense layers. To train a network of such size, two GPUs were used. Additionally, the - to that date uncommon - activation function ReLU was implemented (cf. 3.4.2). ReLU helped to keep the vanishing gradients in check and the computational effort comparably low.

The problem of overfitting was targeted by using Dropout (cf. 3.4.5) during the training. Discarding a certain percentage of the random kernels in each training encourages

the network to learn sparse activations for independent features. All in all the network was able to diminish the error rate from 26.2% to 15.3%, significantly improving the state of the art in image classification.

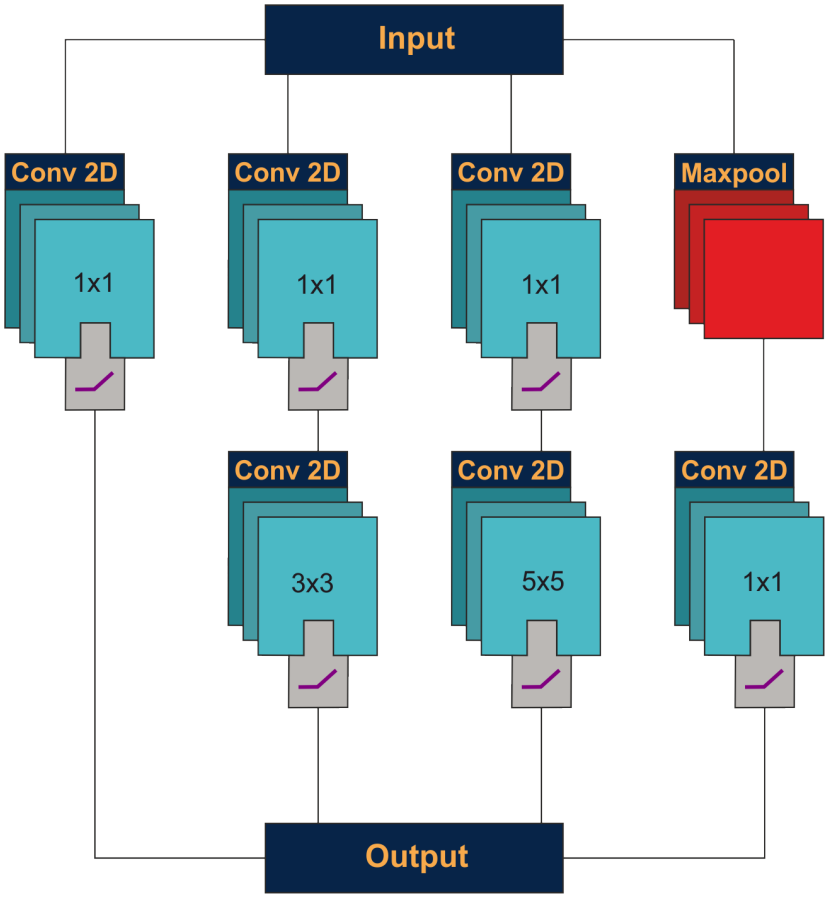

Figure 29: **Inception architecture** Inception V1 building block. Inputs are processed in 4 different paths. 1x1, 3x3, 5x5 convolution and max-pooling are computed in parallel. The outputs are concatenated back together. To keep the filter sizes from exploding, the inputs are compressed prior to the 3x3 and 5x5 convolutions and after the max-pooling.

In one of the subsequent ImageNet classification challenges (LSVRC-2014), GoogLeNet [104] set a new benchmark. Instead of increasing the complexity of the network by adding additional layers and kernels, Szegedy et al. invented the Inception architecture (Figure 29), a network in the network. The inception building block performs three convolutions with increasing kernel size (1x1,3x3,5x5) and a max-pooling operation in parallel. The outputs are subsequently concatenated back together. The key idea is to approximate a sparse structure in neural networks with dense components. Inception realizes this by performing ReLU-activated 1x1 convolutions before 3x3 and 5x5 convolutions as well as after the max-pooling. These 1x1 convolutions compress information by reducing the dimensions of the filters, further encouraging sparse activation of feature detectors. Additionally, this allows a nine-fold reduction of parameters compared to AlexNet. The whole Inception building block is repeated multiple times throughout

the network. Though, information is collected in various stages and scales.
The architecture was further enhanced in Inception V3 [105]. It was shown that larger convolutional filters like 5x5 can be replaced with two smaller ones like 3x3 reducing the number of parameters and increasing computational speed significantly. The concept was further refined by introducing asymmetric filters, i.e. replacing a 3x3 convolution with two subsequent 1x3 and 3x1 convolutions (in middle layers 12-20) and using Batch Normalization.
While AlexNet achieved optimal performance with a depth of 5 layers, GoogLeNet managed to use 22 layers to reach its full potential. However, despite the introduced regularization techniques there still seems to be a maximum depth after which networks stop improving. At this point, shallower networks still outperform deeper ones, indicating that vanishing gradients still lead to problems. ResNet (Fig. 30) [106] tackles that problem by introducing a so-called identity shortcut. The identity shortcut skips convolutions every few layers (denoted by F) adding the outputs element-wise:

$$H(x) = F(x) + x \tag{94}$$

The additional term shortens the gradient chain, making it easier for updates to propagate deep into the middle layers of the network. Using this architecture with shortcuts through the entire network, deeper networks are expected to outperform shallower ones, since the information of what can be called the shallower subnetwork can flow freely to the end of the network. In fact, He et al. [107] managed to train a network with 1100 layers that outperform its 110 layers counterpart. The authors also noted that any kind of modulation on the skip connection impedes the learning rate of the network substantially. Multiplying a constant factor λ to the backpropagation of a network with n layers, the derivation of the l-th layer adds up to:

$$\frac{\delta\epsilon}{\delta x_l} = \frac{\delta\epsilon}{\delta x_n}\left(\lambda^{n-l-1} + \frac{\delta}{\delta x_l}\sum_{i=l}^{n-1}\lambda^{n-i-2}F(x_i)\right). \tag{95}$$

One can immediately see that for large n and small l, i.e. earlier layers in a deep network, the term for the identity shortcut tends to vanish for $\lambda < 1$ or explode for $\lambda > 1$. This forces gradients to propagate over the layers, losing the advantages of the shortcut.

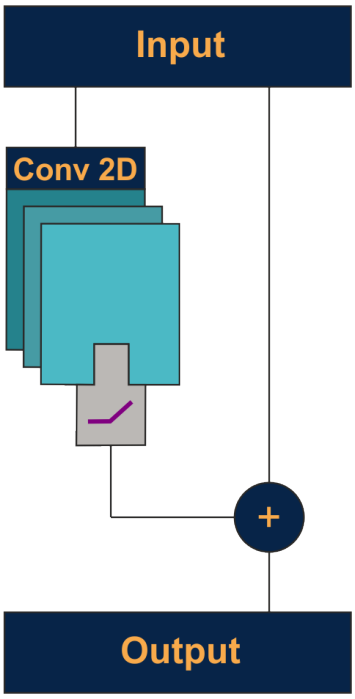

Figure 30: **ResNet** The processing of the network is skipped every view layers with an identity shortcut.

3.4.7 From discriminating to generating models

The previously described networks focus on the assignment of images to predefined classes, i.e. discriminating between given categories. However, CNNs are also able to generate completely new images from a given training. The Variational Auto Encoder (VAE) [108] is similar to a normal auto encoder trained to estimate so-called latent variables $\mathbf{z}$, e.g. a variable that describes the dataset $\mathbf{x}$ but is not directly accessible. Therefore, the network uses an encoder p that translated the data $\boldsymbol{x}$ into latent space and a decoder q to transform from latent space back to data space. The difference between input $\boldsymbol{x}$ and output $q(p(\boldsymbol{x}))$ can be used to train the network. The VAE further constrains the latent representation $\boldsymbol{z}$ to be as close to a normal distribution, with zero mean and variance one, as possible:

$$\mathcal{N}(0, I), \tag{96}$$

where I denotes the identity matrix. The KL-Divergence between the normal distribution and $\boldsymbol{z}$ is added to the loss term. This improves the representation of $\boldsymbol{x}$ in $\boldsymbol{z}$ and allows the sampling of $\boldsymbol{z}$ vectors that were not covered within the training to generate new data. Similar to other neural networks VAE can be optimized with SGD.
The big advantage of VAEs is, that certain features can be parameterized into the latent variable. Shifting that latent variable in a certain direction enhances the desired feature. White et al. [109] were able to model the expression in the faces of input images by tuning the latent variable. Faces with neutral expressions were tuned to sad or smiling.
Another generative approach is the one of Generative Adversarial Network (GAN) [110]. GANs contain two independent network architectures, a generator and a discriminator. While the generator builds images from random noise, the discriminator tries to distinguish "real images" from artificially generated ones. The loss of the generator is computed with the accuracy of the discriminator. This leads to a competition of both networks, ideally resulting in an accuracy of 50%, indicating that ground truth and artificial data are indistinguishable. Ronneberger et. al. target the task of image segmentation with their architecture U-Net (Figure 31) [111]. A fully convolutional network that produces output images in the same dimensions as the input images, classifying each pixel into a category. The key idea is to extract and compress features from the original image by max-pooling in a downsampling step. The sparse information is then used in the subsequent upsampling to generate a new image with a segmentation. The upsampling is supported by concatenating the results of the downsampling step to the corresponding layer in the upsampling.

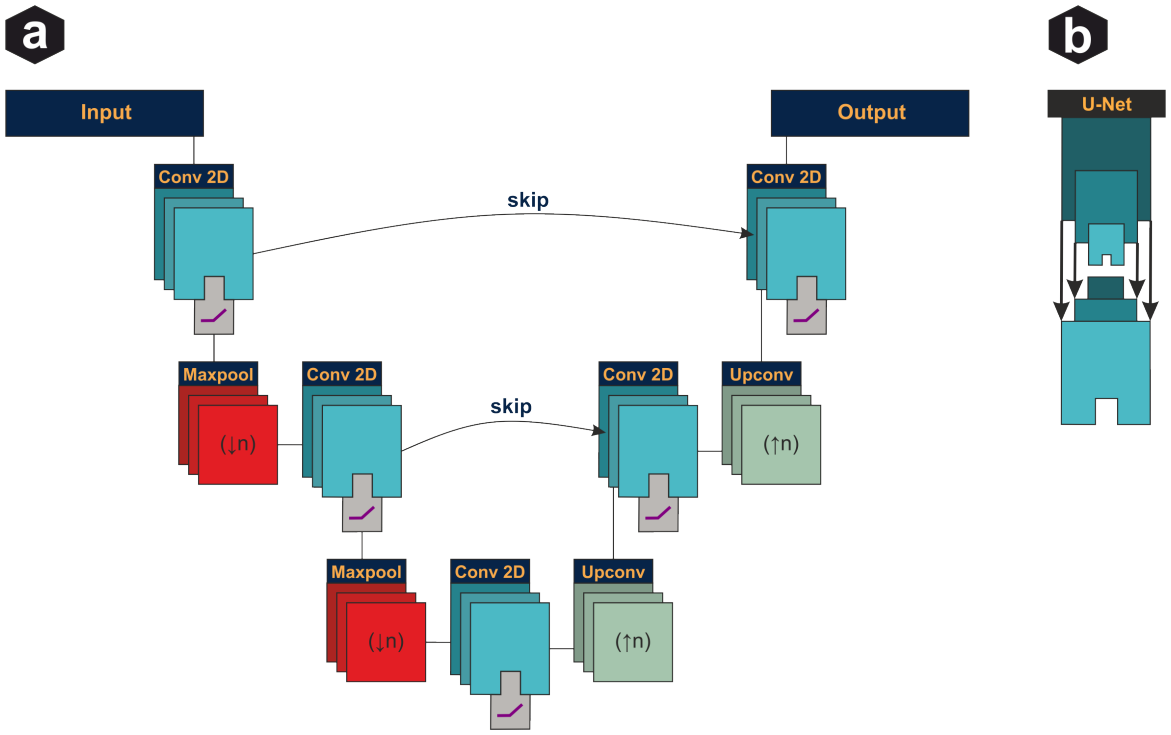

Figure 31: **U-Net concept.** a) U-Net is an image-to-image network that shows peak performance for segmentation tasks. Input images are downsampled by a combination of convolutional and max-pooling layers, spatially compressing information. Subsequently, the information is upsampled again to build a new image. Concatenating the corresponding slides of the downsampling path supports the process. The output image equals the input image in terms of dimensions. b) Simplified symbol for further usage.

4 Projects

This chapter will explain the key aspects and implementation of the three main projects of this thesis. **`LineProfiler`** is a software to objectively evaluate filamentous structures and was used to examine the quality of expanded microtubule and synaptonemal complex data. **`Automated Simple Elastix`** implements advanced image registration to determine expansion factors and distortions of ExM samples. **`Recursive Compressed Sensing Artificial Intelligence (ReCSAI)`** uses AI to enhance Confocal Lifetime SMLM by learning to reconstruct precise localisations from non-linear PSFs.

4.1 LineProfiler

Evaluating the quality and resolution of fluorescence microscopy images is challenging. The measurement depends on several factors, such as imaging modality, labelling density and sample preparation. To obtain comparable data, scientists often use well-known reference structures. Corresponding benchmarks are created by a theoretical model or electron microscopy. Common examples of these structures are microtubules. Microtubules are hollow filamentous structures and part of the cytoskeleton of eukaryotic cells. The tube has a diameter of 25 nm and is formed by two entangled helical

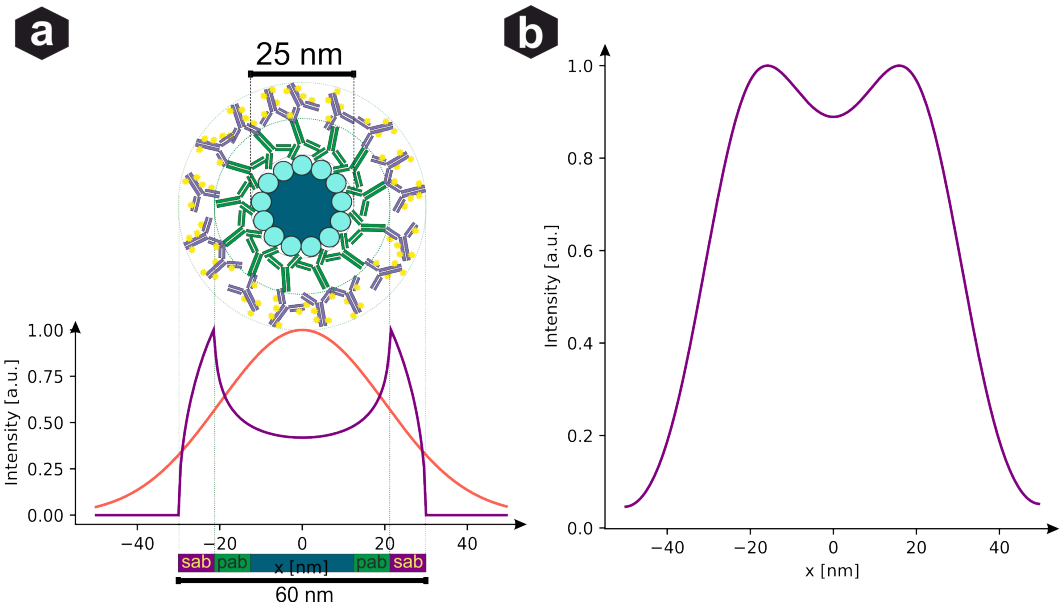

Figure 32: **Microtubule** a) Structure and intensity profile of microtubules labelled with primary (green) and secondary (purple) antibodies. Microtubules have a diameter of 25 nm. Antibodies have a size of ~ 8.75 nm. Therefore, the fluorescent molecules are distributed on a cylinder with an inner diameter of 43.5 nm and an outer diameter of 60 nm, projected on two dimensions. This yields the intensity profile shown in purple. Assuming emitters can be localised with a localisation precision of $\sigma = 10$ nm (red profile), the measured intensity distribution resembles the one shown in b).

sub-units α- and β-tubulin. Labelling the structure with primary and secondary antibodies results in a cylindrical fluorescent signal with inner radius 21.25 nm and outer radius 30 nm [112]. Considering a lateral cut x, y through a microtubule filament as shown in Figure 32 the intensity profile can be estimated with the analytic description of a circle: $r^2 = x^2 + y^2$ and therefore $y = \sqrt{r^2 - x^2}$. We assume emitters to be distributed homogeneously over the area covered by the secondary antibody. The corresponding equation formulates as:

$$I(x) = \begin{cases} 0 & \text{for} \quad |x| > r_2 \\ 2\sqrt{r_2^2 - x^2} & \text{for} \quad |x| < r_2 \ \ \& \ \ |x| > r_1 \\ 2(\sqrt{r_2^2 - x^2} - \sqrt{r_1^2 - x^2}) & \text{for} \quad |x| < r_1 \end{cases} \tag{97}$$

Here, $|x| > r_2$ describes the region in x outside of the labelled antibodies which yield no fluorescent signal. $|x| < r_2 \ \ \& \ \ |x| > r_1$ describes an x region where the cylinder volume is completely filled with fluorescent dyes, while $|x| < r_1$ covers the hollow part of the cylinder, where the signal originates from the fluorescent volume above and bellow the filament.

In real measurements, this theoretic formula is blurred with a Gaussian function $G(\sigma)$. Therefore, the resulting intensity is given by

$$J(x) = I(x) * G(\sigma). \tag{98}$$

The corresponding σ equals the localisation precision of the emitter. With decreasing precision, the sharp peaks get more and more blurry, as can be seen in Figure 32 b, until the dip vanishes and only one peak is left.
Estimating the parameters of Equation 98 for a cross-sectional microtubule profile allows, in theory, direct conclusions about the resolution. However, real data, being afflicted by noise and inconsistent labelling is often more difficult. Thus, several cross-sectional profiles have to be taken along the filament to produce interpretable data. Hand-picked regions over small sections of the image are, however, willingly or unwillingly biased by the scientist. To target this problem, we developed `LineProfiler`, a software that fits filament-like structures to a Cubic-spline (C-spline) and averages line profiles from all over the image. The project was divided into the following four components:
(i) Data Collection: The key idea is to describe the flow of a filamentous structure with an analytical term and follow its course, taking perpendicular profiles in predefined intervals. A suitable tool for interpolating noisy data is the C-spline. C-splines fit a n-dimensional polynomial to a set of points. If n is equal to the number of points, the data can be perfectly fitted and all points are on the analytical line. Picking a smaller n applies a certain degree of smoothing. Since the C-spline is an n-dimensional polynomial, picking the derivative and rotating the value by 90° yields the direction of the desired line profile. Automatically gathering these profiles from all filaments in the image simplifies the workflow and provides a huge amount of objectively collected data for further evaluation.
(ii) Data evaluation: The gathered line profiles are subsequently fitted to a set of selected functions: either to a bi-Gaussian distribution, that yields the peak-to-peak distance and can be used to calculate the microtubule diameter or to the cylinder function (c.f. Equation 98), that directly describes the emitter distribution along the filament.
(iii) Software packaging: In the third step the described concepts are packaged into a performant and maintainable software solution. The finished software features include a Graphical User Interface (GUI), threads, coroutines and a maintainable structure.
(iv) Application: In the last step the software is tested in real applications. We used `LineProfiler` to evaluate expanded Synaptonemal Complex (SC) and microtubule datasets.

4.1.1 Data collection

C-splines are interpolated from a set of data points. Given an image containing filamentous structures, it is necessary to bin these extended structures down to coordinates. As explained in section 3.2.3 (nonlinear Filters) this can be achieved by converting the image into a binary format and reducing the resulting set of pixels to a subset

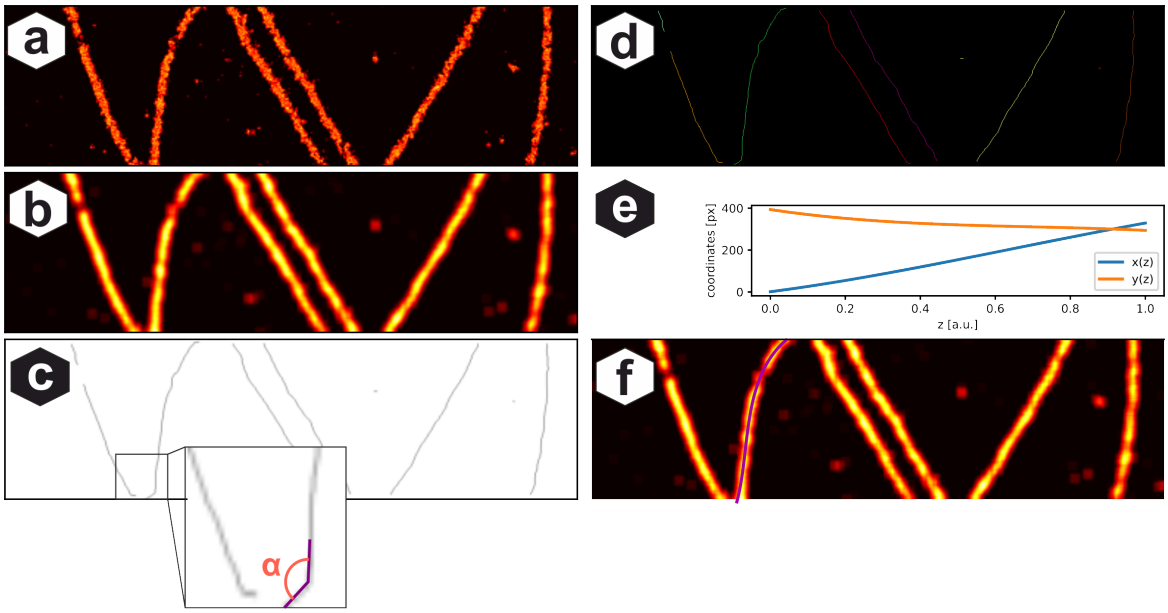

Figure 33: **`LineProfiler` data collection** a) Sample image of post-labelled expanded microtubules. The data was collected under *d*STORM microscopy and proExM expansion b) Image blurred with kernel size 20. c) Skeletonized image and line continuity check. If $\alpha < 90°$ the line is separated at this point. d) Labeled lines: each colour corresponds to one connected structure e) Reparametrization and smoothing along the filament. f) Spline for one filament (purple) plotted into the blurred image.

that sufficiently describes the structure of the line. A suitable algorithm for this task is the skeletonize algorithm, which erodes two-dimensional binary shapes until only one-dimensional lines remain. The coordinates of the remaining pixels can be used for the subsequent interpolations.
In Python, there are already packages that provide performant implementations for most image-processing algorithms. The most important ones for the implementation of the `LineProfiler` algorithm are `skimage` [113] and `pyopencv` [114], `numpy` [115] and `PyQt` [116]. A full list of all packages can be found in the requirements file of `LineProfilers` Github.

First, image data has to be converted to a set of discrete and neighbouring pixel coordinates e.g. points. To compensate for bad labelling efficiency or holes in the structure, the image is convolved with a two-dimensional blur (Figure 33 b). Appropriate kernel size values depend on parameters such as the pixel size and labelling quality and typically range from 5 px in well-labelled SIM images to 50 px in corroded expanded localisation data. The image is converted to binary values by applying the Otsu thresholding algorithm [117]. The result is skeletonized (Figure 33 c), to reduce the number of nonzero points in the image. This already gives a first estimation for the centre of the line but it lacks a smooth direction, which is necessary to collect perpendicular profiles.
In the second step, we use `sk-image`s label (Figure 33 d) to identify all coordinates

belonging to one consecutive line. The label function assigns an equal number to all pixels $p \neq 0$ that are either directly (connectivity=1) or diagonally (connectivity=2) connected. In the resulting feature map, all coordinates of connected components have the same unique value. While this separates unconnected structures, filaments crossing each other are not distinguished. Hence, these filaments have to be separated in an additional step, which we postpone to the coordinate collection of a line.
The first approach to finding these coordinates is to iterate over the maximum value of the label image and write coordinates with the current iteration value into a list. The `label` algorithm marks the background with zero. Hence, the label zero is excluded to prevent the algorithm from collecting the background. Starting with one, data points can be obtained by using the `numpy where` function. Scanning the image line by line for the correct values, yields the correct indices, but in an unsorted order. Considering a microtubule with the shape of a circle, two values would be collected for a line. These are, however, not neighbouring each other, resulting in a zig-zag interpolation. Therefore, an additional step, sorting the coordinate points is required. `Sk-learn` provides, with its `NearestNeighbor` class, easy access to a number of suitable sorting algorithms, automatically selecting the most appropriate one in dependency of the dataset. The resulting k-dimensional tree structure can be used to sort the list of data points from start to end along the line. Alternatively, the image foresting transform (cf. Section 3.2.3) could be used to select indices along the line, reducing run-time. This, however, does not represent a performance bottleneck in the software and results in only minor overall performance gains.
As previously mentioned, we also have to consider how to handle points where multiple filaments cross each other. We implemented a metric to exclude these points from the evaluation since we see no possibility to separate the superimposed signals. We assume filaments to follow a smooth line, i.e. not to entail sharp edges. Computing a rolling angle between two subsequent line segments of 15 coordinate points length enables us to estimate a probability for the breaking point between the lines to be a crossing point of multiple filaments. If the direction change exceeds 90°, the line is separated into two filaments at this point.
The sorted coordinates of the filament enable us to describe it with a parameterized expression. The C-spline is described by piecewise polynomials of the third degree:

$$\mathrm{spl}(z) = \sum_{k=0}^{3} a_k z^k, \tag{99}$$

where z is the spline coordinate space for the data points $[p_n, p_{n+1}]$ and a_k the k-th prefactor. The overall spline is given by the sum of all piecewise functions $\mathrm{spl}(z) = \sum_n \mathrm{spl}(z, n)$. These functions have to satisfy four boundary conditions to make up

a cubic spline. Value, first derivative and second derivative have to be equal at the intersection and the second derivative has to be zero:

$$\begin{aligned}
\mathrm{spl}(z,n) &= \mathrm{spl}(z,n+1) \\
\frac{d}{dz}\mathrm{spl}(z,n) &= \frac{d}{dz}\mathrm{spl}(z,n+1) \\
\frac{d^2}{dz^2}\mathrm{spl}(z,n) &= \frac{d^2}{dz^2}\mathrm{spl}(z,n+1) \\
\frac{d^2}{dz^2}\mathrm{spl}(z,n) &= 0.
\end{aligned}$$

We create one spline for x and one for y coordinates denoted as $\mathrm{spl}_x(z)$ and $\mathrm{spl}_y(z)$ respectively (cf. Figure 33 e). Here, the spline coordinate space $z_n \in [0,1]$ describes a coordinate transform along the filament. I.e. $z = 0$ is the start of the filament, increasing z changes the position linearly along the filament and $z = 1$ marks its end. The initial coordinates are set to $x(0), y(0)$ and z increases by the cumulative euclidean distance from the start point to the n-th points, divided by $\max(z)$ to achieve normalization. Using the `scipy UnivariateSpline` class on x, y in dependency of z yields an interpolation of the given points. The smoothing factor s can be freely adjusted and should be in the order of magnitude of the standard deviation of the sample, multiplied with the number of data points. `Scipy` uses s to increase the number of spline knots until the condition

$$\sum_n (y_n - \mathrm{spl}_y(z_n))^2 \leq s \tag{100}$$

is satisfied. Note that the same condition holds for x_i.
Since splines are described by an analytic polynomial computing the derivative is straightforward. With two gradients $g_x = \frac{d\mathrm{spl}_x(z)}{dz}$ and $g_y = \frac{d\mathrm{spl}_y(z)}{dz}$ the angle of the filament in respect to a vertical line can be computed by

$$\beta = \arctan\left(\frac{g_y}{g_x}\right). \tag{101}$$

Rotating by $\gamma = \beta + \frac{\pi}{2}$ yields the desired perpendicular direction, with respect to the filament. Splines and gradients can be sampled arbitrarily, but we found one sample point per input coordinate to be a sufficient rate. The profiles at position z are collected along a line with center position $[\mathrm{spl}_x(z), \mathrm{spl}_y(z)]^T$ in direction γ. The length of the line depends on the sample and pixel size. Defining an arbitrary width w the starting point p is given by:

$$\begin{pmatrix} p_x \\ p_y \end{pmatrix} = \begin{pmatrix} \mathrm{spl}_x(z) + w\cos(\gamma) \\ \mathrm{spl}_y(z) + w\sin(\gamma) \end{pmatrix}. \tag{102}$$

An example for a profiled microtubule filament can be seen in Figure 35 a.
Values for the profile are taken by the `scipy` function `map_coordinates`, which performs spline interpolation for floating point coordinates. Similar to the one-dimensional line profile it is possible to create a 2D x-z projection along the filaments. This can be achieved by collecting a profile for each stack which is subsequently projected into an image with intensity distribution instead of a graph.
Collecting data for the SC is more challenging. The structure does not describe a line, but a helical shape projected on two dimensions (Figure 34 a). The required reference line can either be collected over another image channel, labelled with SYCP1N, which is positioned at the center of the helix (Figure 34 b) or by creating a `floodfill` image (Figure 34 c) of the helical structure and subsequently reducing it to a skeletonized line.

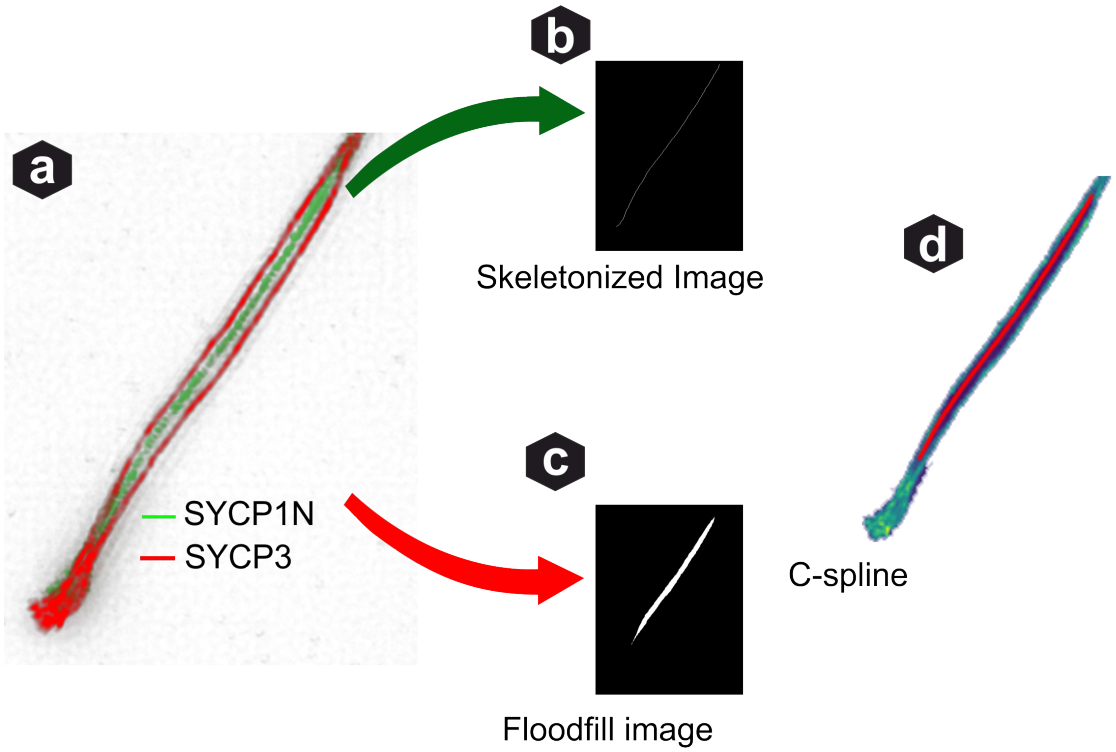

Figure 34: **Data collection SC** To profile the helical structure of SC (a) it is either necessary to provide another colour channel running in the centre of the helix (b) or to use a one channel evaluation, depending on a `floodfill` of the area the helix encapsulates (c).

4.1.2 Data evaluation

For the scientist evaluating the data, it is particularly important to see where and how the profiles are collected. As can be seen in Figure 35 b, we draw a coloured line for each profile into an image. Thereby, all profiles of one filament are rendered in the same colour. This provides an additional metric to perform a visual check of the algorithm's

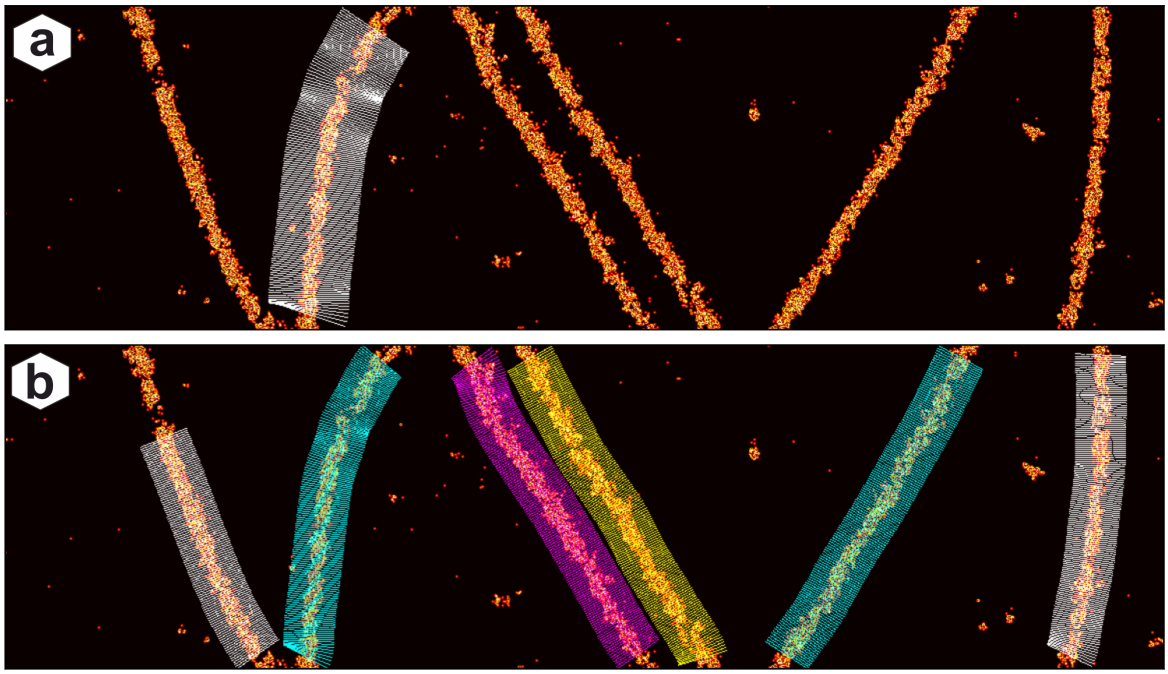

Figure 35: **`LineProfiler` profiles** a) Line profile of the strand used in the previous image. The sampling rate is bisected to get a better grasp of the line orientation and the underlying data. b) Evaluated image. Each connected filament is marked with a different colour. Profiles in the evaluation are also colour coded to simplify the assignment to the corresponding data.

performance or to identify filaments/regions of interest.

In the next step, we process the collected profiles. Here, we average the data for each filament. The plots are colour-coded corresponding to the profiles in the image, denoted with a number and saved in the data folder. The last plot averages the profile of all found filaments.

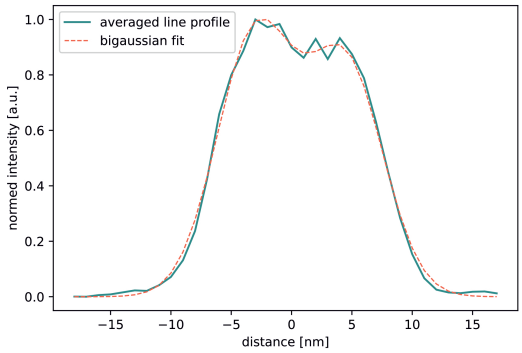

Figure 36: **Averaged profile and bi-Gaussian fit** Profiles are averaged from the filament shown in Figure 33.

To quantify data and image quality, the averaged profiles can be fitted with a set of functions. While the theoretical intensity distribution for microtubules has already been deduced in Equation 97, it is not always the optimal solution for a fit. The function contains non-linearities and a convolution. Therefore, it does not always converge reliably.

Possible alternatives are a Gaussian function

$$y(x) = Ie^{-\frac{(x-\mu)^2}{\sigma^2}} + b \qquad (103)$$

for good SNR and bad resolution, a bi-Gaussian, i.e. the sum of two Gaussians, for good resolution and good SNR, or a tri-Gaussian for bad SNR and good resolution.

For convenience, we use `scipy optimize` with a trust region reflective algorithm (similar to Levenberg-Marquardt second order optimization, cf. Section 3.2.7) for optimization. This optimizer is generally robust and allows constraining parameters with upper and lower bounds.
We require all parameters to be greater than zero. The intensity I is constrained with an upper bound of the data maximum value. Sigma σ remains unconstrained to the upper end. The centre μ is constrained to the coordinates covered by the line profile. As an initial guess, we estimate the following parameters: The intensity is estimated with $\max(\text{data})/m$, where m is the number of Gaussians used in the fit. Sigma is arbitrarily set to 0.5 as it does not play a big role in convergence. The centre μ of the Gaussian fit is estimated as the x-coordinate of the peak. The parameters are fit with a least square loss. An example of a bi-Gaussian fit is shown in Figure 36.

4.1.3 Software packaging

One of the key ideas of `LineProfiler` was to create a user-friendly, portable and performant software solution. Instead of the common IPython Notebooks or command line scripts we decided to build a GUI. For this purpose, we used the package `PyQt` [116] which comes with a convenient designer app. The components of the GUI can be assembled by drag and drop. Saving the design yields an `.ui` file, which can be converted to auto-generated Python code by running the provided `make_gui.bat` file. Specific functionality is added by inheritance of the GUI class, so changing and recompiling the design does not break the code.
The interface can be seen in Figure 37. It provides a data field, which can be used by drag and drop or by clicking the 'Open' button. Clicking on a file in the list and subsequently clicking 'Show selected' loads the data into memory. Checking a checkbox in the channel selection shows the corresponding data in the widget. In the 'config' tab the evaluation mode can be selected. Each mode comes with default parameters. These work empirically for most datasets, but they can be changed if necessary. The Look Up Table (LUT) should be adjusted to a value, where the structure is clearly visible and as continuous as possible. Note that changing the LUT does not change the values used in the evaluation. It is only used to estimate an appropriate spline profile.

Python is the preferred programming language for data science [118] and provides easy access to machine learning and the most common evaluation algorithms. Software with GUI is, however, rarely written in Python. One of the major reasons for this is, that the language is single-threaded, i.e. all of the code runs in a single process.

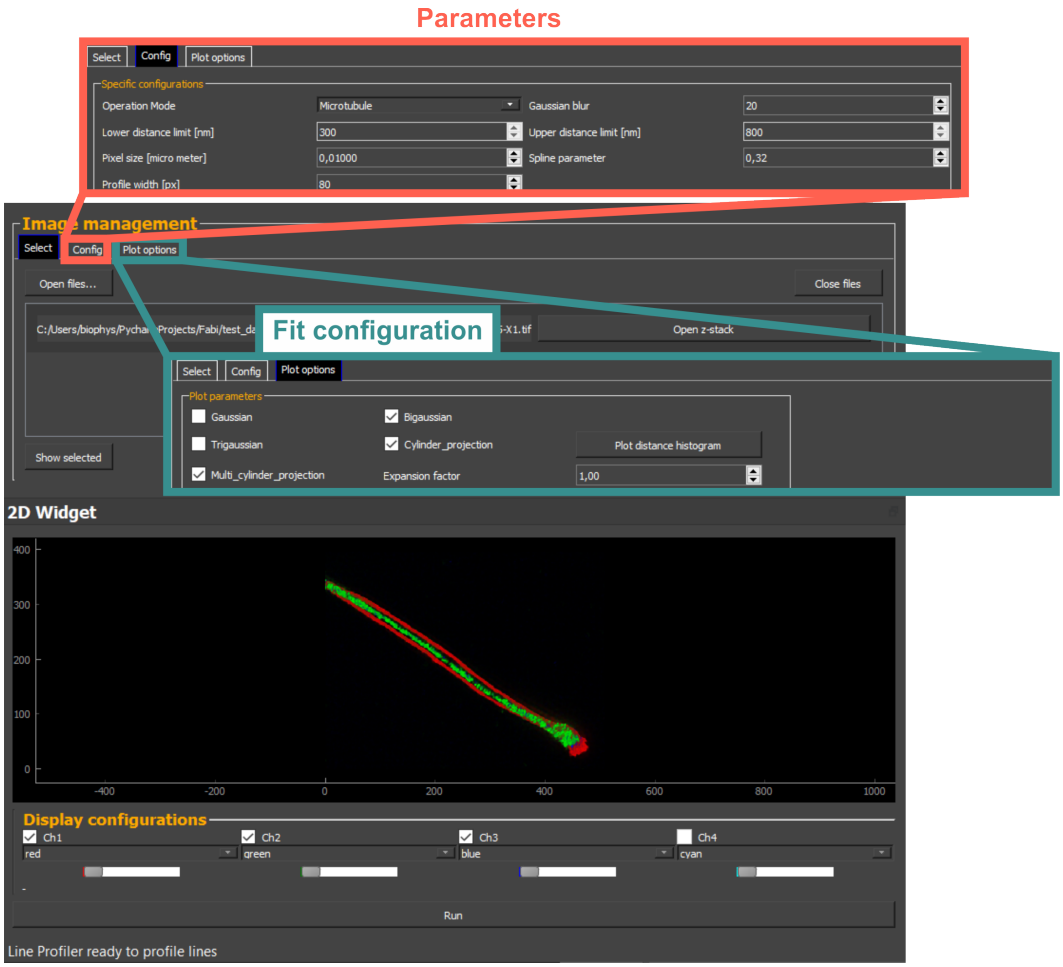

Figure 37: **Graphical User Interface** of the `LineProfiler` executable. Data can be loaded by dragging and dropping image files into the files widget. Clicking on a file and pressing 'show selected', shows the image channels with checked checkboxes in the 2D Widget. In the configuration tab, the evaluation configuration can be adapted to the selected file. Selecting the corresponding mode for the evaluated data provides suitable default parameters. 'Plot options' contains multiple checkboxes for functions that should be fitted to the data. Upon clicking 'run', an evaluation thread is started. Multiple evaluations can be started in parallel. The results are written in the data folder of `LineProfilers` current location. For each file, a folder with the file name is created.

But why does Python work on a single thread in the first place? This can be pinned down to the Python Global Interpreter Lock: "The Python Global Interpreter Lock or GIL, in simple words, is a mutex (or a lock) that allows only one thread to hold the control of the Python interpreter. This means that only one thread can be in a state of execution at any point in time." [119]
While this may seem counterintuitive in the first place, GIL was one of the reasons for the huge success of Python. It prevents race conditions and deadlocks in reference counting and garbage collecting and, therefore, allows the user to not care about memory management. This has, however, several disadvantages for designing a responsive evaluation software:

- Only one file can be evaluated at a time.
- The resources of the computer are not used efficiently, which slows down evaluation time significantly.
- The user interface is locked while processing, so it does not respond to user input and appears laggy.

Fortunately, there are some workarounds for using multiple threads in the same process. Qts `QThread` class can be used to start additional processes, which can communicate with the main process over signals. In `LineProfiler` we use these threads to start evaluation tasks while keeping the GUI in the main thread. This way the software stays responsive, while the computation happens in the background. This does not speed up the evaluation of a single file, but it allows the processing of multiple files with different evaluation parameters in parallel.
On clicking the ‘run‘ button, the current parameters are used to assemble a thread in the ‘process factory‘. Next to the parameters, the thread is connected with multiple signals that emit the collected data into microservices. These microservices are located in the main thread and perform the fitting and saving of data files upon receiving a signal that the thread's computation is finished.
The last point on the specification list is the portability of the software. By default Python software is not easily transferable to different systems, at least not for a user that is unfamiliar with Python programming. Running a Python-based program requires an installation of this language and either knowledge about the usage of the command line or a suitable IDE. By providing a ‘requirements‘ file the setup of the Python environment gets a bit easier. The used packages with the corresponding version number are capsuled into one file, which can be used to set up a suitable environment in one command.
However, we decided on the alternative to use the package `Pyinstaller` to compile the program into an executable file. `Pyinstaller` scans the code for all dependencies and hooks them into a single folder. Packed together the code can be ported from one system to another by copy and paste. Despite the extended size of the software package $\sim 500\,\mathrm{MB}$, we think that the additional portability out-weights the cost of additionally required system memory. The latest (executable) release of `LineProfiler` can be found under: https://github.com/super-resolution/lineprofiler.
Executables can be created from the source code by running the `main.spec` file, which includes some additional hooks for packages that `Pyinstaller` does not support by default.

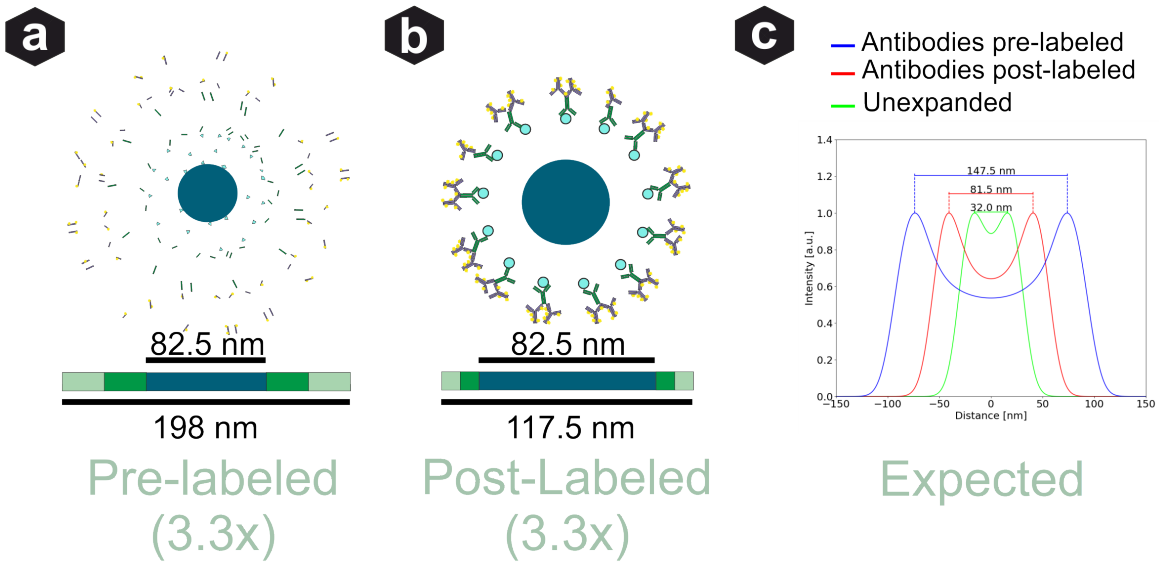

Figure 38: **Expanded microtubules** a) Pre-labeling: Primary and secondary antibodies are labelled before digestion and expansion. On digestion, the antibodies are torn apart and linked into the gel. The fluorescent signal is distributed over a large area and the linkage error is multiplied by the expansion factor. b) Post-labeling: Primary and secondary antibodies are labelled after digestion and expansion. Emitters are distributed over the 8.75 nm of the secondary antibody. The labelling error remains small. c) Predicted intensity profiles for un-expanded, pre-labelled and post-labelled microtubule filaments.

4.1.4 Application

Despite the major improvements SMLM yields for fluorescent microscopy images, the resolution is still limited. As shown in Section 2.4.2 emitters can only be localised with a certain precision. Since photons are emitted in a discrete statistical process, the 'real' distribution can only be determined with a precision defined by the CRLB. The CRLB is in approximation inversely proportional to the number of emitted photons and, therefore, dependent on the singlet-singlet transitions the used emitter can perform until bleaching. Hence, changing measurement conditions and evaluation algorithms can only improve the method up to this physical limit. Current dyes yield enough photons to provide a resolution of up to $\sim 10\,\text{nm}$. Still roughly a factor of ten above molecular resolution.

However, in combination with expansion microscopy molecular resolution seems achievable. In our work "Molecular resolution imaging by post-labelling expansion single-molecule localization microscopy (Ex-SMLM)" [11] we imaged, amongst other structures, microtubule filaments under different expansion protocols and compared the results with theoretical predictions. Performing *d*STORM in combination with ExM protocols is rather difficult. Expanded hydrogels do not work well with the required switching buffer. Ionic interactions between ions of buffer and gel induce water loss and shrinkage of the gel. Therefore, the gel was re-embedded into an uncharged polyacry-

lamide gel. Alexa Fluor 532, losing only $\sim 50\%$ of its pre-gelation intensity was used as a dye for pre-labelling.

We compared post-expansion images with pre-expansion images to determine whether the gel expanded isotropically and to the expected expansion factor. Only minimal distortions were found. The expansion factor was determined to be ~ 3.1x, which is $\sim 20\%$ less than the expected value of 3.9x, predicted by the expansion protocol. We pin this deviation down to the re-embedding process.

TIRF is the commonly used illumination method for *d*STORM, exciting only small volumes of the sample close to the coverslip. This method is, however, not suitable for ExM-*d*STORM. The gel expands into three dimensions, moving structures further away from the coverslip. With the exponential decay of the evanescent wave under increasing distance from the glass, structures are out of the TIRF excitation range. Therefore, epi-illumination was chosen for excitation.

To evaluate the images, we performed theoretical simulations for a microtubule fila-

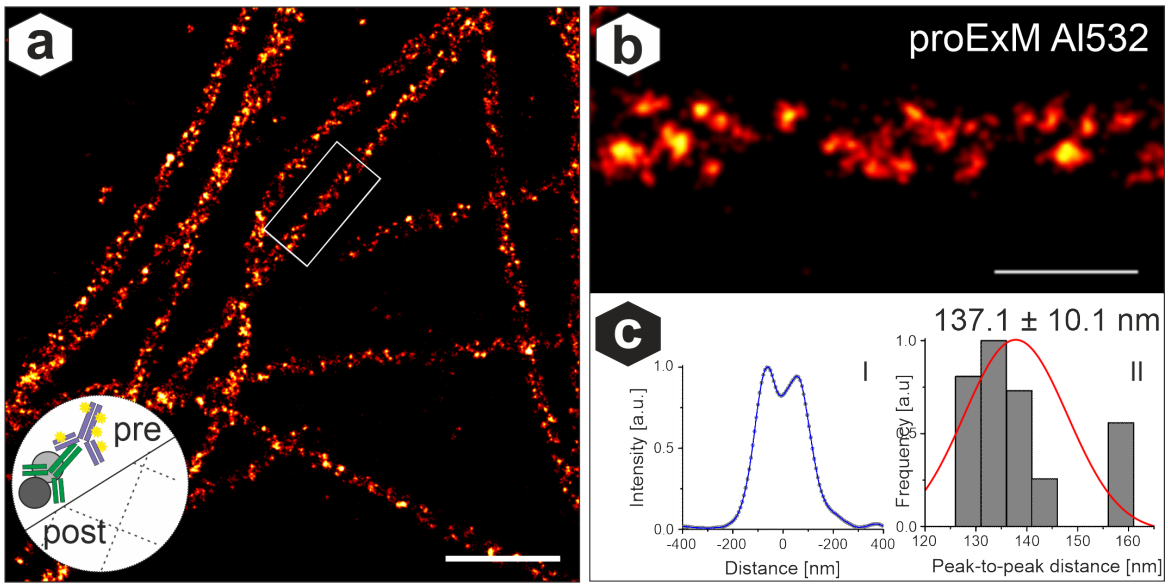

Figure 39: **Pre-labelled microtubules** a) *d*STORM image of microtubules labelled before expansion. b) Zoom in on one filament. c) (I) Profile of the filament in b computed with `LineProfiler`. (II) Histogram of the peak-to-peak distances of the evaluated filaments. Scale bars, 2 µm (a) 0.5 µm (b)

ment intensity distribution under different conditions. The results can be seen in Figure 38.

On pre-labelling, shown in Figure 39, antibodies are digested with other protein structures and linked into the hydrogel. Antibodies and emitters are now disconnected and succumb to the dynamics of the gel. Subsequently expanding the sample increases the distance from the target structure to the antibodies and therefore also the linkage error. The distance from the emitters to the target protein adds up to the length of the primary and secondary antibodies multiplied by the expansion factor.

Overall, the microtubule is expanded to $25 \times 3.3 = 82.5$ nm. The primary antibody

adds 57.75 nm, resulting in an inner radius of 140.25 nm. The fluorescence is also distributed over 57.75 nm, giving an outer radius of 198 nm. Estimating a localisation precision of 10 nm, yields a peak to peak distance of 147.5 nm. We included all profiles in the evaluation that do not contain crossing points or multiple filaments next to each other. With eight filaments from four expanded cells with `LineProfiler` we determined a mean distance of (133.8 ± 13.2) nm (std). Considering the shrinkage due to re-embedding and the varying localisation precision, this value is in good agreement with the predicted model.

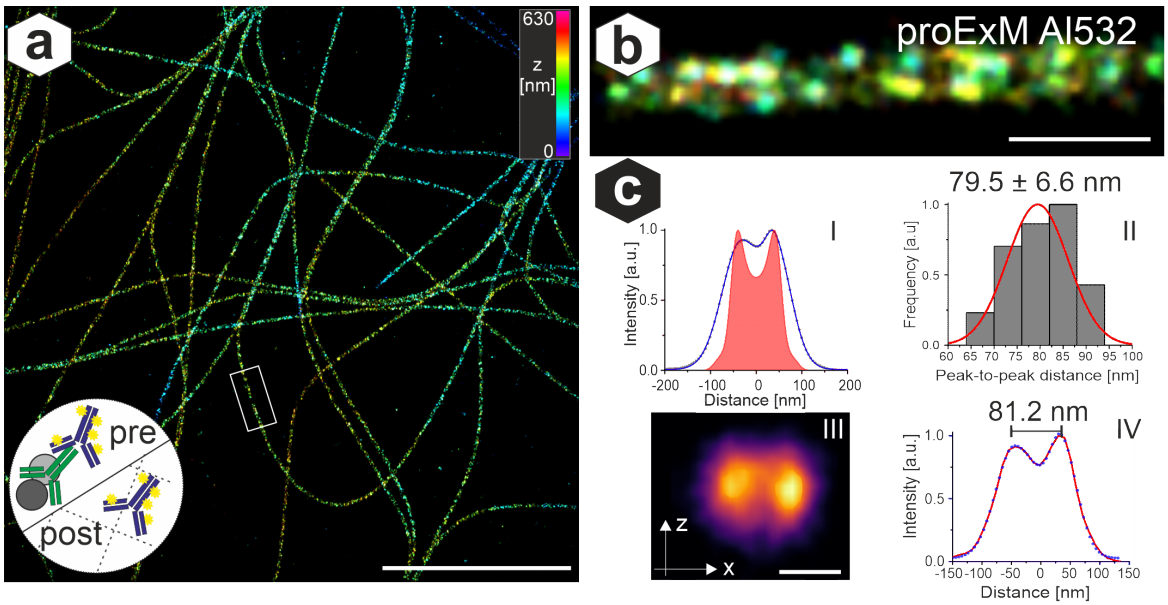

Figure 40: **Post-labelled microtubules 3D** a) *d*STORM image of unexpanded microtubules. b) Zoom in on one filament. c) (I) Overlay of profile in b and theoretical approximation. (II) Histogram of the peak-to-peak distances of the evaluated filaments. (III) X-Z projection showing the hollow structure of the filament. (IV) Peak-to-peak determination. Scale bars, 10 µm (a) 0.5 µm (b) 0.1 µm(c; III)

With post-labelling, shown in Figure 40, the tubulin is labelled after expansion. Therefore, primary and secondary antibodies maintain a size of 8.75 nm, significantly decreasing the linkage error. Inner and outer radius add up to 100 nm and 117 nm respectively, yielding a peak to peak distance of 81.5 nm. With eleven filaments out of two cells, we determined an experimental value of (79.5 ± 6.6) nm (std) for the mean distance. Performing an $x - z$ evaluation along the strands even revealed a profile with a visible hollow structure. This underlines the quality of the measurement. Figure 40 c shows the hollow structure and the referring filament.

We further measured antibody-labelled unexpanded microtubules (cf. Figure 41) to prove the concept of `LineProfiler` on a known dataset. The measurements give a peak to peak distance of (36.2 ± 5.4) nm (std). The value is close to the theoretical prediction of 32 nm.

We further evaluated DNA-labeled microtubules and different ExM protocols. A summary of the resulting peak-to-peak distances and sample sizes is shown in Table 1.

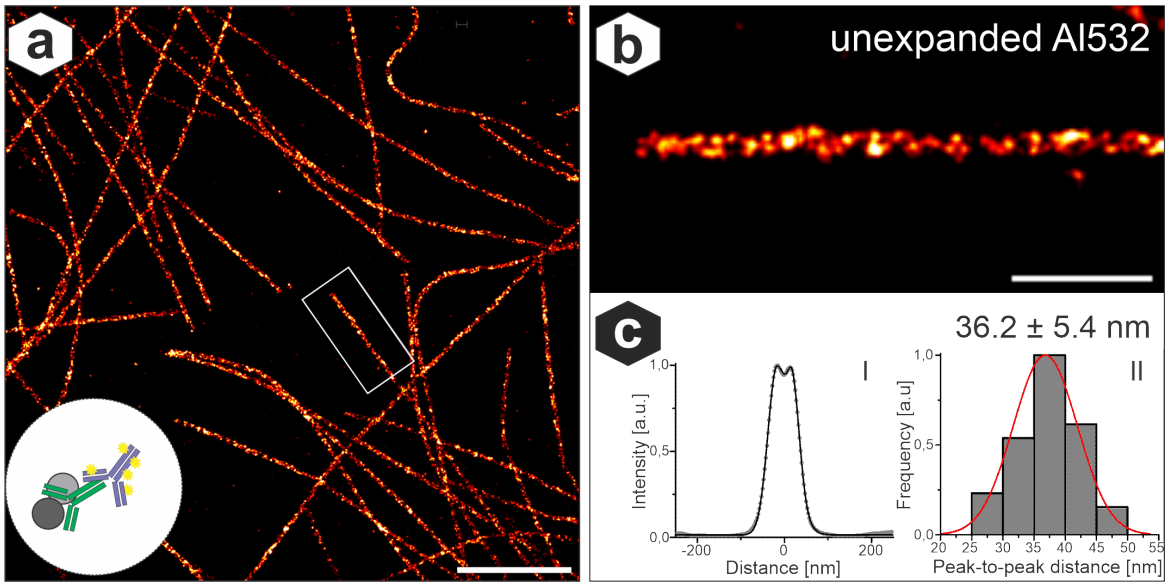

Figure 41: **Unexpanded microtubules** a) *d*STORM image of unexpanded microtubules. b) Zoom in on one filament. c) (I) Profile of the filament in b computed with `LineProfiler`. (II) Histogram of the peak-to-peak distances of the evaluated filaments. Scale bars, 2 µm (a) 0.5 µm (b)

Sample	Peak-to-peak distance [nm]	n filaments|n cells
Unexpaned Antibody	36.2 ± 5.4	35|12
Unexpaned DNA Cy5	43.9 ± 3.7	7|2
Pre-labeled Antibody	133.8 ± 13.2	8|4
ProEXM pre-labeled Antibody	137.1 ± 10.1	9|2
Post-labeled Antibody	79.5 ± 6.6	11|2
Pre-labeled DNA Cy5	201.0 ± 12.9	22|4
Pre-labeled DNA Al532	226.7 ± 15.3	26|2

Table 1: Peak-to-peak distance of microtubules with standard deviation

The evaluation of expanded microtubules showed us, that even with high localisation precision, a molecular resolution is difficult to attain, due to the linkage error. Expansion microscopy can, however, reduce the linkage error if the sample is post-labelled. On labelling with nanobodies and new protocols which can achieve $10-20$x expansion factor, a spatial resolution of ~ 80.0 nm would be sufficient to achieve molecular resolution.

`LineProfiler` was further used in the paper "Tracking down the molecular architecture of the synaptonemal complex by expansion microscopy" [120] to evaluate SC data. The SC was imaged under SIM with 4x expansion. The lateral element SYCP3 and the central elements SYCP1N and SYCE3 were imaged in different colors. SYCP3 describes a helical structure, projected into the lateral plane. As can be seen in Figure 42, the distance between the filaments broadens and shrinks periodically. To find the

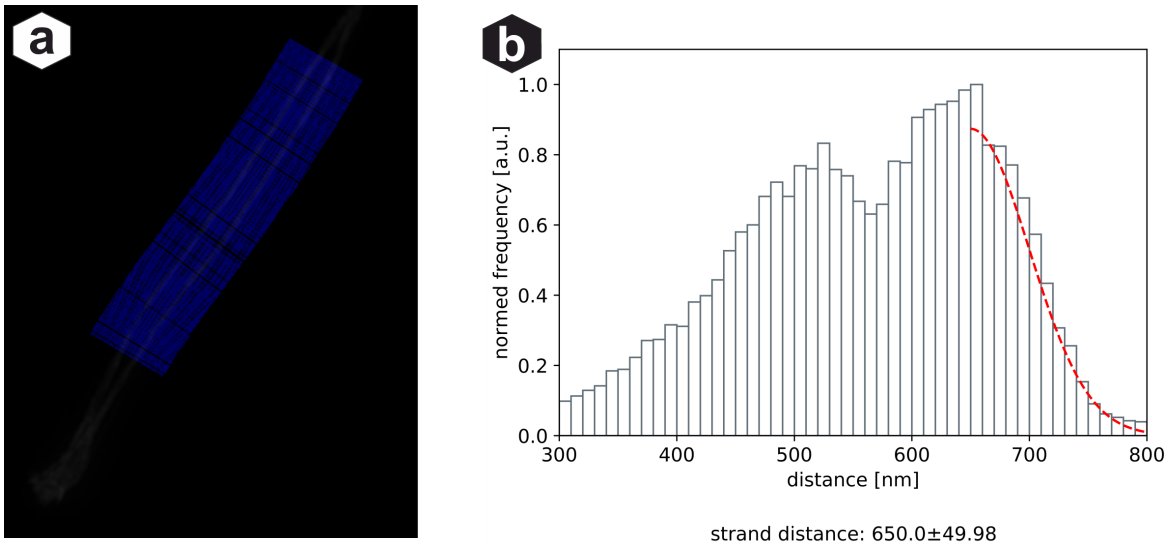

Figure 42: **Synaptonemal Complex evaluation** a) Line profiles (blue) over the helical structure determined over the central element SCYCP1N. b) Distance histogram of the profiles fitted with a half-Gaussian distribution. The fit yields the strand distance and a standard deviation.

distance between the filaments, the point, where the structure is exactly plain, has to be found. In other words, there should not be an unknown z-component, that vanishes due to projection into the plain. This is exactly the case at the point, where the projected filaments are the furthest apart. Collecting a line profile here and evaluating the peak-to-peak distance yields the diameter of the helix.

The centre of the helix, i.e. the starting point of the line profile, can be found with two different methods. If the central element of the helix, SYCP1N, is imaged, the corresponding signal can be straightforwardly used to collect the splines centre and orientation. If no central element is imaged, the centre can be estimated by a skeletonized `floodfill` of the area encapsulated by the projected helix.

Applying these splines to the SYCP3 channel and histogramming the peak-to-peak distances gives values with a Gaussian decay at the upper end. Fitting the upper end with a half-Gaussian function results in an approximation of the maximum distance, i.e. the point where the helical structure is in plain. Conveniently the fit comes with a standard deviation, which can be used as an accuracy quantification for the evaluation.

Evaluating post-labelled SCs resulted in an averaged diameter of $\sim$ 870 nm of the structures. This would be a 4.35 fold expansion in comparison to the previously determined 200 nm by electron microscopy. Evaluating *d*STORM data of SCs resulted in a peak-to-peak distance of $\sim$ 220 nm. We speculate that the additional distance here is introduced by the labelling with antibodies, which bind in undigested SCs mainly to the outside of the structure. Binding on the centre of the SC might be limited due

to cohesin complexes. This marks once again the importance of the linkage error in super-resolved fluorescence microscopy.

4.1.5 Discussion

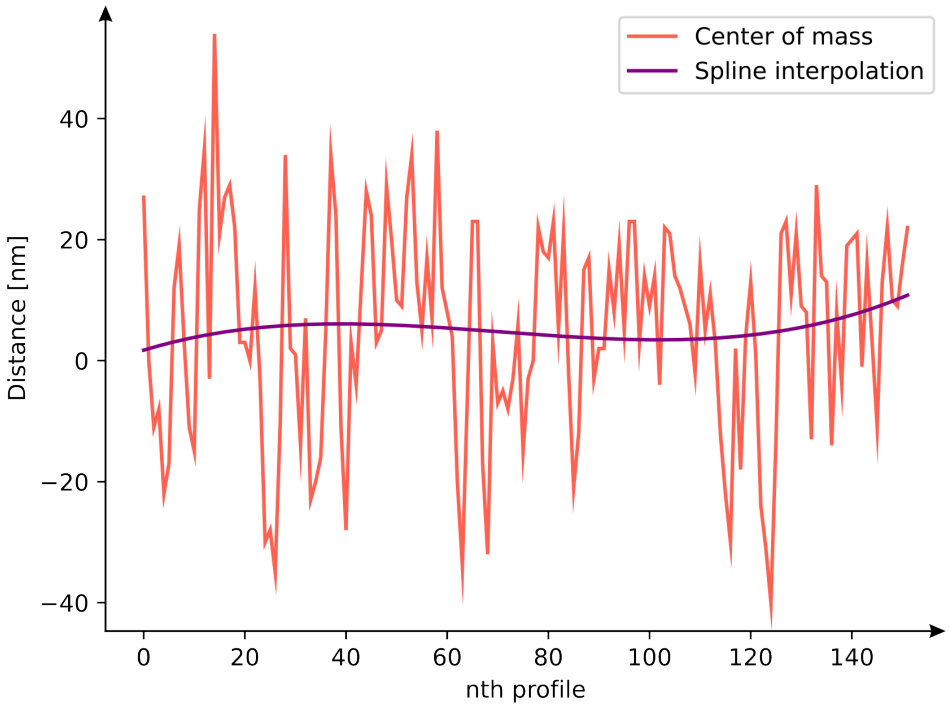

Figure 43: **Shifted line profiles** Difference between the centre of the filament shown in Figure 33 estimated by the spline (zero) and the centre of mass calculated with the individual profiles (red). Spline fitted shift of the profile (magenta).

`LineProfiler` is an objective tool for the evaluation of filamentous structures. The software provides an easy to use interface and allows for high data throughput.
One of the keys for the filament evaluation is to describe the line as precisely as possible. While the skeletonize algorithm already provides a good estimation for the centre of the structure, results could be further improved by weighting the estimation with the intensity values of the image instead of using the binary structure as an approximation. While a direct implementation seems difficult, we used a workaround, collecting profiles with the skeletonize estimation before applying further alignment to the collected profiles. As shown in Figure 43 we computed the centre of mass of the collected data points for each profile. To compensate for incomplete labelling or regions with low intensity, resulting in an abrupt shift of the centre of mass, we used a C-spline to determine a more accurate pathway. Profiles were subsequently aligned according to the C-spline with sub-pixel accuracy. The resulting histograms showed, however, no visible improvements and the applied shifts were generally minor. We, therefore, deduced, that the skeletonized line already reaches the limits of accuracy. Since the secondary alignment also results in performance loss, we did not include it in the final software release.

While colour-coding lines and profiles is a sufficient solution for small images with few filaments, we acknowledge, that recognizing profiles in images with hundreds of filaments is difficult. Cropping filaments out of the image and saving them together with the plot, might be a suitable solution, that is not implemented yet. Currently, the best option for a user is to crop the image by hand to evaluate smaller regions of interest with fewer filaments.
Crossing points of filaments could be computed and excluded by the number of nearest neighbours of each coordinate. Recognizing a count ≥ 2 is straightforward and would probably reduce runtime and increase the accuracy of found crossing points. The feature is, however, difficult to implement since the corresponding algorithm is deeply linked to the used packages.
In the context of SMLM it is beneficial to have algorithms that work on image and on-point cloud data. Future work could include an implementation for localisation files. Here, Region Of Interest (ROI)s can be selected by rendering an image, while the underlying localisations can be directly used to estimate a spline. The resulting profiles could incorporate the intensity and accuracy of the points to improve the performance and accuracy of the algorithm.

4.2 Automated Simple Elastix

One of the major questions of expansion microscopy is if the performed dilation of proteins of interest and emitters is reliable. Ruptured structures are common and the expansion factor measured for the gel can often not be applied to the structure itself. The scientist has to be certain that the expanded structure seen under the microscope maintains the features of its un-expanded counterpart. A common method to check this is to take images before and after expansion. With these images, one can compare the structures and see whether the expansion is isotropic.
To quantify occurring distortions and measure the structural expansion factor, we developed `Automated Simple Elastix`, a script based on the popular image alignment tool `elastix` [121] [122]. The workflow of `Automated Simple Elastix` is shown in Figure 44. The key idea is to adjust the pre-expansion image to the post-expansion image with a transformation that has as many DOFs as an isotropic expansion. An isotropic expansion should only affect the size of the structure, i.e. the scaling s.
However, the position under the microscope also changes, since the sample has to be processed in between the measurements. Though, the transformation has to further compensate for a change in rotation and translation. Therefore, a suitable choice for the alignment is the Similarity Transform.
Optimizing the Similarity Transform for the pre- and post-expansion dataset gives an estimation for the structural expansion factor (by the scale s) if the optimization process converged properly. To quantify the convergence, i.e. the quality of the transform,

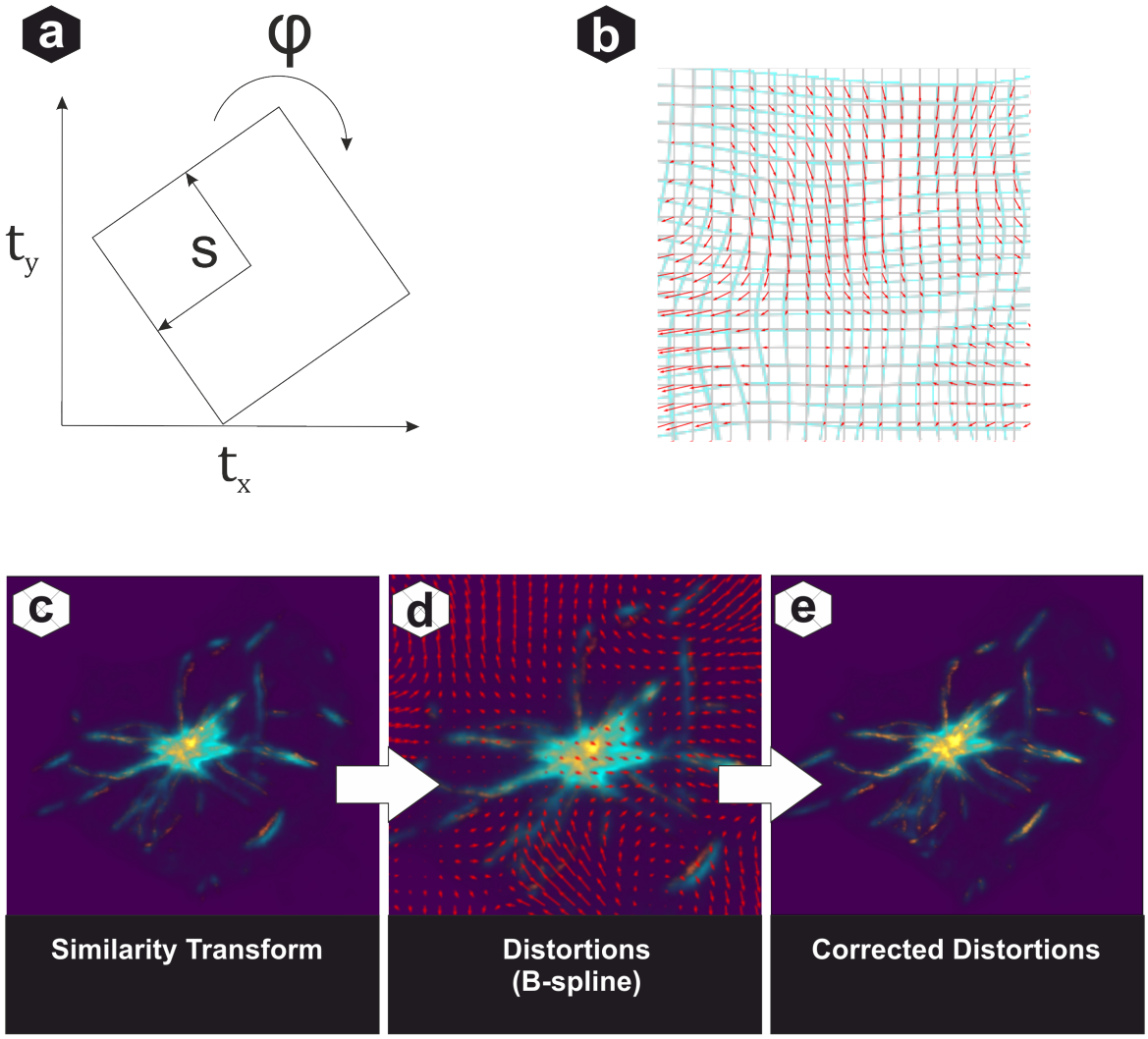

Figure 44: **`Automated simple Elastix`** uses two alignment levels to adjust the pre-expansion to the post-expansion image. The first level, the Similarity Transform (a, c), has four DOF and only compensates for isotropic expansion and position under the microscope. As can be seen in c, the image is still not perfectly aligned. The second level of alignment uses a non-rigid B-spline Transform (b, e). The difference between c and e is classified as distortion and rendered into the Similarity Transformed image as a vector map.

one of the commonly used image correlation indices (cf. Section 3.2.4) can be used. To quantify occurring distortions, further alignment is necessary. Calculating the differences of the Similarity Transform aligned image to a perfectly aligned image reveals the remaining distortions. Its parameters and the applied Similarity Transform are shown in Figure 44 a and d respectively. For the 'perfect' alignment we use a non-rigid B-Spline transformation (cf. Section 3.2.6). This adjusts for non-linearities while keeping the overall structure intact. The transformation is further applied to a meshgrid containing the x and y coordinates of the image (Fig. 44 b). Subtracting the original meshgrid from the shifted one, yields a vector map of the remaining distortions as included in Figure 44 c.

4.2.1 Implementation

The key component of our implementation is `Simple Elastix` [123], a Python binding for `elastix` and `SimpleItk` [124]. Since the only wheels provided are for outdated Python versions, the package has to be compiled from scratch with a C++ compiler like Visual Studio, to work for the desired Python version.
We provided a class structure that is able to work with different input formats. Image data can either be provided as the path to a `tif` file, a `numpy` array or a `SimpleItk` image instance. For the algorithm to run, a source image and a target image are required. The source image is aligned to fit the target image. Providing an initial estimation for the expansion factor improves the chances of a good alignment and the algorithm's speed.
We initialize the Similarity transform with a parameter file, which is stored in the `tmp` folder. It can be freely adjusted to fit different data with different needs. Overlay of source and target image are saved in `jpg` and `svg` format in the data folder. We render the estimated expansion factor consisting of the initial estimate multiplied by the scale factor of the Similarity transform, as well as the Pearson correlation above the image. Subsequently, we perform the B-spline transform. As in the first step the corresponding parameter file is stored in the `tmp` folder. The overlay of the B-spline-aligned images is saved in the data folder. Additionally, we use the computed B-spline transform to compute a distortion map. For this purpose, we initialize two meshgrids

$$X = \begin{pmatrix} 1 & 2 & ... \\ 1 & 2 & ... \\ ... & ... & ... \end{pmatrix} \text{ and } Y = \begin{pmatrix} 1 & 1 & ... \\ 2 & 2 & ... \\ ... & ... & ... \end{pmatrix}, \tag{104}$$

for each coordinate one, and apply the transform. The difference between the transformed and original meshgrid yields a vector map indicating the shifts performed by the B-spline transform. We plot this shift in the Similarity transformed image and save it in the data folder.

4.2.2 Application

In "Subdiffraction-resolution fluorescence imaging of immunological synapse formation between NK cells and A. fumigatus by expansion microscopy" [13] we expanded NK cells and invasive aspergillosis to gain deeper insights into the immune response against fungal infections. Despite being harmless for healthy patients, this type of infection leads to severe diseases for people with compromised immune systems. Common examples are cancer patients, patients with diseases like Influenza or Covid or patients with genetic defects.
We imaged the NK cell microtubules since they play a key role in the directional transport of granules, the substance lysing the invading fungi. Microtubules are rearranged

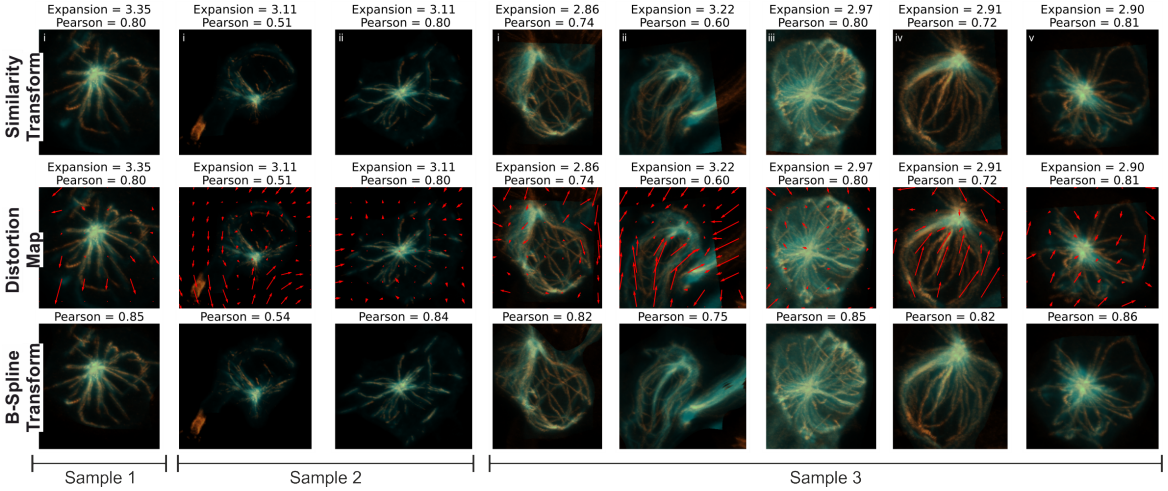

Figure 45: **NK Cell Alpha Tubulin** The three samples originate from three different hydrogels. The expansion factor slightly varies between the samples. The expanded NK cells only show slight distortions in comparison to their unexpanded counterparts. Full alignment with a B-spline transform only yields a small improvement in Pearson correlation, indicating a mainly isotropic expansion. Sample 3 ii shows a high expansion factor compared to other cells of the same sample. We attribute this to the second cell visible in the lower right corner of the image.

during contact with the fungi to actively transport the granule towards the target. This process has not yet been investigated due to the limited resolution of classical fluorescence microscopy.

Taking into account, that the interaction of two different structures of interest is imaged, it is especially important to make sure, that the expansion happens isotropically. Therefore, we used `Automated Simple Elastix` to determine the expansion factor of fungi and NK cells and compared it with the expansion of the gel measured by hand. The analysis process and the results are shown in Figure 45 (NK cells) and Figure 46 (fungi). We determined the expansion factor of the gel to $(3.90 \pm 0.13)\,\mathrm{nm}$ (std). Mapping the structural expansion with `Automated Simple Elastix`, we determined an expansion factor of $(3.71 \pm 0.11)\,\mathrm{nm}$ (std) for $n = 4$ fungi. NK cells expanded in contrast much less with an expansion factor of $(3.05 \pm 0.16)\,\mathrm{nm}$ (std) for $n = 8$ cells. This behaviour was already observed in Büttner et al. [125] for different cellular compartments.

Comparing distortion maps and Pearson correlation indices, it can be seen, that NK cells expanded less but more isotropic. The distortion maps for the mitoRFP structures contain larger errors indicating larger shifts for individual structures. A larger difference between the Pearson correlation before and after the B-spline transform confirms a significant improvement of the structural overlay by nonlinear alignment. Figure 45 sample 3 ii one can see comparably large distortions as well as a larger expansion factor. We attribute this to the second structure visible in the lower right corner. Taking a look at the corresponding distortion map it can be seen, that nonlinear alignment shrinks

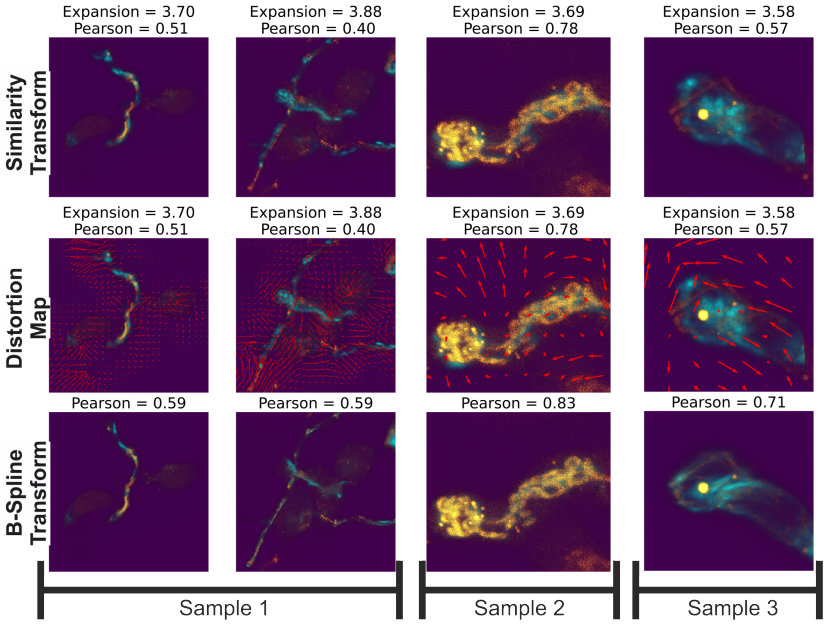

Figure 46: **Fungi mitoRFP** The expansion factor of fungi is generally closer to the one in the gel as in NK cell values vary slightly from measurement to measurement. Comparing the Pearson correlation of similarity and B-spline transform one can see larger improvements. This indicates larger distortions during the expansion process. The overall structure is, however, still intact.

the post-expansion image towards the centre of the structure. Thus, we conclude, that aligning multiple structures in one image introduces a bias towards the expansion factor of the gel, aligning the centre of mass of the underlying structures.
We pin the differences in structural expansion down to shortened digestion time used to preserve the signal of the imaged fungi and propose an improved homogenization protocol to equalize the expansion factors. Nevertheless, we consider the isotropic expansion sufficient to deduce significant results from the expanded data.

4.2.3 Discussion

`Automated Simple Elastix` is a convenient tool to check the isotropic expansion of ExM data. `Elastix` provides a strong state-of-the-art image alignment engine, while the script provides an easily usable interface, automation of the whole process and commonly working parameter files for the transformations. Taking a look at NK cells taken out of the same sample, one can see that the variation of the expansion factor is much smaller than the variation between different samples. Therefore, we estimate, that the error introduced by the protocol is much larger than the error of the algorithm. Therefore, we used the standard deviation as a measurement of uncertainty.

If one wants to quantify the structural expansion factor it is beneficial to crop the structures of interest, since the structural expansion factor can differ from the one of the gel. Comparing Figure 45 with Figure 47 it can be seen, that the alignment of multiple structures yields an expansion factor biased towards the overall expansion factor

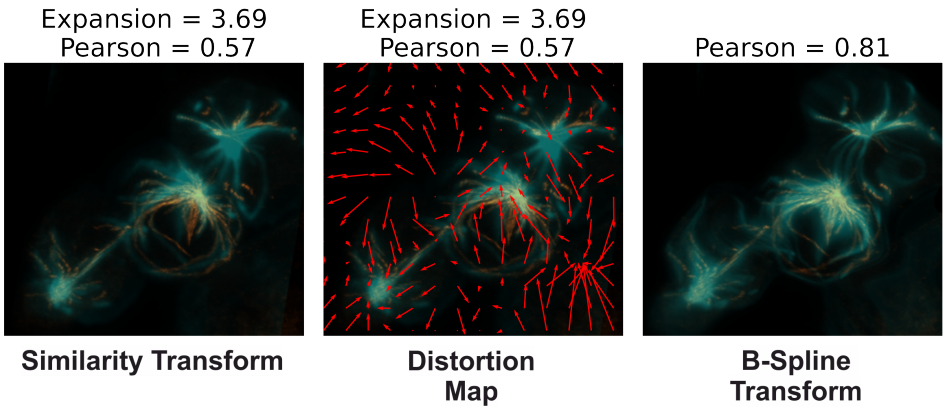

Figure 47: **Alignment of multiple NK cells** The estimated expansion factor is close to the one determined for the gel. The distortion analysis and Pearson correlation indices indicate additional nonlinearities. The cells expand less than the gel resulting in an alignment of the centre of mass.

of the gel.

Further improvements could be made in the context of portability. As previously mentioned `Simple Elastix` has to be compiled from source which is a rather difficult task for a scientist unfamiliar with C++ coding. Compiling wheels for all newer Python versions is a time-consuming task, but would greatly improve the portability of the code.

Making a GUI and building an executable file as in `LineProfiler` would also solve the problem of portability and additionally improve the handling of the software.

4.3 ReCSAI

As mentioned in Section 2.4 SMLM is an important technique to resolve fluorescent emitters with high resolution by determining the centre of the corresponding PSF. While this task can be easily solved for sparse emitters at good Signal to Noise Ratio (SNR), reality is often more difficult. Low SNR as well as overlapping or varying PSFs still pose a challenge. Selecting a suitable method for one's dataset is crucial. Sage et al. [45] give a good review of the performance of state on the art algorithms for different applications.

4.3.1 Challenges of Confocal *d*STORM

Most classical fitters rely on a Gaussian approximation of the PSF. This is a good approach for well-separated emitters in good SNR. As shown in Wolter et al. [8] using a method like the Levenberg-Marquardt algorithm results in a fast and stable fitter. Quasi-second order optimisation yields quadratic convergence for the convex PSF and is, therefore, given a suitable starting position, able to find maxima in a few optimization steps.

Performing a Maximum Likelihood Estimation (MLE), by optimizing the negative log-likelihood achieves slightly better estimations, but requires more optimization steps. The corresponding fitters are, therefore, much slower.
While these algorithms cover the majority of use cases, one of the problems of SMLM remains unsolved: bringing super-resolved dyes into an extended biological context. This is difficult since only a few dyes are suitable for *d*STORM. With an overlapping absorption and emission spectrum, distinguishing the labelled structures is challenging. Combining *d*STORM with other highly resolved imaging modalities that do not depend on photoswitchable dyes like SIM, is an option to get at least a poorly resolved grasp of the targets surroundings [72].
Another promising workaround is to distinguish emitters with similar emission wavelengths by their lifetime. A suitable approach for this is a TCSPC measurement, performed at a FLIM setup. As described in Section 2.2.2, this includes a confocal excitation and detection. The corresponding scan of the sample poses a challenge in combination with the temporal switching emitters of a *d*STORM measurement. Emitters can switch into their non-fluorescent OFF-state during the acquisition of an image. This leads to nonlinear disrupted PSFs. Assuming a radial symmetry as commonly done in classical fitters does not give a precise estimation of the centre of the photon distribution. I.e. applying a Levenberg-Marquardt optimization on a least square loss would shift the estimated centre in the direction of the present data, away from the actual centre. Hence, the irregular chopped PSFs require a large degree of flexibility, which can only be provided by ANNs.
Being essentially high dimensional function approximators, ANNs can work with said irregularities if they are fed with large training datasets. Since the ground truth of acquired images is commonly unknown, the first component required for an ANN fitter is a precise **simulator** that can deliver huge datasets in reasonable time intervals.
The fitting of emitters is a redundant problem. Hence, we decided to crop emitters before fitting by denoising the image and subsequently finding local maxima. For this step, we implemented a **trainable wavelet filter**. With regard to the ANN, we have implemented several modern **architectures** and compared their performances. Additionally, we implemented a **differentiable Compressed Sensing layer** into the network that either works as a prior or full computational compartment, depending on the used architecture. In order to map the output of our layer in coordinates, we had to select suitable **activation functions** for the resulting components. Finally, we defined a **loss function** that trains the network in the best possible way.
The trained network is applied to confocal *d*STORM data and the performance is evaluated with Fourier Ring Correlation as well as with `LineProfiler`.

4.3.2 Simulation

Simulating confocal *d*STORM measurements is challenging. The data-creation has to be performed on a sub-lattice with at least nanometre resolution in order to place PSFs with the precision required for the final algorithm. Exceeding the pixel size of the camera by a factor of ~ 100, while simulating thousands of frames in a parallel manner, can rapidly lead to computational bottlenecks. Additionally, the algorithm has to take the switching of emitters with respect to the scanning time of the detector into account. We will start by explaining the general concept before talking further about the necessary optimizations for a performant tool.
For our images we create an artificial space $\{S\}^{N,N}$, with dimension $N = s_{px}s_{im}$. Here, s_{px} and s_{im} denote the size of a pixel in nanometres and the number of pixels in one frame respectively. The final image will be projected on $\{I\}^{s_{im},s_{im}}$. Emitters are simulated with the following properties:

- Position x: Spatial position in x direction in nanometre
- Position y: Spatial position in y direction in nanometre
- Lifetime τ: Time in the fluorescent ON state until returning into the non-fluorescent OFF state
- Switch on countdown t_{on}: Time offset for an emitter to switch onto the fluorescent ON state.
- Photon count p: Intensity of the emitter in n photons per frame.

A sufficiently large set of such emitters is written into the emitter set L. To emulate the switching behaviour we define a subset of emitters in the fluorescent ON state L_{on}. In each frame emitters are added to L_{on} according to a Poisson distribution:

$$P_\lambda(k) = \frac{\lambda^k}{k!}e^{-\lambda}. \tag{105}$$

For the rendering process, we define a time variable t. t is increasing with each horizontal line by $\Delta t = 6\,\text{ms}$. Column-wise movements of the detector are neglected since they are an order of magnitude faster. The switching of the emitters, therefore, only occurs sparsely.
The detector line covers an area of s_{px} entries of S per line. These entries are rendered simultaneously. If a localisation is added to L_{on}, it is checked whether $t_{on} > 0$. If this is the case, t_{on} is reduced by $\Delta t = 6\,\text{ms}$ and the emitter is not rendered in the current line. If $t_{on} <= 0$ and $\tau > 0$, the part of the emitter that is covered by the current detector line is drawn. Here, we add a clipped version of the PSF to S. Thereby, the centre of the PSF is given by the x and y coordinate of the localisation. p photons

are distributed over the PSF. If the emitter is in an ON state Δt is subtracted from the remaining lifetime τ. If the lifetime τ drops below 0, the emitter is not rendered anymore. So for a time point t, we render all active emitters, where parts of the PSF overlap with the current detector line. Figure 48 shows the principle of this simulation and the generated data.

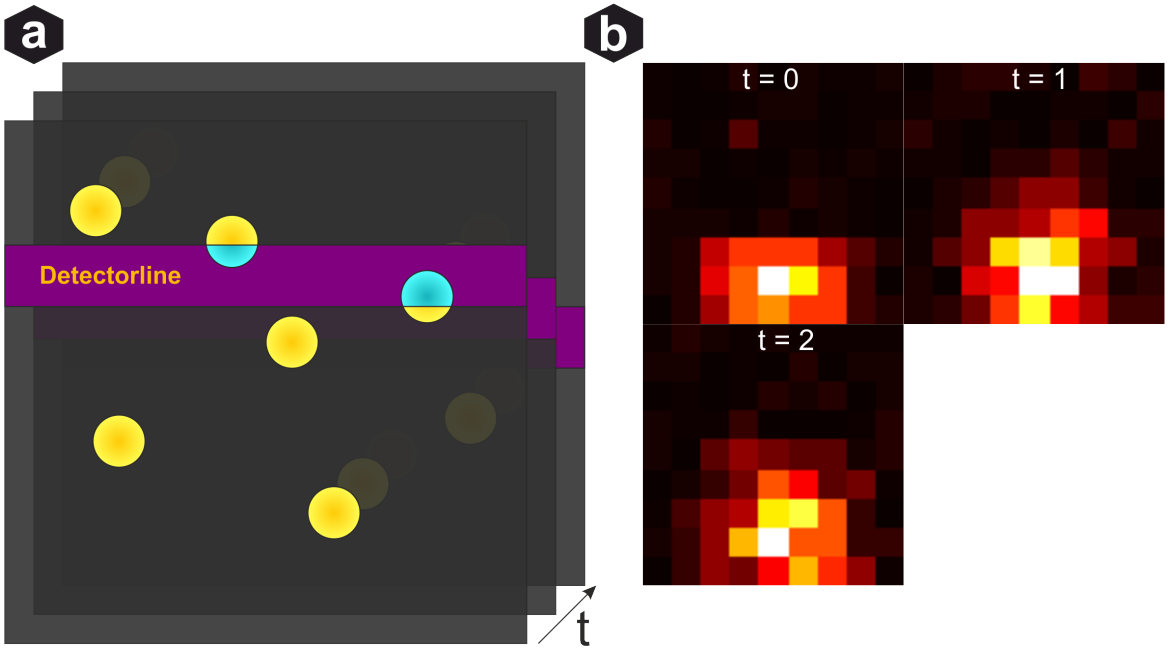

Figure 48: **Simulation** a) The image is scanned line by line by a galvo scanner. While intra-line scanning is relatively fast, switching can occur during the sampling of subsequent lines. We simulate this process, by rendering components of the PSF relative to the detector line. b) Simulated data. PSFs are afflicted by noise and feature the characteristic cut-off in between lines. We simulate crops in three consecutive time-frames to enable our network to take the temporal context into account.

We use a PSF, pre-sampled on a grid with nanometre resolution. The model utilizes an airy disc and is provided by the `astropy` package [126]. Implementing a measured PSF would be straightforward, but the quality of the PSF does not represent a bottleneck in Confocal *d*STORM evaluations yet.
The required radius is drawn from the σ of the current localisations multiplied by three ($r = 3\sigma \in [525, 555]$). Emitters rendered into the image have to be added to a ground truth list. Those, where the lifetime passes before a sufficient amount of their PSF was rendered, have to be excluded from that list, because we do not want to disturb the training process with a ground truth that is not represented in the data. We set the necessary threshold for a localisation to be accepted to 40% of the y-coordinate space covered by the PSF.
Summing everything up: S now contains a set PSFs, incomplete at the top or bottom. This depicts the switching process between fluorescent ON- and non-fluorescent OFF-state during the acquisition with a one-pixel detector. This is typical for confocal *d*STORM measurements. Subsequently, we sum up all photons covered by one pixel,

by using an InterArea interpolation, to yield I. Since localisations are cropped into ROIs, we simulate the data in batches of $[9, 9, 3]$ taking the temporal context of the previous and subsequent frame into account. Each of these batches contains $n \in [0, 10]$ localisations.
The noise in the images was modelled according to [127]. Photons impact the detector with a quantum efficiency E_Q. The measurement triggers the collapse of the wave-function. Thus, the probability of detection at a certain position is given by a Poisson statistic of the expectation value of the PSF at that position $P_{E(x,y)}$ multiplied by the number of photons n_{ph}:

$$e^-_{\text{detector}} = P_{E(x,y)}(n_{ph} E_Q). \tag{106}$$

Subsequently, photons are converted to electrons in the detector. The readout noise of these electrons is Gaussian distributed.

$$e^-_{\text{out}} = \text{sample}(e^{-\frac{1}{\sigma_{\text{dark}}}}) + e^-_{\text{detector}} \tag{107}$$

Dark current only plays a minor role in modern chips with fast acquisition times and was, therefore, neglected. The electrons e^-_{out} are further converted to digital units in the Analog to Digital Converter (ADC) and each pixel of the image $I_{x,y}$ is assigned the corresponding value:

$$I_{x,y} = \begin{cases} e^-_{\text{out}} s & \text{for } e^-_{\text{out}} s < I_{\max} \\ I_{\max} & \text{for } e^-_{\text{out}} s > I_{\max} \end{cases} \tag{108}$$

Here, s is the gain and $I_{\max}$ defines an upper limit for the pixel value defined by the capacity of the chip.

4.3.3 Wavelet filter

Reconstructing images in SMLM is a computationally intensive task. Typical datasets contain thousands of frames with $10^5 - 10^7$ localisations. To reduce the amount of processed data, we crop localisations into ROIs of 9x9 pixels. To identify these ROIs we use a trainable wavelet filter bank. As explained in Section 3.2.5 a wavelet filterbank has these requirements [128]:

- Perfect Reconstruction,
- Orthogonality.

We initialize the filterbank with a Dauchbechies wavelet [129][chapter 6.5]. A common training process to minimize a certain loss would result in variables that do not fulfil the filterbank requirements.
We ensure perfect reconstruction by sharing the weights of the deconstruction filters with the synthesis filters. Keeping low pass Φ and high pass Ψ filters orthogonal is more

difficult and requires the implementation of a constraint. We ensured orthogonality by coupling Φ to the Ψ with a Gram-Schmidt process

$$\Phi_{k+1} = \frac{\Psi_{k+1} \cdot \Phi_{k+1}}{|\Psi_{k+1}|^2} \Psi_{k+1}, \tag{109}$$

while Ψ is freely trained by the network. A bias is then applied to the decomposed image before using a ReLU activation. This combination is used to filter noise frequencies and only sustain frequencies present in the PSF.
Applying the inverse filterbanks, the filtered signal is reconstructed. We perform four wavelet decompositions of the image. The network is trained for 10^4 iterations. The loss is a l_2 loss between the output of the network and a noiseless version of the simulated image. With only a few trainable variables, i.e. the filterbank and the bias, the network is fast and lightweight. The training process is shown in Figure 49.

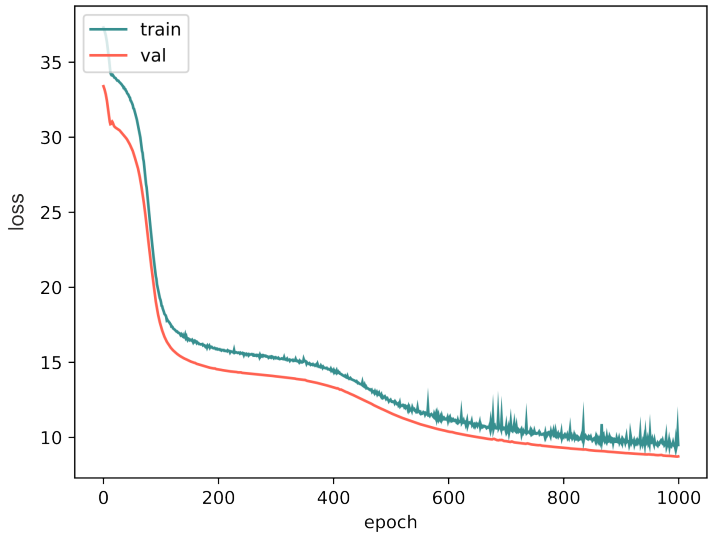

Figure 49: **Wavelet training** Training (cyan) and validation loss (orange) of the Wavelet filterbank network.

To ensure the functionality of the network, we tested for perfect reconstruction of the signal if the bias is set to zero.
Applying a local maximum detection yields the coordinates of the potential localisations. These are cropped to the mentioned $[9, 9, 3]$ array and used for further evaluation. Figure 50 b shows the denoising and thresholding of the network as well as the areas of interest marked by the red rectangles. Figure 50 a shows the corresponding data.

4.3.4 Architectures

To integrate compressed sensing into an ANN, we implemented several state-of-the-art network architectures. The simplest one implements a CNN. As previously described, the input is a crop of the ROI around the PSF. We also use a fixed output format for all networks. This output has the same size as the input dimension to encourage

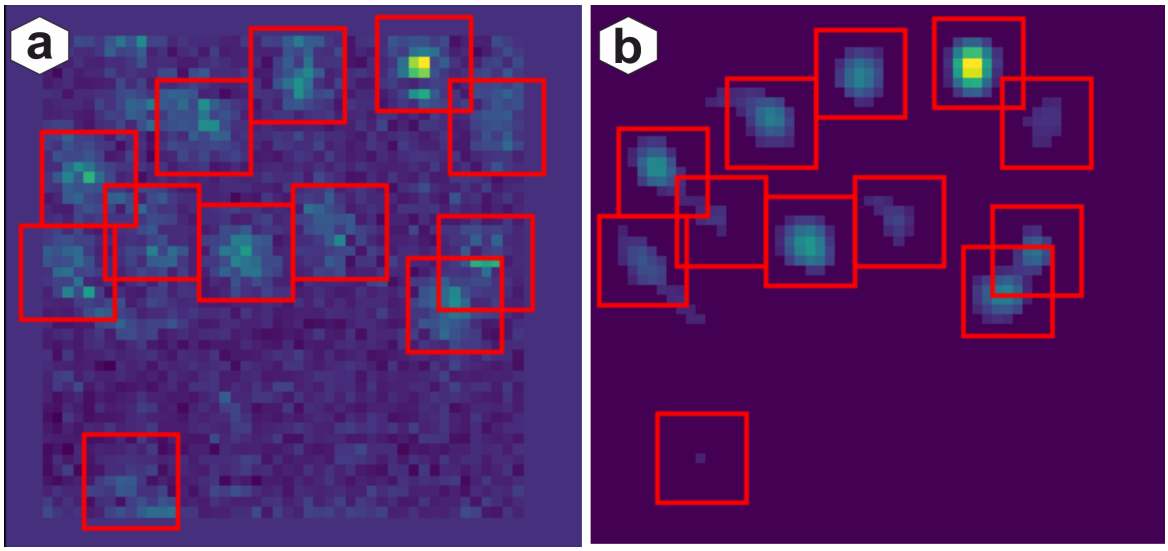

Figure 50: **Wavelet binning** The network learns to extract frequencies that are PSF like. By applying independent thresholds on the Wavelet decompositions noise frequencies are filtered out (a unfiltered, b filterd). With a local maximum detection potential ROIs (red squares) can be identified.

the local processing of information. This means that each pixel of the output's feature map contains information about the corresponding region of the input image. We create eight of these feature maps $\{\boldsymbol{F}_{i,j,k}\}^{9,9,8}$ per crop $\{\boldsymbol{I}_{i,j,k}\}^{9,9,3}$. These feature maps contain all the necessary information about the potential underlying localisations. $\boldsymbol{F}_{k=0}$ indicates a probability of whether a localisation is present in the underlying pixel. $F_{k\in[1,2]}$ denotes the offset from the pixel's centre to the localisation in x and y direction, respectively. Hence, it gives sub-pixel accuracy for the coordinates. $\boldsymbol{F}_{k\in[3,4]}$ computes an uncertainty for the x and y component. $\boldsymbol{F}_{k=5}$ indicates an intensity value proportional to the number of photons in the localisation. $\boldsymbol{F}_{k=6}$ yields the corresponding precision of the intensity. $\boldsymbol{F}_{k=7}$ maps the background.
With the definition of this input and output format, we engineered and compared promising network architectures. We started with a simple CNN and step by step increased the complexity.
Our **CNN** architecture uses the FISTA CS layer as a prior. We chose an 8-fold magnification and $\lambda = 0.03$. The resulting sub-lattice is downsampled by three consecutive convolutional layers with stride 2 and subsequent ReLU activation. The outputs contain 64, 128 and 256 feature maps, respectively, allowing for a smooth broadening of information flow. After the downsampling, the computed feature maps have the same dimension as the input image. Hence, we can add a skip connection here and concatenate the feature maps with the original image. Subsequently, we apply a set of horizontal convolutions, i.e. convolutions that maintain the image size. We apply a 256 (1x1) convolution with LeakyReLU activation, followed by a spatial feature detector composed of a (7x1) and (1x7) convolution. Here, we start to slowly reduce the

dimensionality of the input images to 128 and 64, respectively. We apply a BatchNormalization and two further convolutions with 32 and 8 filters. Both with a kernel size of (3x3). The former is again activated with LeakyRelu, while the latter applies the featuremap-wise convolution described in Section 4.3.6.

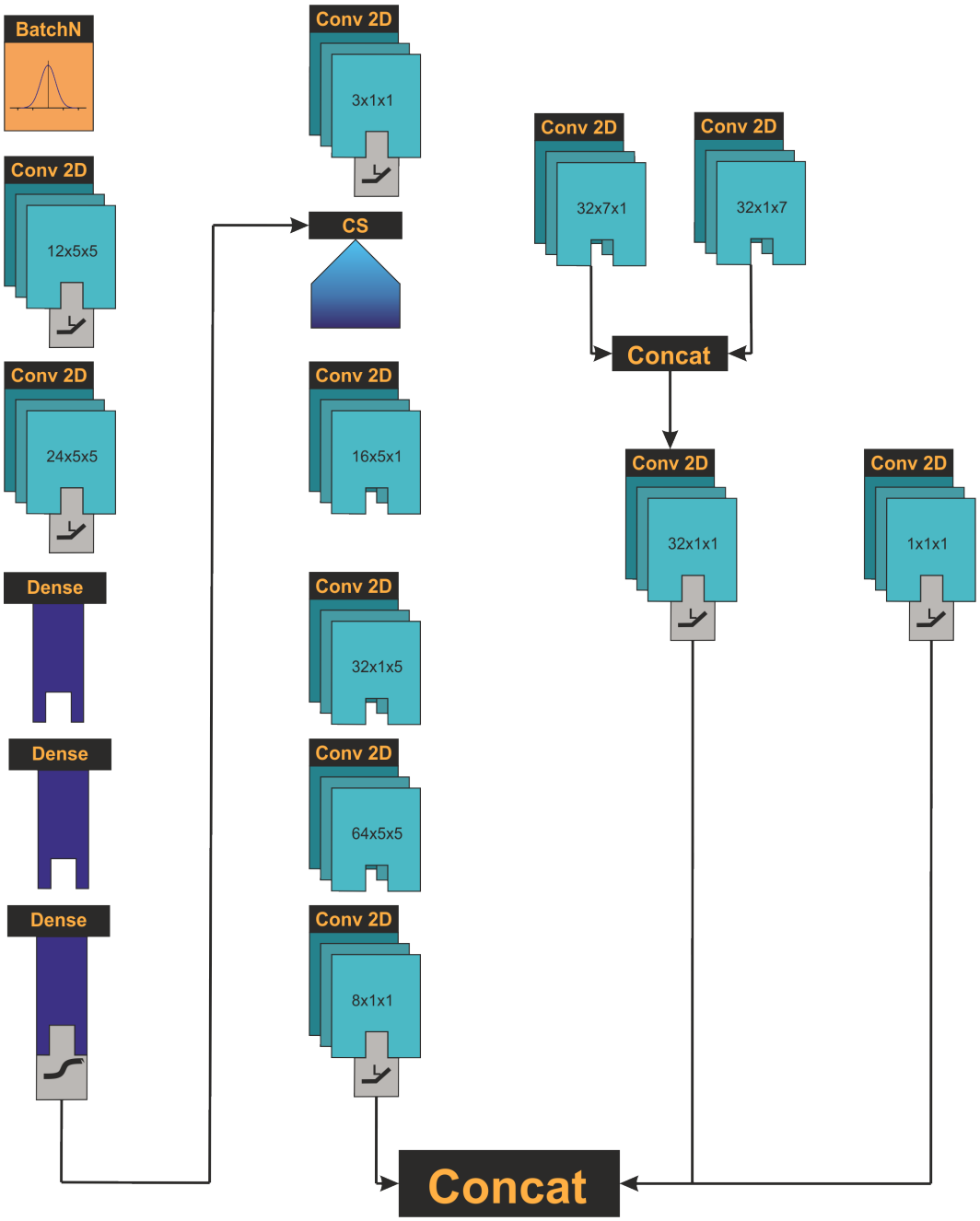

Figure 51: **CS Inception** Modified Inception building block including a trainable CS layer.

CS Inception is a more complex approach based on [105] described in Section 3.4.6. A schematic overview of the layers is given in Figure 51. To include compressed sensing into the inception layers, we slightly varied the architecture. We applied a BatchNormalization on the input before applying two downsampling operations with 12 and 24

feature maps, a kernel size of 5 and a stride of 3. The remaining 24 values are further processed in three hidden layers, with 24, 12 and 1 neuron. The output of this path is used to determine the λ value for the crop. Similar to the CNN approach the sublattice is downsampled with feature detectors in x and y direction and an additional convolution. Parallel we process the input in a feature detector path and a pass-through. Before concatenating the outputs of the different paths we apply a so-called bottleneck layer. A (1x1) convolution that reduces the dimensionality of the input. Here we use 8, 32 and 1 feature maps for the CS path, the feature detector and the pass-through, respectively. The adjusted inception building block (cf. Figure 51) is called two times before applying two bottleneck layers separated by a BatchNormalization and a convolutional layer, reducing the output to the 8 feature maps required as output.

CS U-Net integrates compressed sensing as a prior. As shown in Figure 52 the sublattice is integrated as an elevated starting point for the left side of the U. After downsampling to the original image size the inputs are concatenated. The rest of the network is implemented as described in Section 3.4.7.

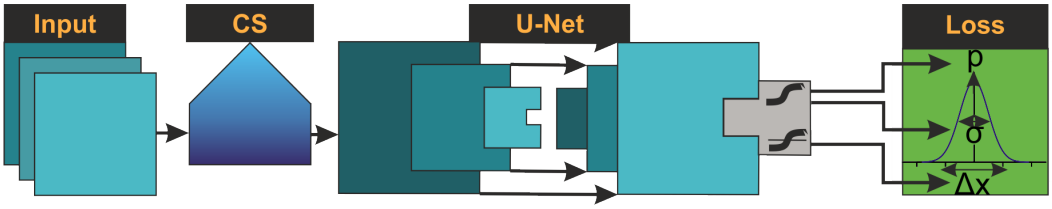

Figure 52: **U-Net** Modified U-Net using CS as a prior. The left side of the U-Net is elevated because CS outputs images larger size than the resulting image.

Even integrated into the Inception architecture, CS still includes an iterative optimization process. Despite our adjustments to make this part fully differentiable, it does not learn efficiently. A possible explanation may be found in the number of iterations. Those can be compared with a very deep neural network without the necessary shortcuts to enable efficient learning (cf. Section 3.4.6). Therefore, we tried to implement the CS part deeper into the network architecture, while reducing the number of required iterations.
Taking a look at the corresponding literature, one can find, that similar concepts have already been successfully implemented. For example, residual U-Nets have been applied in the context of medical imaging. Combining residual learning with two independent U-Nets Le et al. [130] achieved 200-fold acceleration in the reconstruction of simultaneous multi-slice datasets. Mizusawa et al. [131] replaced an iterative reconstruction algorithm for CT images with stacks of independent U-Nets. Achieving similar results under significantly reduced computation time. The most interesting work for our purpose was, however, Lu et al. [132]. Here, the Compressed Sensing (CS) algorithm was

replaced by a residual convolutional network. During the computation, feature space and sub-lattice stay connected. The estimation of iteration is generated by:

$$x_{k+1} = BN(x_k + x_0 + u) \tag{110}$$

where u describes an update computed with a downsampling convolution from the sublattice and BN the BatchNormalization. The concept stays very close to the CS algorithm applying iterative updates computed from the sublattice, while the residual connections enable efficient learning. Therefore, we implemented a concept adapted for our purposes:

Recursive U-Net uses a lightweight U-Net to iteratively switch between feature and image space. All operations of this U-Net use a stride of 3. Feature maps increase from 128 to 256 in the down path and decrease from 256 to the output dimension in the up path. In an initial call, we compute the 8 feature maps required for the output. These are used further used to reconstruct an image representation with the decoder a U-Net as formerly described with output dimension one. We add the estimated background $\boldsymbol{F}_{k=7}$ to the image and compute the difference between the prediction and the original. This difference can be used by the update encoder, another U-Net with output dimension 8, to enhance the initial prediction. It is activated with a tanh function to apply small changes in the range from $\in [-1, 1]$ to the initial prediction. After repeating the iterative encoding and decoding four times, the feature map is activated with the activations described in Section 4.3.6. The workflow of this algorithm is shown in Figure 53.

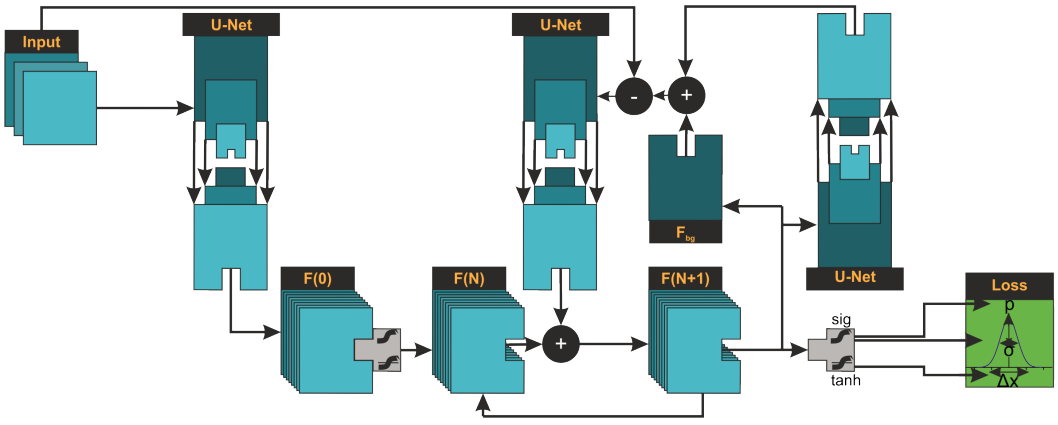

Figure 53: **Recursive U-Net** Algorithm unrolling of Compressed Sensing into the network. The network iteratively switches between feature and image space and receives feedback on the expressiveness of its estimates.

4.3.5 Differentiable Compressed Sensing layer

As explained in Section 3.3 we want to reconstruct the equation $A\boldsymbol{x} = \boldsymbol{b}$. This already implies that the measured image as well as the sub-lattice image are represented as vectors. For example the vector $\{b_k\}^K$ with $K = MN$ for a measured image $\{I_{m,n}\}^{M,N}$ would be constructed as $b_{k=mM+n} = I_{m,n}$. Considering a matrix-vector multiplication

$$b_k = \sum_{l=0}^{L} A_{k,l} x_l, \tag{111}$$

the k-th entry of b is the multiplication of the k-th row of A with the corresponding components of x. This means, that the k-th row of A has to describe the influence each pixel in the sub-lattice has on the first pixel in the measured image.
Bringing this into context with localisation microscopy: A ground truth emitter located at the centre of a pixel has the biggest influence on the pixel final value, while the influence of emitters further away decays. Since this is the case in x and y, the corresponding function, i.e. the PSF, has to be distributed over the row, so that every new line also includes the corresponding line of the PSF. Including a downsampling of $s = (\downarrow 10)$ into the operation adds up to:

$$A_{k,l} = \mathrm{PSF}(\mathrm{pos}_y, \mathrm{pos}_x), \tag{112}$$

with

$$\mathrm{pos}_x = m - v = \mod(k, M) - \mod(l, sM) \tag{113}$$

and

$$\mathrm{pos}_y = n - u = \left\lfloor \frac{k}{M} \right\rfloor - \left\lfloor \frac{l}{sM} \right\rfloor. \tag{114}$$

As one can quickly see, this produces a lot of computational overhead. The computation of one pixel in the measured image includes all pixels of the sub-lattice. Increasing the image size by a factor of two results in a $2^4 = 16$ fold increase in operations, since the image and sub-lattice double their size in x and y direction. However, for standard SMLM measurements including a radius of five pixels in the image, i.e. 50 pixels in sub-lattice space, would be completely sufficient, since the remaining components of the PSF are below the noise level anyway. There are three options to improve the computational speed of the method:
(i) Reduce the size of the input image. Potential ROIs can be pinned down and cropped around a localisation. Subsequently, the algorithm can work with a very small image.
(ii) Use sparse matrices. For large images a majority of the entries of the measure-

ment matrix are close to zero. These values have only a minor impact on the output of the computation and can therefore be neglected. A sparse matrix, e.g

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 3 & 0 & 2 \\ 0 & 5 & 0 \end{pmatrix} \tag{115}$$

can be written in the Compressed Sparse Row format:

$$v = \begin{pmatrix} 1 \\ 3 \\ 2 \\ 5 \end{pmatrix}, \; r = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 2 \end{pmatrix}, \; c = \begin{pmatrix} 2 \\ 0 \\ 2 \\ 1 \end{pmatrix}. \tag{116}$$

Here, v denotes the values of the entries, c the column and r the row. Performing matrix-vector multiplication, one can select all values from one row and use the corresponding row indices to get the vector entries needed for multiplication. The rest can be neglected since one of the two product components, the matrix entry is zero. Considering the vector $\boldsymbol{t}^N$ with $N = 3$, the product for row 1 can be computed as:

$$d_1 = \sum_{i \in r_i = 1} v_i t_{c_i}. \tag{117}$$

If most entries of A are zero, this saves a lot of operations.
Further performance can be gained by implementing the algorithm on a GPU. In comparison to the CPU the GPU can run more tasks in parallel with the drawback of reduced clock frequency per processing unit. The GPU is, therefore, able to outperform the CPU if the computation can be broken down into several independent tasks, as it is the case for matrix-vector multiplication. Moreover, the special architecture of the GPU can be used to further accelerate the process. GPU threads are clustered in so-called blocks. Threads within a block can communicate with each other and share an additional memory, the shared memory. The shared memory is a lower cache level and is much faster and smaller than the global memory, to which all threads have access. Using this structure, we perform the computation of one matrix row in one block. The corresponding row of A is transferred into shared memory, while the vector is read from global memory. A vector u for the result, also allocated in shared memory, is computed by the multiplication of the components. The vector components are subsequently summed up by parallel reduction:

$$u_i = u_i + u_{2i} \tag{118}$$

with $i \in I/2$ repeated $\log_2(I)$ times.
Summing the entries of the second half of the vector components on the first half of the vector components reduces the number of subsequent operations needed from n to $\log_2(n)$.

(iii) Replace the measurement matrix with a strided convolution Using the radial symmetries of the PSF the matrix-vector multiplication can be further accelerated. As previously explained the measurement matrix A can be broken down into a convolution. This two-dimensional convolution can be accelerated by separating the PSF into two one-dimensional components for whom holds:

$$\mathrm{PSF} = \mathrm{PSF}_x \otimes \mathrm{PSF}_y. \tag{119}$$

This reduces the number of required operations from $M \cdot N \cdot n^2$ to $M \cdot N \cdot 2n$. Adding all together, the multiplication with the matrix is replaced with two subsequent convolutions in x and y direction followed by an average pooling with stride ($\downarrow 10$) and kernel size (10,10). The multiplication of the transposed matrix is replaced by two subsequent upward convolutions with stride ($\uparrow 10$).

The choice of suitable hyperparameters for CS algorithms is challenging. It is a balance between precise reconstruction and computation speed. The outputs for different hyperparameters are shown in Figure 54. The reconstruction of a whole film is shown in Figure 55 .Considering for example FISTA, time to convergence and result highly depends on λ. λ implies the error rate within which the original image should be reconstructed. Therefore, smaller λ implies less background and leads to slower convergence, while high λ implies more background and a higher convergence rate. For standard implementations, λ is set for the whole image, either by estimating noise or by subjective user input. By evaluating ROIs our approach is able to choose a suitable λ in dependence on the local noise.
Implementing CS into a neural network is for most parts straight forward. Matrix-vector multiplications and convolutions are differentiable operations and most common libraries offer optimized functions. The challenging part is the non-differentiable soft thresholding. This operation has to be approximated by differentiable functions to allow a backpropagation through the CS component of the network. We implemented a combination of two ReLU activations to build a function with comparable output:

$$o = \mathrm{RELU}(i - \lambda) - \mathrm{RELU}(-i + \lambda). \tag{120}$$

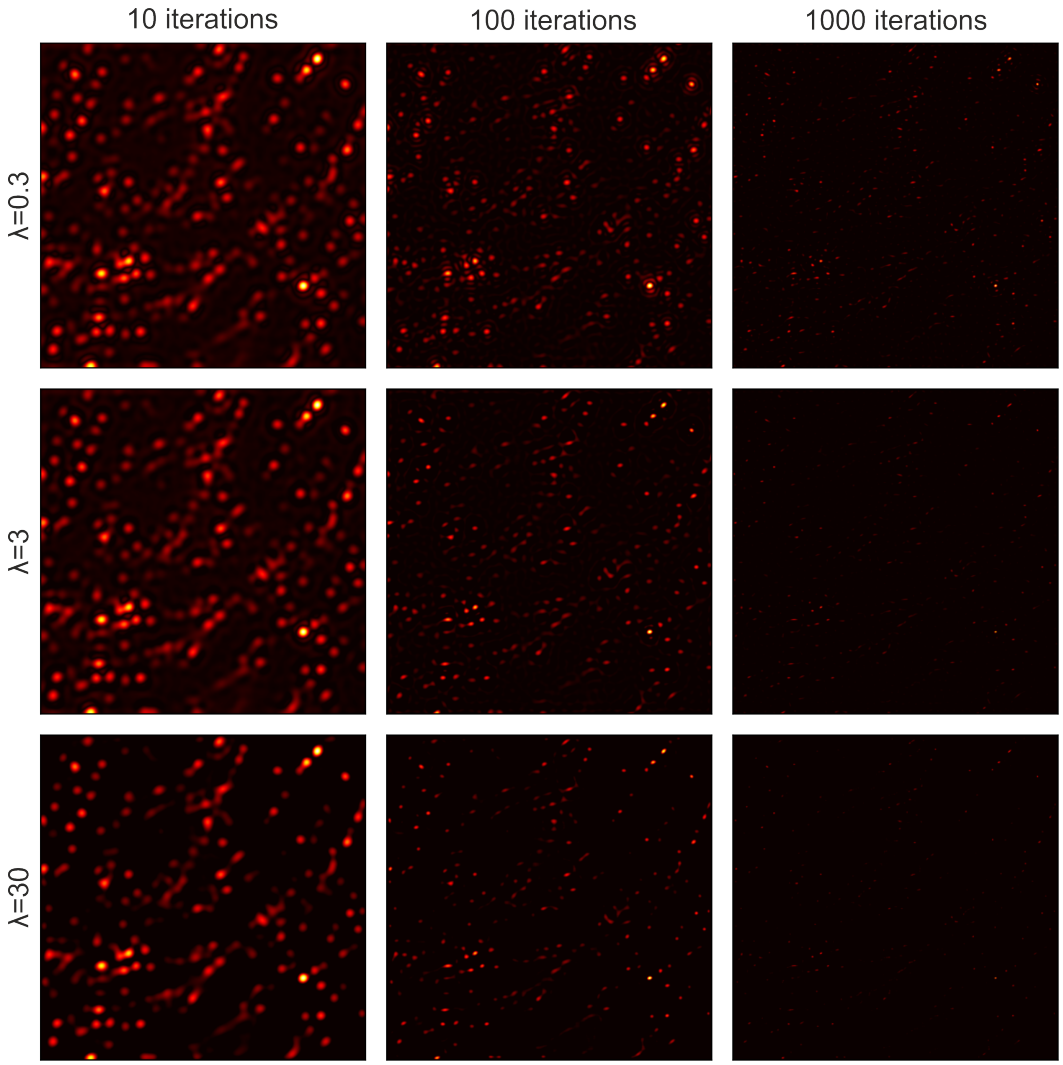

Figure 54: **Compressed Sensing** Output for different hyper-parameters. Lower λ implies less noise in the image and results in a more precise reconstruction. On the downside, this slows down the convergence process. A high λ leads to a loss of emitters since these are assumed as background.

4.3.6 Activations

Since the final output of the network should contain coordinates, we constrained the feature maps to a certain range, using activation functions. The probability for a pixel to contain a localisation, i.e. the classifier feature map, should map the output in a range of $p \in [0, 1]$. This was implemented with sigmoid activation. Subpixel coordinates should cover a range of $[-1, 1]$ to allow some overlap between adjacent pixels. However, pixels further away should not play a role in the reconstruction to maintain the advantages of locality. Hence, a suitable activation function is tanh. The errors of the predicted variables σ_x, σ_y and σ_N were constrained to the interval $[0, 3]$ by a sigmoid activation with the result multiplied by three.

Softmax was also considered as an activation function for the classifier feature map. Despite a higher learning rate, this approach has major drawbacks. Normalizing over

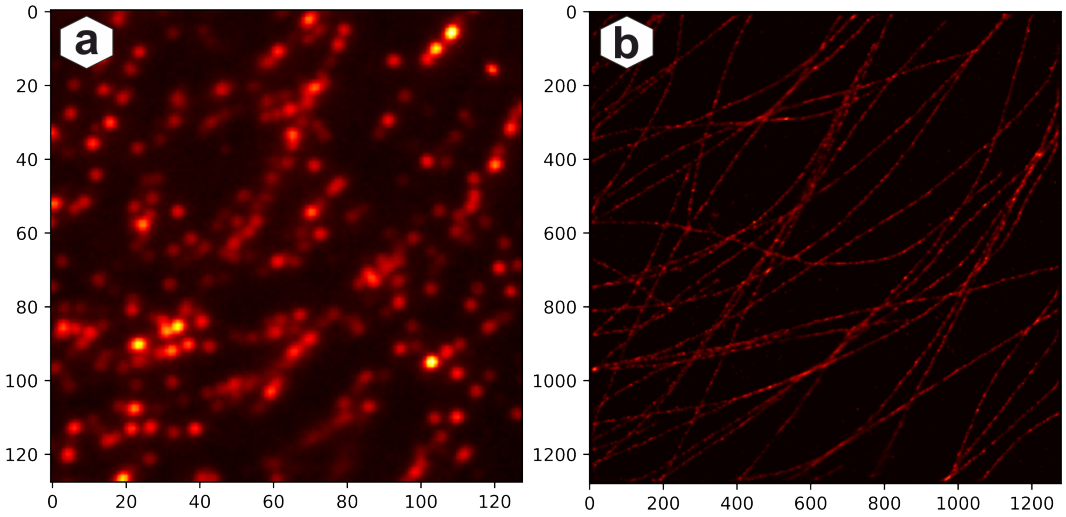

Figure 55: **CS output** a) High density SMLM raw data frame. b) Reconstruction with FISTA $\lambda = 30$; 1000 iterations per frame; reconstruction time $\sim 3600\,\mathrm{s}$

the feature map the outputs always cover the range $p \in [0, 1]$. Giving rise to false positives if there are no localisations present in the observed region, or false negatives if there are several.

A schematic display of the used feature maps, activation functions and loss is shown in Figure 56.

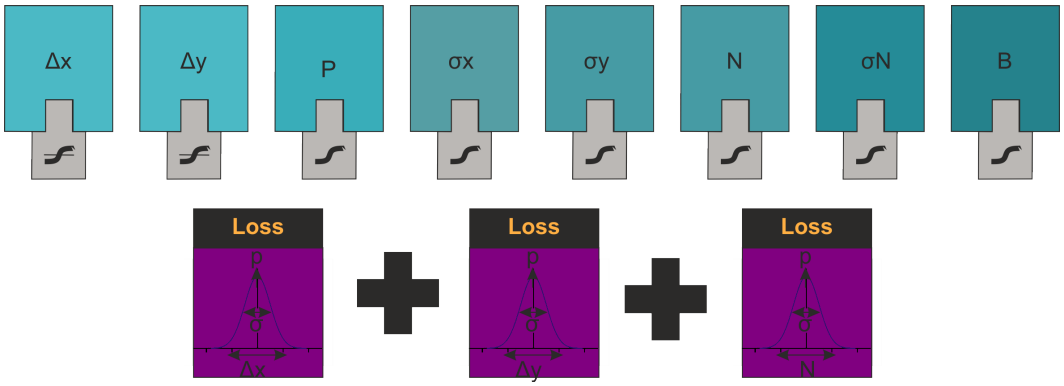

Figure 56: **Activation** functions used for the eight constructed feature maps. The coordinate offsets Δx and Δy are mapped with a tanh function. The remaining feature maps, including the probability P, the positional uncertainties σ_x and σ_y, the intensity N, the intensity uncertainty σN and the background estimation B, are activated with a sigmoid function.

4.3.7 Loss function

As previously described, a suitable loss function is crucial for a performant neural network. Defining a loss function to fit localisations is, however, rather difficult. Localisations are absolute entities, i.e. they are either present or not. Their distribution is also arbitrary. There are areas with high and areas with low density, not to mention that the amount of localisations varies from frame to frame. Hence, using a fixed-sized

result tensor is challenging, since outputs might be over or underrepresented in the output. Consider a feature space of ten tensors with a probability for a localisation, as well as an x and y value. While the first output would nearly always predict a localisation, the 10th output would close to never predict a value. The result is a bias of the network towards these values.

Therefore, we coupled the prediction to the local context of the underlying localisation. As described in Section 4.3.6 each pixel of the output feature space, which equals the input image in x and y size, is able to predict one localisation. Since the position of these localisations can not be determined with arbitrary accuracy, the probability for a feature space pixel to contain an emitter is distributed over adjacent pixels.

This loss was already successfully applied to reconstruct SMLM images with artificial intelligence. The corresponding software `DECODE` [52] sets new benchmarks in all categories of the super-resolution fight club [45]. Therefore, we decided to adapt it. Carrasco et al. [133] and Murphy et al. [134] give good introductions to grasp the concept of probabilistic learning with multivariate Gaussian mixture models.

We use the feature spaces provided by our network to construct a multivariate Gaussian density function for every pixel located at x, y of the input image with dimension N:

$$\mathcal{N}(\boldsymbol{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right). \tag{121}$$

Σ denotes the covariance matrix

$$\Sigma = \begin{pmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_N^2 \end{pmatrix}, \tag{122}$$

which is in our case a diagonal matrix, since x, y and N are independent variables. The vector $\boldsymbol{x}$ is given by the data points of the ground truth. The vector $\boldsymbol{\mu} \in \mathbb{R}^D$ is composed of the predicted variables of the network $\boldsymbol{\mu} = (x, y, N)^T$ and has the dimension $D = 3$. Given M distributions, we need the probability of a data point x_i to originate in the distribution m:

$$p(z_{i,m} = 1|x_i). \tag{123}$$

Here z describes a latent variable, that we will never actually see. It has a value of one if the datapoint x_i originates the distribution m. Concluding this yields the mixing coefficient of distribution m:

$$p_m = p(z_m = 1). \tag{124}$$

This means that the relative intensity of the Gaussian distribution is proportional to the number of points within that distribution. For our case, with only up to one data

point per distribution $p_m \in [0, 1/I]$. The probability of observing all the latent variables $p(\{z_m\}^M)$ can be derived as the product of all $p(z_m = 1)^{z_m}$. Using Equation 124 yields

$$p(\boldsymbol{z}) = \prod_{m=1}^{M} p_m^{z_m}. \tag{125}$$

With the same logic, we can derive the probability for a datapoint given $\boldsymbol{z}$:

$$p(x_i|\boldsymbol{z}) = \prod_{m=1}^{M} \mathcal{N}(x_i|\mu, \Sigma)_m^{z_m}. \tag{126}$$

Using marginalisation over $\boldsymbol{z}$ leads to

$$p(x_i) = \sum_{m=1}^{M} p(x_i|\boldsymbol{z})p(\boldsymbol{z}) = \sum_{m=1}^{M} p_m \mathcal{N}(x_i|\mu, \Sigma)_m. \tag{127}$$

The joint probability is given by the product of all data points in the ground truth:

$$p(\boldsymbol{x}) = \prod_{i}^{I} \sum_{m}^{M} p_m \mathcal{N}(x_i|\mu, \Sigma)_m. \tag{128}$$

With the full model for all data points we can optimize the network by minimizing negative log-likelihood:

$$L_{\text{loc}} = -\frac{1}{I} \sum_{i}^{I} \ln(\sum_{m}^{M} p_n \mathcal{N}(x_i|\mu, \Sigma)_m), \tag{129}$$

which we define as localisation loss. We add the normalization term $\frac{1}{I}$ to keep p_m in the range of $p_m \in [0, 1]$. Taking a look at Figure 57, L_{loc} minimizes σ of the multivariate normal distributions as well as the position Δx, Δy. The mixture coefficient is, however, unconstrained. Hence, we use an additional loss term to keep p_m close to either zero or one. Further, we want to reinforce $\sum_m^M p_m = I$. Here, we use again a Gaussian distribution optimized on the true count I:

$$\mathcal{N}(I|p_m, \sigma_c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp\left(-\frac{(I - \sum_m p_m)^2}{2\sigma_c^2}\right), \tag{130}$$

where σ_c is defined as the variance of a Bernoulli random variable $\sigma_c^2 = \sum_m p_m(1 - p_m)$. We optimize again the negative log-likelihood

$$L_{\text{count}} = \frac{(I - \sum_m p_m)^2}{2\sigma_c^2} + \ln(\sqrt{2\pi}\sigma_c). \tag{131}$$

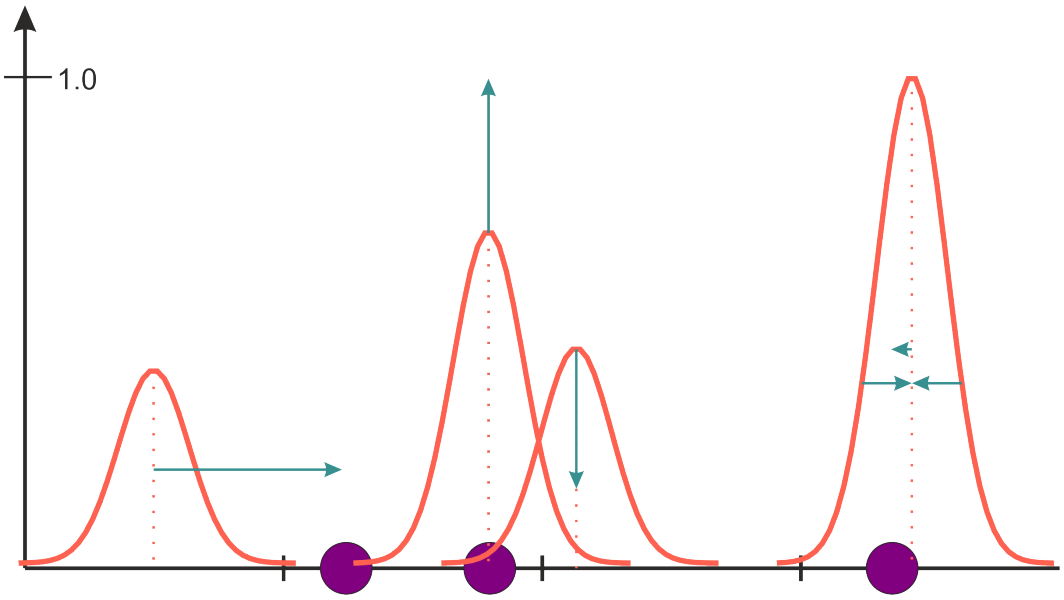

Figure 57: **ReCSAI Loss** Desired characteristics of the ReCSAI loss. Each pixel yields a multivariate normal distribution (here projected in 1D). The cyan arrows indicate how the network should optimize the distributions. The mean value should describe the position of the ground truth (magenta) as accurate as possible (small σ). One emitter should be covered by one distribution. The probability of a distribution should be either one for an underlying emitter or zero for no emitter. If two emitters are located within one pixel the distributions of adjacent pixels should be able to cover it.

The background is included in our model by taking the l_2 loss of the real and the predicted background:

$$L_{\text{bg}} = \sqrt{(P_{\text{bg}} - T_{\text{bg}})^2}, \tag{132}$$

where P_{bg} denotes the predicted map and T_{bg} the ground truth. The full loss is composed as

$$L = L_{\text{loc}} + L_{\text{count}} + L_{\text{bg}}. \tag{133}$$

While this loss works in practice, it tries to optimize a distribution on a discrete data point. However, from a statistical point of view, the ground truth is not a data point, but a distribution as well. As mentioned in Section 2.4 the maximum precision with which an emitter can be reconstructed is limited by the CRLB. Hence, it should not be possible for the network to predict precisions below the CRLB. However, this is a common occurrence during the training process. We speculate that this is either a result of overfitting, or arises due to the fact, that random hits with very high precision yield high negative values in the loss function and, thus, outperform the misses. The prefactor of the loss function would get

$$\lim_{\Sigma \to 0} -\ln\left(\frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\right) = -\infty \tag{134}$$

and the exponent

$$\lim_{\Sigma \to 0} \lim_{\boldsymbol{x} \to \boldsymbol{\mu}} (\boldsymbol{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) = 0. \tag{135}$$

This results in large negative loss values. We propose to include the CRLB or an estimation of it into the ground truth data and use it to describe the ground truth with a distribution. The loss can then be computed using the KL divergence of the predicted and true distribution.

4.3.8 Regularization

Considering the sparse nature of the CS component on the network, it seems intuitive that the implementation of a constraint could enhance the network's performance. An additional l_1-loss after normalization could for example reinforce sparsity and a λ as large as possible. The l_1 loss minimizes if all entries of the sub-lattice are zero. Hence, an additional term that enforces the integrity of the data is necessary. For this, we apply the measurement matrix A to the sub-lattice to create a denoised version of the measured image $s = Ab$. We compare this image to the simulated noiseless ground truth n with an l_2 loss. Overall the loss formulates

$$L_{\text{cs}} = \sum_i |b_i| + \sum_j (s_j - n_j)^2. \tag{136}$$

The additional loss term did, however, not improve the training results. Thus, it was discarded in the final version.

4.3.9 Reconstruction from feature space

As already explained in the work of Speiser et al. [52], a probability threshold has to be set for a feature space pixel to be accepted as localisation. The choice of a suitable threshold is, however, rather difficult. Emitters close to the edge of a pixel, i.e. taking Δx or Δy close to 0.5, result in a probability distribution over the result and the adjacent pixel. Taking fixed values can lead to grid-like artefacts in the reconstruction since the mentioned values close to a border have a reduced probability to be accepted as an emitter. The problem can be solved by preprocessing the feature space to take adjacent values into account.
We tried to apply different filters and found, that a cross-shaped filter delivers optimal performance. The filter kernel F formulates as

$$F = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{137}$$

Following the application of this filter, we apply a threshold of $t_{re} = 0.7$ to accept a local maximum in feature space as an emitter. We only accept the local maximum to prevent double localisation if the value is mainly distributed over two pixels. However, if the probability value exceeds $t_{re} = 1.4$, we assume two underlying emitters and accept the two largest values of the local formation as emitters. This threshold value was determined by computing the JI of the reconstructed validation data in dependency of t_{re}. The resulting metrics are shown in Figure 58. The curve yields a maximum at $t_{re} = 0.7$. Hence, we determined this value as optimal.

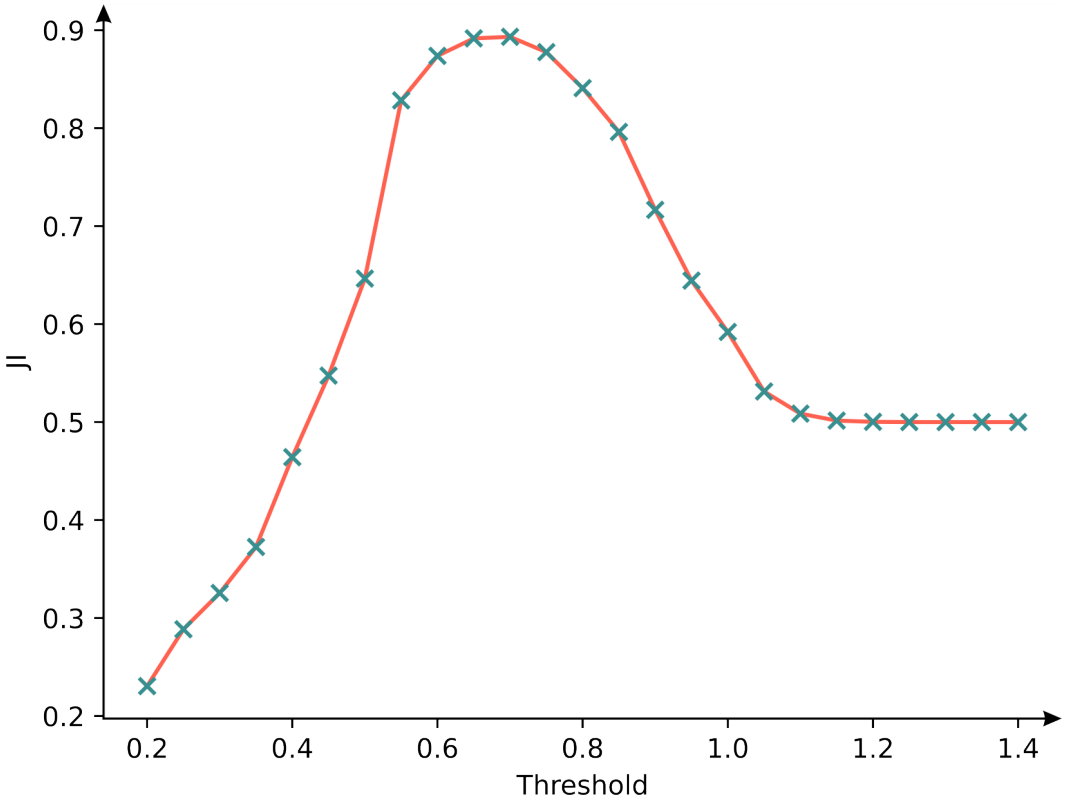

Figure 58: **Parameter determination** The calculated Jaccard Index (JI) for different threshold values is used to determine the optimal threshold for the acceptance of a local maximum as an emitter.

4.3.10 Training

Using the simulations described in Section 48, we created a dataset of 40 batches, each with four sub-batches of 1000 crops. Three of those sub-batches are used for training. The remaining one is used for evaluation. Noise simulations are completely independent and random for each crop. The sigma of the used PSF changes in each batch, taking a random value in the range of $\sigma \in [175, 185]$. This range corresponds to the estimated values that occurred during data acquisition with the used microscope.

We perform 150 training iterations before evaluating the current performance with the evaluation data. Here, we implemented the JI, Root-Mean-Square Error (RMSE) and validation loss as control metrics. Our choice of a suitable optimizer fell on Adam. Our

networks were implemented in `Tensorflow 2` and trained on a Nvidia GTX 1080 TI GPU. The training time of the different architectures is shown in Table 2. The control metrics are plotted in Figure 59.

Network architecture	Training time [s]	Inference time [s]
CS-CNN	1102.0	440.8
CS-Inception	2091.3	502.5
CS-U	596.4	72.5
CS-Res-U	64.1	88.8

Table 2: Training and inference time of different network architectures on an Nvidia GTX 1080 TI. Training times are measured per epoch. For the evaluation of inference time, a FLIMbee dataset with 4500 frames of 45x45 px is used.

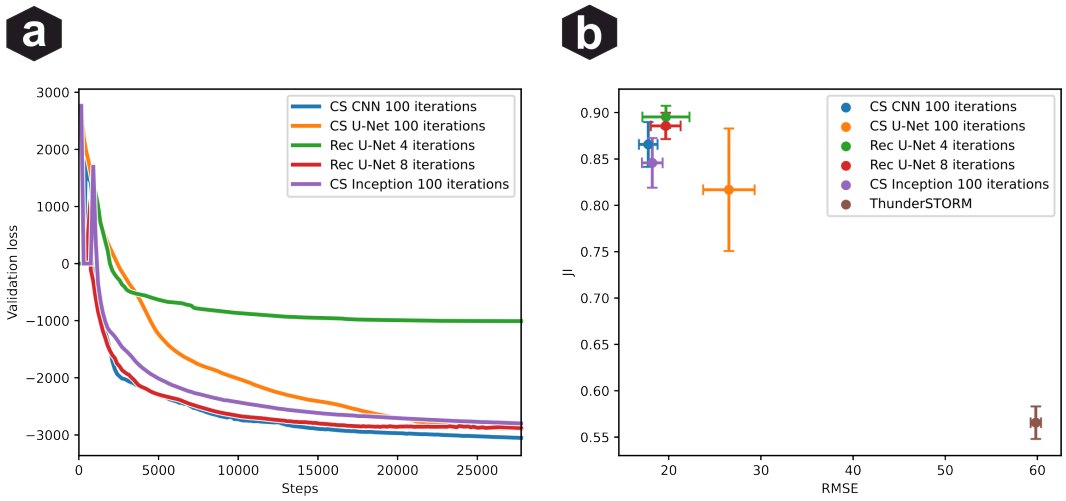

Figure 59: **Network comparison** a) Validation loss in dependency of the training steps. b) Jaccard Index (JI) and RMSE of the tested models. A fit of the same data with ThunderSTORM is shown as a baseline. Error bars denote the standard deviation of the mean for $N = 25$ different validation datasets.

4.3.11 Results

Next to artificial data we also applied our network to real data. Therefore, we performed *d*STORM measurements on a FLIMbee setup with a galvanometric scanner. The data shows interrupted PSFs and intensity variations in between the scanning lines (see Figure 60 a). Our network is able to compensate these non-linearities and precisely determine the position of the emitters. We used the Fourier Ring Correlation (FRC) and `LineProfiler` to asses the prediction quality of the measured microtubules. The resulting super-resolution image is shown in Figure 60 b. Figure 60 c shows one of the line profiles.

For Figure 60 we corrected a linear drift. For Figure 61 b and c we used the `ThunderSTORM` Redundant Cross Correlation (RCC) drift correction to align the localisation data over

time. The FRC values of 0.211 for `CSIncetption` and 0.265 for `Rec U-Net` are a significant improvement over classical fitters like `ThunderSTORM` (0.167).
We additionally integrated an entropy minimization-based approach developed by Cnosson et al. [135] to further optimize the drift correction for difficult data.

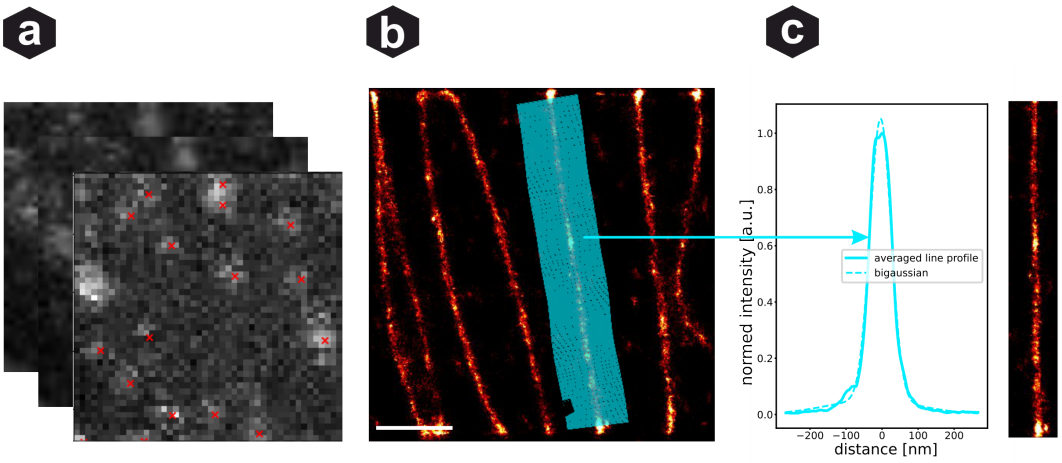

Figure 60: **Evaluation** a) Fitting of disrupted PSFs with artificial intelligence. Red crosses denote the estimated location of a fluorophore. b) Reconstructed super-resolution image of the dataset shown in a). Scale bar = 10 µm. The blue lines mark the taken line profiles of one microtubule. c) Averaged line profile from b) with a bi-Gaussian fit (dashed line). The full filament is shown on the right. marked blue in b

4.3.12 Discussion

With `ReCSAI` we developed a robust AI package for data simulation and evaluation in the context of SMLM lifetime measurements with disrupted PSFs. We introduce a trainable wavelet filter to accurately detect potential emitters and crop them in suitable ROIs. The software integrates CS into different neural network approaches to reconstruct super-resolved images out of difficult datasets.
We learned, that the sparse feature spaces of CS do not work well with the learning concept of neural networks. We speculate, that a large number of zeros lead to vanishing gradients for large fractions of the underlying feature spaces and, therefore, leads to diminished performance. The `Rec U-Net` approach circumvented this problem by using a parameterized representation of the sparse sub-domain. The algorithm achieved the best JI and RMSE on simulated data as well as the best FRC score on real data. On top of that it has a much faster training and evaluation time than the approaches that integrated a full CS algorithm. This reinforces our thoughts, that the parameterized representation of the emitter parameters is at least in the context of neural networks superior to the sparse representation. If a CS algorithm should be included in a network, a better approach could be to describe the sparse feature space in a format like compressed sparse row [137].

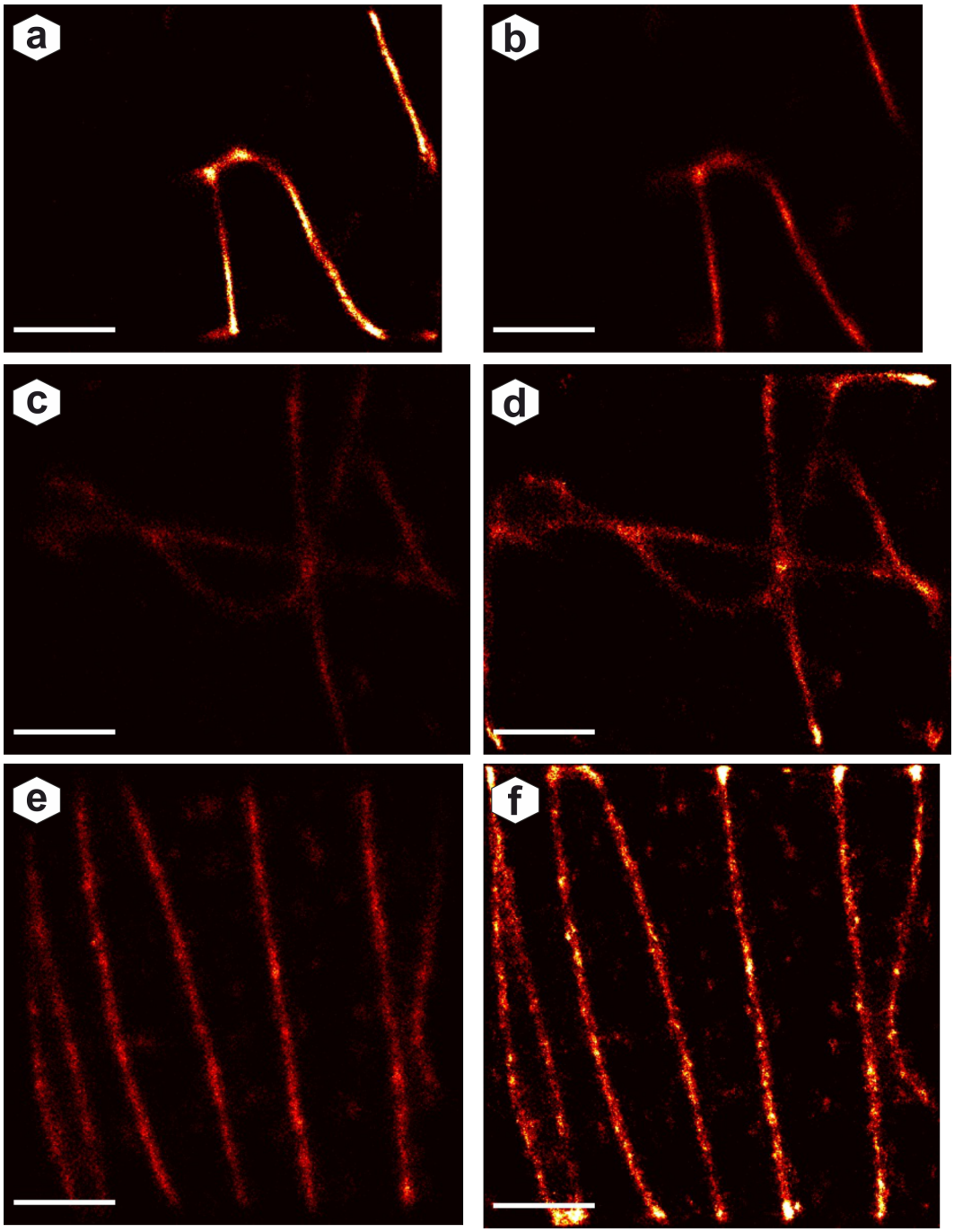

Figure 61: **Reconstruction Comparison** of three images. Left reconstructed by AI and right reconstructed using `ThunderSTORM`. We computed Fourier Ring Correlation Coefficients [136] of 0.310 (a); 0.187 (c); 0.265 (e) (left) and 0.179 (b); 0.187 (d); 0.167 (f) (right), respectively. Scale bar = 10 µm

To reduce the training and run time of the algorithm, we identified potential ROIs with a wavelet filter. The corresponding filterbank is trained on realistic simulated data. It reliably filters background frequencies by applying a learned threshold on the wavelet-transformed image. A subsequent local maxima detection delivers smaller ROIs. This makes our network scalable since the wavelet transform is image size independent and the ROIs for the following processing steps have a fixed size. The method reaches its

limits on the reconstruction of high-density samples. Multiple overlapping PSFs introduce new frequencies, which can lead to the loss of localisations. Close and separated emitters on the other hand can lead to multi-localisation, due to overlapping ROIs.
We observed, that the metrics for identical networks - apart from different CS iteration counts - converged for high training iterations. We suppose, that the network is able to compensate for the missing iterations and that the comparably low amount of iterations is not able to reveal the true benefits of the algorithm.
The computational cost for an initial FISTA layer is excessive, as shown in Table 2. The `Rec U-Net` circumvents this problem by unrolling the algorithm and integrating it better into the neural network architecture. Hence, fewer iterations on smaller feature spaces are needed to achieve good results, reducing the computational cost significantly. The approach resembles the unfolding of CS iterations in a deep neural network, as proposed by Gregor and LeCun [138]. Another similar work worth mentioning is [139], where SOFI images were reconstructed with an unrolled FISTA algorithm working in the correlation domain. Our work confirms that algorithm unfolding is an efficient way to combine the advantages of the classical approach with neural networks and we are certain that there will be many other similar applications in the future.
Currently, the limitation in creating expressive artificial networks for SMLM data is the amount of necessary training data. Creating labelled sets of real data is difficult since the precision is limited by the performance of state-of-the-art fitters and measurements are time-consuming. On the other hand, simulating large datasets solves this problem, but requires a detailed understanding of the measurement process and an accurate noise model.
As done here, the trained network is able to outperform classical approaches for the specific use case but lacks generalization. If the optical parameters or the measurement setup are changed, the network has to be retrained. Classical algorithms do not need this kind of adaption since it is already partially integrated into the model by fitting a PSF. As shown in [140] by the example of operator learning, constraints can be integrated into the network to give it prior knowledge of the underlying problem. In SMLM these constraints can be provided by adding the PSF into the network for example by our implemented CS layer. The reconstruction of the resulting sub-lattice takes the majority of optical parameters out of the equation and yields similar outputs for most measurement conditions. This can add a certain amount of generalisation, making the network applicable to more conditions. Another interesting approach already discussed by Lillicrap et al. [141] is to use an auto-encoder, to combine data creation and reconstruction in one network. Here, the feature space used in the `Rec U-Net` could be used as encoded space, while introducing an additional loss in image space, computed between the output of the decoder and a parallel classical simulation.
As previously described, the neural network fitter `DECODE` [52] is the current state of the art in all categories of the super-resolution fight club [45]. Hence, a comparison to

our approach would be highly interesting. This was, however, not possible since the creation of training data and the reconstruction process are tightly coupled. Depending, as mentioned, on accurate simulations, the standard implementation of `DECODE` is not able to work with interrupted PSFs and integrating our simulations into the evaluation process is not easily feasible. Using an architecture closely related to our `CS U-Net` approach we speculate similar performance.

5 Conclusion and outlook

5.1 LineProfiler

`LineProfiler` is a performant standalone software, that provides a graphical user interface. Packing the code with `Pyinstaller` provides easy portability. The software objectively detects filamentous structures in an image. These structures are subsequently parameterized with a C-spline. The derivative of the underlying cubic polynomials yields the direction of the filament and allows the collection of line profiles perpendicular to the spline. These are summarized and evaluated for each filament. In contrast to previous methods, `LineProfiler` incorporates the direction of the filaments and extends the sample size by collecting profiles all over the image. We were able to use the software to evaluate microtubules under different expansion protocols. We found, that the used protocol has a huge impact on the linkage error and post-expansion labelling is the method of choice if feasible. We further evaluated SC data and found, that epitopes here probably predominantly bind to the outside of the complex due to steric reasons.

`LineProfiler` reaches its limits when filamentous structures contain larger holes, caused by incomplete labelling or digestion during the expansion process. Applying a blur can only compensate for a certain amount of missing data. The same holds when the labelling occurs predominantly on one side of the strand. In summary, the limiting factor is the determination of the course of the underlying filament, which can only be determined up to a certain precision. As described in the discussion part this precision is probably already reached, using the skeletonize algorithm. However, the right degree of smoothing can improve the results. Here, an automated parameter estimation could significantly improve the results for an unfamiliar user.

In the context of SMLM it would be beneficial to interpolate splines in between localisations. This way, the reconstructed data does not have to be rendered into an image, and additional precision can be gained by using the ground data that is not interpolated over discrete pixels.

5.2 Automated Simple Elastix

`Automated Simple Elastix` is a Python script to determine the distortions occurring during ExM. It takes an image of a structure before and after expansion and determines the corresponding structural expansion factor. An initial transformation only compensates for degrees of freedom, that happen during isotropic expansion and re-imaging of the sample. A second transformation resolves the remaining shifts and draws them into a distortion map. All output images are labelled with a Pearson correlation index to grant an objective grasp of the alignment of the underlying structures. We found that expansion factor and isotropy differ for different structures. Performing 4X expansion of fungi and NK cells, the NK cells expanded less but more isotropic.
The performance of `Automated Simple Elastix` highly depends on the underlying structures. The registration works better for more complex since these contain finer patterns and, therefore, more gradients for the alignment. Microtubles for example, usually yield enough complexity to be reliably aligned, while mitoRFP needs more hyperparameter tuning and several attempts for sufficient convergence.
For further work, it might be interesting to include the software in a microscope setup with automatic scanning of the sample, since retrieving regions of the pre-expanded sample in the post-expanded sample is a big nuisance. `Elastix` should be able to align the regions, which should be sufficient to give proper feedback together with a metric for structural similarity. A drawback might be the computation time of ~ 60 `elastix` needs for the alignment of a medium-sized image. This could be targeted, by either downsampling the images or using a GPU-based modern approach like `voxelmorph` [142] for the alignment.

5.3 ReCSAI

`ReCSAI` is a software package to fit nonlinear PSFs in the context of super-resolved confocal lifetime imaging. It provides a simulator for data generation and different state-of-the-art neural network architectures. It is able to reconstruct localisations with improved accuracy compared to classical fitters.
`ReCSAI` integrates compressed sensing into deep neural networks and achieves improved computation times in comparison to classical implementations. Here, deeper architectures like inception as well as the temporal context of the previous and subsequent frame improve the model's performance significantly. Using the local context by reconstructing feature maps with the original crop size, also enhances the image quality.
The integration of compressed sensing into artificial neural networks is a promising concept, but further work needs to be done to improve the details of implementation. Optimal performance requires full convergence of the CS part of the network. This is, however, difficult since the underlying sub-lattice is sparse which leads to vanishing gradients for the huge amount of zero pixels. Algorithm unfolding on the other hand

appears to be a more efficient way to integrate compressed sensing into deep learning. As previously mentioned, the localisation of emitters underlies a physical limit that is for good data approached by standard fitters. More sophisticated approaches are necessary under special conditions, like high emitter density or nonlinear PSFs. Small amounts of additional precision can be gained if an improved estimation of the PSF, for example by measured beads, is used.
A point that is not incorporated by most algorithms, is the additional information that can be gained by taking the temporal domain into account. Processing three consecutive frames already significantly improved the results in our current work. Processing information from the whole film could further improve the results by accumulating photons from multiple blinking events. The algorithm SOFI uses autocorrelation up to the third order in the context of temporal emitter fluctuations. Since the Off-state of emitters in SMLM is much longer than three frames, there is no advantage for the reconstruction. Long short-term memory neural networks, as used in natural language processing, can process information over longer time and frame periods and could, therefore, present a suitable solution for future work.
Considering the progress in expansion microscopy, another approach would be to reconstruct emitters without the need for photo-fluorescent switching. Under tenfold expansion antibodies have a distance of $\sim 100\,\mathrm{nm}$. This is close to what algorithms like higher-order SIM or SOFI can resolve. Using SOFI in combination with a high-density fitter as done in `SPARCOM` [143] could, under the right measurement conditions, suffice to separate the underlying emitters.

5.4 Closing remarks

"Seeing is believing"

Optical microscopy is a valuable asset for biological imaging. Especially the field of techniques that surpass the resolution limit has become a rapidly evolving field in the last two decades. The corresponding methods become more and more sophisticated and involve complicated biological protocols or computational reconstructions. The limits of what is possible are fast exhausted, so it is important to take a step back from time to time and evaluate. "Seeing is believing", but sometimes what we see is not based on nature, but on reconstruction artefacts or distorted samples.
In this thesis, I presented new and enhanced algorithms for the quality assurance of super-resolved microscopy data. These can help us to trust highly processed images. Further, I developed an AI and CS-based fitter that helps us to approach the limits of what is physically possible.
Nevertheless, the potential of super-resolution microscopy is far from depleted, especially on the computational side. Applying highly accurate methods in the clinical environment requires additional automation steps and quality assurance. In the con-

text of fundamental research, the time domain yields the most promising potential for further enhancement. Expansion microscopy could enable the separation of emitters without the need for switchable dies. Bringing emitters from several switching events together could highly increase the number of collected photons and, therefore, localisation precision. The topic remains exciting. Let us see what the future brings.

Abbreviations

***d*STORM** *direct* Stochastic Optical Reconstruction Microscopy

ADC Analog to Digital Converter

AI Artificial Intelligence

ANN Artificial Neural Networks

APD Avalanche Photo Diode

CMOS Complementary Metal-Oxide-Semiconductor

CNN Convolutional Neural Networks

CPU Computational Processing Unit

CRLB Cramér-Rao Lower Bound

CS Compressed Sensing

DNA DeoxyriboNucleic Acid

DOF Degrees of Freedom

EMCCD Electron-Multiplying Charge-Coupled Device

ExM Expansion Microscopy

FISTA Fast Iterative Shrinkage Thresholding Algorithm

FLIM Fluorescence-Lifetime Imaging Microscopy

FRC Fourier Ring Correlation

FWHM Full Width at Half Maximum

GAN Generative Adversarial Network

GPU Graphics Processing Unit

GUI Graphical User Interface

ISC Inter System Crossing

ISTA Iterative Shrinkage Thresholding Algorithm

JI Jaccard Index

LUT Look Up Table

MLE Maximum Likelihood Estimation

PALM Photoactivated Localisation Microscopy

PSF Point Spread Function

RCC Redundant Cross Correlation

RMSE Root-Mean-Square Error

ROI Region Of Interest

SC Synaptonemal Complex

SGD Stochastic Gradient Descent

SIM Structured Illumination Microscopy

SMLM Single Molecule Localisation Microscopy

SNR Signal to Noise Ratio

SOFI Super-resolution Optical Fluctuation Microscopy

STED Stimulated Emission Depletion

STORM Stochastic Optical Reconstruction Microscopy

TCSPC Time-Correlated Single-Photon Counting

TIRF Total Internal Reflection Fluorescence

VAE Variational Auto Encoder

References

[1] Raasch C, Lee V, Spaeth S, Herstatt C. The rise and fall of interdisciplinary research: The case of open source innovation. Research Policy. 2013 Jun;42(5):1138-51. Available from: https://www.sciencedirect.com/science/article/pii/S0048733313000279.

[2] Okamura K. Interdisciplinarity revisited: evidence for research impact and dynamism. Palgrave Communications. 2019 Nov;5(1):1-9. Number: 1 Publisher: Palgrave. Available from: https://www.nature.com/articles/s41599-019-0352-4.

[3] Savage N. Computer logic meets cell biology: h ow cell science is getting an upgrade. Nature. 2018 Dec;564(7734):S1-3. Bandiera_abtest: a Cg_type: Spotlight Number: 7734 Publisher: Nature Publishing Group Subject_term: Cell biology, Careers, Computer science. Available from: https://www.nature.com/articles/d41586-018-07595-4.

[4] Schermelleh L, Ferrand A, Huser T, Eggeling C, Sauer M, Biehlmaier O, et al. Super-resolution microscopy demystified. Nature Cell Biology. 2019 Jan;21(1):72-84. Number: 1 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/s41556-018-0251-8.

[5] Lelek M, Gyparaki MT, Beliu G, Schueder F, Griffié J, Manley S, et al. Single-molecule localization microscopy. Nature Reviews Methods Primers. 2021 Jun;1(1):1-27. Number: 1 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/s43586-021-00038-x.

[6] Sigal YM, Zhou R, Zhuang X. Visualizing and discovering cellular structures with super-resolution microscopy. Science (New York, NY). 2018 Aug;361(6405):880-7. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6535400/.

[7] Heilemann M, vandeLinde S, Schuettpelz M, Kasper R, Seefeldt B, Mukherjee A, et al. Subdiffraction-Resolution Fluorescence Imaging with Conventional Fluorescent Probes. Angewandte Chemie International Edition. 2008 Aug;47(33):6172-6.

[8] Wolter S, Löschberger A, Holm T, Aufmkolk S, Dabauvalle MC, van de Linde S, et al. rapidSTORM: accurate, fast open-source software for localization microscopy. Nature Methods. 2012 Nov;9(11):1040-1. Number: 11 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/nmeth.2224.

[9] Ovesný M, Křížek P, Borkovec J, Švindrych Z, Hagen GM. ThunderSTORM: a comprehensive ImageJ plug-in for PALM and STORM data analysis and super-resolution imaging. Bioinformatics. 2014 Aug;30(16):2389-90. Available from: https://doi.org/10.1093/bioinformatics/btu202.

[10] Chen F, Tillberg PW, Boyden ES. Expansion microscopy. Science. 2015 Jan;347(6221):543-8. Publisher: American Association for the Advancement of Science. Available from: https://www.science.org/doi/10.1126/science.1260088.

[11] Zwettler FU, Reinhard S, Gambarotto D, Bell TDM, Hamel V, Guichard P, et al. Molecular resolution imaging by post-labeling expansion single-molecule localization microscopy (Ex-SMLM). Nature Communications. 2020 Jul;11(1):3388.

[12] Zwettler FU, Reinhard S, Sauer M. Ex-dSTORM and automated quantitative image analysis of expanded filamentous structures. Methods in Cell Biology. 2021;161:317-40.

[13] Trinks N, Reinhard S, Drobny M, Heilig L, Löffler J, Sauer M, et al. Subdiffraction-resolution fluorescence imaging of immunological synapse formation between NK cells and A. fumigatus by expansion microscopy. Communications Biology. 2021 Oct;4(1):1-12.

[14] Ernst Abbe. Beiträge zur Theorie des Mikroskops und der mikroskopischen Wahrnehmung. Archiv für Mikroskopische Anatomie. 1873 Dec;(9):413-68.

[15] Rayleigh. XXXI. Investigations in optics, with special reference to the spectroscope. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science. 1879 Oct;8(49):261-74. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/14786447908639684. Available from: https://doi.org/10.1080/14786447908639684.

[16] Kulaitis G, Munk A, Werner F. What is resolution? A statistical minimax testing perspective on super-resolution microscopy. arXiv:200507450 [math, stat]. 2020 Oct.

[17] Novotny L, Hecht B. Propagation and focusing of optical fields. In: Principles of Nano-Optics. 2nd ed. Cambridge University Press; 2012. p. 45-85.

[18] Chalfie M, Tu Y, Euskirchen G, Ward WW, Prasher DC. Green Fluorescent Protein as a Marker for Gene Expression. Science. 1994 Feb;263(5148):802-5. Publisher: American Association for the Advancement of Science. Available from: https://www.science.org/doi/10.1126/science.8303295.

[19] Griffiths DJ, Schroeter DF. Introduction to Quantum Mechanics. 3rd ed. Cambridge University Press; 2018. Available from: https://www.cambridge.org/core/product/identifier/9781316995433/type/book.

[20] Franck J, Dymond EG. Elementary processes of photochemical reactions. Trans Faraday Soc. 1926;21(February):536-42. Publisher: The Royal Society of Chemistry. Available from: http://dx.doi.org/10.1039/TF9262100536.

[21] Born M, Oppenheimer R. Zur Quantentheorie der Molekeln. Annalen der Physik. 1927;389(20):457-84. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/andp.19273892002. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1002/andp.19273892002.

[22] Schleich WP, Greenberger DM, Kobe DH, Scully MO. Schrödinger equation revisited. Proceedings of the National Academy of Sciences. 2013 Apr;110(14):5374-9. Publisher: Proceedings of the National Academy of Sciences. Available from: https://www.pnas.org/doi/10.1073/pnas.1302475110.

[23] Lakowicz JR, Masters BR. Principles of Fluorescence Spectroscopy, Third Edition. Journal of Biomedical Optics. 2008;13(2):029901. Available from: http://biomedicaloptics.spiedigitallibrary.org/article.aspx?doi=10.1117/1.2904580.

[24] Kasha M. Characterization of electronic transitions in complex molecules. Discussions of the Faraday Society. 1950 Jan;9(0):14-9. Publisher: The Royal Society of Chemistry. Available from: https://pubs.rsc.org/en/content/articlelanding/1950/df/df9500900014.

[25] Tokunaga M, Imamoto N, Sakata-Sogawa K. Highly inclined thin illumination enables clear single-molecule imaging in cells. Nature Methods. 2008 Feb;5(2):159-61. Number: 2 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/nmeth1171.

[26] Datta R, Heaster TM, Sharick JT, Gillette AA, Skala MC. Fluorescence lifetime imaging microscopy: fundamentals and advances in instrumentation, analysis, and applications. Journal of Biomedical Optics. 2020 May;25(7):071203. Publisher: SPIE. Available from: https://www.spiedigitallibrary.org/journals/journal-of-biomedical-optics/volume-25/issue-7/071203/Fluorescence-lifetime-imaging-microscopy--fundamentals-and-advances-in-instrumentation/10.1117/1.JBO.25.7.071203.full.

[27] Strutt JW. LVIII. On the scattering of light by small particles. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science. 1871 Jun;41(275):447-54. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/14786447108640507. Available from: https://doi.org/10.1080/14786447108640507.

[28] Raman CV. A new radiation [Reproduced from Indian J. Phys., 1928, 2, 387–398]. Current Science. 1998;74(4):382-6. Publisher: Temporary Publisher. Available from: http://www.jstor.org/stable/24101519.

[29] Novák T, Gajdos T, Sinkó J, Szabó G, Erdélyi M. TestSTORM: Versatile simulator software for multimodal super-resolution localization fluorescence microscopy. Scientific Reports. 2017 Apr;7(1):951. Number: 1 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/s41598-017-01122-7.

[30] Shot Noise limits the Resolution of ICCD and EMCCD Cameras;. Available from: https://stanfordcomputeroptics.com/technology/dynamic-range/photon-noise.html.

[31] Ryan DP, Dunlap MK, Gelfand MP, Werner JH, Van Orden AK, Goodwin PM. A gain series method for accurate EMCCD calibration. Scientific Reports. 2021 Sep;11(1):18348. Number: 1 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/s41598-021-97759-6.

[32] Mandracchia B, Hua X, Guo C, Son J, Urner T, Jia S. Fast and accurate sCMOS noise correction for fluorescence microscopy. Nature Communications. 2020 Dec;11(1):94. Available from: http://www.nature.com/articles/s41467-019-13841-8.

[33] The Nobel Prize in Chemistry 2014; 2022. Publication Title: NobelPrize.org. Available from: https://www.nobelprize.org/prizes/chemistry/2014/summary/.

[34] Betzig E, Patterson GH, Sougrat R, Lindwasser OW, Olenych S, Bonifacino JS, et al. Imaging Intracellular Fluorescent Proteins at Nanometer Resolution. Science. 2006 Sep;313(5793):1642-5. Publisher: American Association for the Advancement of Science. Available from: https://www.science.org/doi/10.1126/science.1127344.

[35] Hell SW, Wichmann J. Breaking the diffraction resolution limit by stimulated emission: stimulated-emission-depletion fluorescence microscopy. Optics Letters. 1994 Jun;19(11):780-2. Publisher: Optica Publishing Group. Available from: https://opg.optica.org/ol/abstract.cfm?uri=ol-19-11-780.

[36] Gustafsson MGL. Surpassing the lateral resolution limit by a factor of two using structured illumination microscopy. Journal of Microscopy. 2000;198(2):82-7. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1046/j.1365-2818.2000.00710.x. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1046/j.1365-2818.2000.00710.x.

[37] Dertinger T, Colyer R, Iyer G, Weiss S, Enderlein J. Fast, background-free, 3D super-resolution optical fluctuation imaging (SOFI). Proceedings of the National Academy of Sciences. 2009 Dec;106(52):22287-92. Publisher: Proceedings of the National Academy of Sciences. Available from: https://www.pnas.org/doi/10.1073/pnas.0907866106.

[38] Rust MJ, Bates M, Zhuang X. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM). Nature Methods. 2006 Oct;3(10):793-6.

Number: 10 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/nmeth929.

[39] Löschberger A, van de Linde S, Dabauvalle MC, Rieger B, Heilemann M, Krohne G, et al. Super-resolution imaging visualizes the eightfold symmetry of gp210 proteins around the nuclear pore complex and resolves the central channel with nanometer resolution. Journal of Cell Science. 2012 Feb;125(3):570-5. Available from: https://doi.org/10.1242/jcs.098822.

[40] Jungmann R, Steinhauer C, Scheible M, Kuzyk A, Tinnefeld P, Simmel FC. Single-Molecule Kinetics and Super-Resolution Microscopy by Fluorescence Imaging of Transient Binding on DNA Origami. Nano Letters. 2010 Nov;10(11):4756-61. Publisher: American Chemical Society. Available from: https://doi.org/10.1021/nl103427w.

[41] Ries J. SMAP: a modular super-resolution microscopy analysis platform for SMLM data. Nature Methods. 2020 Sep;17(9):870-2. Number: 9 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/s41592-020-0938-1.

[42] Huang B, Wang W, Bates M, Zhuang X. Three-Dimensional Super-Resolution Imaging by Stochastic Optical Reconstruction Microscopy. Science. 2008 Feb;319(5864):810-3. Publisher: American Association for the Advancement of Science. Available from: https://www.science.org/doi/10.1126/science.1153529.

[43] Pavani SRP, Thompson MA, Biteen JS, Lord SJ, Liu N, Twieg RJ, et al. Three-dimensional, single-molecule fluorescence imaging beyond the diffraction limit by using a double-helix point spread function. Proceedings of the National Academy of Sciences. 2009 Mar;106(9):2995-9. Publisher: Proceedings of the National Academy of Sciences. Available from: https://www.pnas.org/doi/full/10.1073/pnas.0900245106.

[44] Mortensen KI, Churchman LS, Spudich JA, Flyvbjerg H. Optimized localization analysis for single-molecule tracking and super-resolution microscopy. Nature Methods. 2010 May;7(5):377-81. Number: 5 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/nmeth.1447.

[45] Sage D, Pham TA, Babcock H, Lukes T, Pengo T, Chao J, et al. Super-resolution fight club: assessment of 2D and 3D single-molecule localization microscopy software. Nature Methods. 2019 May;16(5):387-95.

[46] Small A, Stahlheber S. Fluorophore localization algorithms for super-resolution microscopy. Nature Methods. 2014 Mar;11(3):267-79. Number: 3 Publisher: Na-

ture Publishing Group. Available from: https://www.nature.com/articles/nmeth.2844.

[47] Foucart S, Rauhut H. An Invitation to Compressive Sensing. In: Foucart S, Rauhut H, editors. A Mathematical Introduction to Compressive Sensing. Applied and Numerical Harmonic Analysis. New York, NY: Springer; 2013. p. 1-39. Available from: https://doi.org/10.1007/978-0-8176-4948-7-1.

[48] Min J, Vonesch C, Kirshner H, Carlini L, Olivier N, Holden S, et al. FALCON: fast and unbiased reconstruction of high-density super-resolution microscopy data. Scientific Reports. 2015 May;4(1):4577.

[49] Zhu L, Zhang W, Elnatan D, Huang B. Faster STORM using compressed sensing. Nature Methods. 2012 Jul;9(7):721-3.

[50] Boyd N, Schiebinger G, Recht B. The Alternating Descent Conditional Gradient Method for Sparse Inverse Problems. arXiv:150701562 [math]. 2015 Jul. ArXiv: 1507.01562. Available from: http://arxiv.org/abs/1507.01562.

[51] Nehme E, Weiss LE, Michaeli T, Shechtman Y. Deep-STORM: super-resolution single-molecule microscopy by deep learning. Optica. 2018 Apr;5(4):458-64.

[52] Speiser A, Müller LR, Hoess P, Matti U, Obara CJ, Legant WR, et al. Deep learning enables fast and dense single-molecule localization with high accuracy. Nature Methods. 2021 Sep;18(9):1082-90.

[53] Ouyang W, Aristov A, Lelek M, Hao X, Zimmer C. Deep learning massively accelerates super-resolution localization microscopy. Nature Biotechnology. 2018 May;36(5):460-8. Number: 5 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/nbt.4106.

[54] Li Y, Chen H, Cui Z, Timofte R, Pollefeys M, Chirikjian G, et al.. Towards Efficient Graph Convolutional Networks for Point Cloud Handling. arXiv; 2021. ArXiv:2104.05706 [cs]. Available from: http://arxiv.org/abs/2104.05706.

[55] Ku T, Swaney J, Park JY, Albanese A, Murray E, Cho JH, et al. Multiplexed and scalable super-resolution imaging of three-dimensional protein localization in size-adjustable tissues. Nature biotechnology. 2016 Sep;34(9):973-81. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5070610/.

[56] Chang JB, Chen F, Yoon YG, Jung EE, Babcock H, Kang JS, et al. Iterative expansion microscopy. Nature methods. 2017 Jun;14(6):593-9. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5560071/.

[57] Kopriva DA. Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers. Springer Science & Business Media; 2009. Google-Books-ID: fZyqWPNjx4AC.

[58] Szeliski R. Computer Vision. Texts in Computer Science. London: Springer London; 2011. Available from: http://link.springer.com/10.1007/978-1-84882-935-0.

[59] Glassner AS. Principles of Digital Image Synthesis. 1995:1400.

[60] Shannon CE. Communication in the Presence of Noise. Proceedings of the IRE. 1949 Jan;37(1):10-21. Publisher: Institute of Electrical and Electronics Engineers (IEEE). Available from: https://doi.org/10.1109/jrproc.1949.232969.

[61] Smith SW. The Scientist and Engineer's Guide to Digital Signal Processing. USA: California Technical Publishing; 1997.

[62] Osgood B. The Fourier transform and its applications; 2022. Available from: https://see.stanford.edu/materials/lsoftaee261/book-fall-07.pdf.

[63] Cormen TH, editor. Introduction to algorithms. 3rd ed. Cambridge, Mass: MIT Press; 2009. OCLC: ocn311310321.

[64] Soille P. Morphological Image Analysis. Berlin, Heidelberg: Springer Berlin Heidelberg; 1999. Available from: http://link.springer.com/10.1007/978-3-662-03939-7.

[65] Falcão A, Stolfi J, Lotufo R. The Image Foresting Transform: Theory, Algorithms, and Applications. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 2004 Feb;26:19 29.

[66] Lee TC, Kashyap RL, Chu CN. Building Skeleton Models via 3-D Medial Surface/Axis Thinning Algorithms. CVGIP Graph Model Image Process. 1994;56:462-78.

[67] Fiorio C, Gustedt J. Two linear time Union-Find strategies for image processing. Theoretical Computer Science. 1996 Feb;154(2):165-81. Available from: https://www.sciencedirect.com/science/article/pii/0304397594002622.

[68] Dunn KW, Kamocka MM, McDonald JH. A practical guide to evaluating colocalization in biological microscopy. American Journal of Physiology - Cell Physiology. 2011 Apr;300(4):C723-42. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3074624/.

[69] Pearson K. Note on Regression and Inheritance in the Case of Two Parents. Proceedings of the Royal Society of London. 1895;58:240-2. Publisher: The Royal Society. Available from: https://www.jstor.org/stable/115794.

[70] Manders EMM, Verbeek FJ, Aten JA. Measurement of co-localization of objects in dual-colour confocal images. Journal of Microscopy. 1993 Mar;169(3):375-82.

[71] Spearman C. The Proof and Measurement of Association between Two Things. The American Journal of Psychology. 1904;15(1):72-101. Publisher: University of Illinois Press. Available from: https://www.jstor.org/stable/1412159.

[72] Reinhard S, Aufmkolk S, Sauer M, Doose S. Registration and Visualization of Correlative Super-Resolution Microscopy Data. Biophysical Journal. 2019 Jun;116(11):2073-8.

[73] Unser M. Splines: a perfect fit for signal and image processing. undefined. 1999. Available from: https://www.semanticscholar.org/paper/Splines%3A-a-perfect-fit-for-signal-and-image-Unser/e9e366674f191444ad6f1adeb9e96b6cf0bce826.

[74] Kochenderfer MJ, Wheeler TA. Algorithms for optimization. Cambridge, Massachusetts: The MIT Press; 2019.

[75] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. arXiv; 2017. ArXiv:1412.6980 [cs]. Available from: http://arxiv.org/abs/1412.6980.

[76] Yuan W, Gao KX. EAdam Optimizer: How ϵ Impact Adam. arXiv; 2020. ArXiv:2011.02150 [cs, stat]. Available from: http://arxiv.org/abs/2011.02150.

[77] Gauß-Newton-Verfahren; 2022. Available from: https://mathepedia.de/Gausz-Newton-Verfahren.html.

[78] Vexx23. Answer to "Difference between Newton's method and Gauss-Newton method"; 2018. Available from: https://math.stackexchange.com/a/2687949.

[79] Elad M. Sparse and Redundant Representations. New York, NY: Springer New York; 2010. Available from: http://link.springer.com/10.1007/978-1-4419-7011-4.

[80] Blumensath T, Davies ME. Iterative Thresholding for Sparse Approximations. Journal of Fourier Analysis and Applications. 2008 Dec;14(5):629-54. Available from: https://doi.org/10.1007/s00041-008-9035-z.

[81] Beck A, Teboulle M. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. SIAM Journal on Imaging Sciences. 2009 Jan;2(1):183-202.

[82] TURING AM. I.—COMPUTING MACHINERY AND INTELLIGENCE. Mind. 1950 Oct;LIX(236):433-60. Available from: https://doi.org/10.1093/mind/LIX.236.433.

[83] Block N. Psychologism and Behaviorism. Philosophical Review. 1981;90(1):5-43. Publisher: Duke University Press.

[84] Searle JR. Minds, brains, and programs. Behavioral and Brain Sciences. 1980 Sep;3(3):417-24. Publisher: Cambridge University Press. Available from: https://www.cambridge.org/core/journals/behavioral-and-brain-sciences/article/abs/minds-brains-and-programs/DC644B47A4299C637C89772FACC2706A.

[85] Butz MV. Towards Strong AI. KI - Künstliche Intelligenz. 2021 Mar;35(1):91-101. Available from: https://doi.org/10.1007/s13218-021-00705-x.

[86] Fjelland R. Why general artificial intelligence will not be realized. Humanities and Social Sciences Communications. 2020 Jun;7(1):1-9. Number: 1 Publisher: Palgrave. Available from: https://www.nature.com/articles/s41599-020-0494-4.

[87] Lucci S, Kopec D. Artificial Intelligence in the 21st Century. Mercury Learning & Information; 2015. Available from: https://books.google.de/books?id=wje_DgAAQBAJ.

[88] Mcculloch WS, Pitts W. A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY:17.

[89] Sharma S, Sharma S, Athaiya A. ACTIVATION FUNCTIONS IN NEURAL NETWORKS. International Journal of Engineering Applied Sciences and Technology. 2020 May;04(12):310-6. Available from: https://www.ijeast.com/papers/310-316,Tesma412,IJEAST.pdf.

[90] Nwankpa C, Ijomah W, Gachagan A, Marshall S. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. arXiv; 2018. ArXiv:1811.03378 [cs]. Available from: http://arxiv.org/abs/1811.03378.

[91] Nair V, Hinton GE. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. ICML'10. Madison, WI, USA: Omnipress; 2010. p. 807-14.

[92] Maas AL, Hannun AY, Ng AY. Rectifier nonlinearities improve neural network acoustic models. In: in ICML Workshop on Deep Learning for Audio, Speech and Language Processing; 2013. .

[93] Ding B, Qian H, Zhou J. Activation functions and their characteristics in deep neural networks. In: 2018 Chinese Control And Decision Conference (CCDC); 2018. p. 1836-41. ISSN: 1948-9447.

[94] Kullback S, Leibler RA. On Information and Sufficiency. The Annals of Mathematical Statistics. 1951 Mar;22(1):79-86. Publisher: Institute of Mathematical Statistics. Available from: https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-22/issue-1/On-Information-and-Sufficiency/10.1214/aoms/1177729694.full.

[95] Lippmann RP. An Introduction'to Computing with Neural Nets:19.

[96] Goh ATC. Back-propagation neural networks for modeling complex systems. Artificial Intelligence in Engineering. 1995 Jan;9(3):143-51. Available from: https://linkinghub.elsevier.com/retrieve/pii/095418109400011S.

[97] Rumelhart DE, Hinton GE, Williams RJ. Learning internal representations by error propagation. In: Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations. Cambridge, MA, USA: MIT Press; 1986. p. 318-62.

[98] Dowat J. Internal Covariate Shift: An Overview of How to Speed up Neural Network Training; 2021. Available from: https://medium.com/analytics-vidhya/internal-covariate-shift-an-overview-of-how-to-speed-up-neural-network-training-3e2a3dcdd5cc.

[99] Shimodaira H. Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of Statistical Planning and Inference. 2000 Oct;90(2):227-44. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0378375800001154.

[100] Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. In: Advances in Neural Information Processing Systems. vol. 25. Curran Associates, Inc.; 2012. Available from: https://papers.nips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

[101] Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. arXiv; 2012. ArXiv:1207.0580 [cs]. Available from: http://arxiv.org/abs/1207.0580.

[102] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: Proceedings of the 32nd International Conference on Machine Learning. PMLR; 2015. p. 448-56. ISSN: 1938-7228. Available from: https://proceedings.mlr.press/v37/ioffe15.html.

[103] LeCun Y, Bottou L, Bengio Y, Ha P. Gradient-Based Learning Applied to Document Recognition. 1998:46.

[104] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al. Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2015. p. 1-9. ISSN: 1063-6919.

[105] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the Inception Architecture for Computer Vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016. p. 2818-26.

[106] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE; 2016. p. 770-8.

[107] He K, Zhang X, Ren S, Sun J. Identity Mappings in Deep Residual Networks. arXiv; 2016. ArXiv:1603.05027 [cs]. Available from: http://arxiv.org/abs/1603.05027.

[108] Kingma DP, Welling M. Auto-Encoding Variational Bayes. arXiv; 2014. ArXiv:1312.6114 [cs, stat]. Available from: http://arxiv.org/abs/1312.6114.

[109] White T. Sampling Generative Networks. arXiv; 2016. ArXiv:1609.04468 [cs, stat]. Available from: http://arxiv.org/abs/1609.04468.

[110] Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al.. Generative Adversarial Networks. arXiv; 2014. ArXiv:1406.2661 [cs, stat]. Available from: http://arxiv.org/abs/1406.2661.

[111] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab N, Hornegger J, Wells WM, Frangi AF, editors. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Lecture Notes in Computer Science. Cham: Springer International Publishing; 2015. p. 234-41.

[112] Weber K, Rathke PC, Osborn M. Cytoplasmic microtubular images in glutaraldehyde-fixed tissue culture cells by electron microscopy and by immunofluorescence microscopy. Proceedings of the National Academy of Sciences. 1978 Apr;75(4):1820-4. Available from: https://pnas.org/doi/full/10.1073/pnas.75.4.1820.

[113] Van der Walt S, Schönberger JL, Nunez-Iglesias J, Boulogne F, Warner JD, Yager N, et al. scikit-image: image processing in Python. PeerJ. 2014;2:e453. Publisher: PeerJ Inc.

[114] Bradski G. The OpenCV Library. Dr Dobb's Journal of Software Tools. 2000.

[115] Harris CR, Millman KJ, Walt SJvd, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature. 2020 Sep;585(7825):357-62. Publisher: Springer Science and Business Media LLC. Available from: https://doi.org/10.1038/s41586-020-2649-2.

[116] PyQT. PyQt Reference Guide. 2012. Available from: http://www.riverbankcomputing.com/static/Docs/PyQt4/html/index.html.

[117] Otsu N. A Threshold Selection Method from Gray-Level Histograms. IEEE Transactions on Systems, Man, and Cybernetics. 1979 Jan;9(1):62-6. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics.

[118] Luna J. Top programming languages for data scientists in 2022; 2022. Available from: https://www.datacamp.com/blog/top-programming-languages-for-data-scientists-in-2022.

[119] Ajitsaria A. What Is the Python Global Interpreter Lock (GIL)? – Real Python; 2017. Available from: https://realpython.com/python-gil/.

[120] Zwettler FU, Spindler MC, Reinhard S, Klein T, Kurz A, Benavente R, et al. Tracking down the molecular architecture of the synaptonemal complex by expansion microscopy. Nature Communications. 2020 Jun;11(1):3222.

[121] Shamonin D, Bron E, Lelieveldt B, Smits M, Klein S, Staring M. Fast Parallel Image Registration on CPU and GPU for Diagnostic Classification of Alzheimer's Disease. Frontiers in Neuroinformatics. 2014;7. Available from: https://www.frontiersin.org/articles/10.3389/fninf.2013.00050.

[122] Klein S, Staring M, Murphy K, Viergever MA, Pluim JPW. elastix: A Toolbox for Intensity-Based Medical Image Registration. IEEE Transactions on Medical Imaging. 2010 Jan;29(1):196-205. Conference Name: IEEE Transactions on Medical Imaging.

[123] Marstal K, Berendsen F, Staring M, Klein S. SimpleElastix: A User-Friendly, Multi-lingual Library for Medical Image Registration. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2016. p. 574-82. ISSN: 2160-7516.

[124] Beare R, Lowekamp B, Yaniv Z. Image Segmentation, Registration and Characterization in R with SimpleITK. Journal of Statistical Software. 2018 Sep;86:1-35. Available from: https://doi.org/10.18637/jss.v086.i08.

[125] Büttner M, Lagerholm CB, Waithe D, Galiani S, Schliebs W, Erdmann R, et al. Challenges of Using Expansion Microscopy for Super-resolved Imaging of Cellular Organelles. ChemBioChem. 2021;22(4):686-93. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cbic.202000571. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1002/cbic.202000571.

[126] Astropy Collaboration, Robitaille TP, Tollerud EJ, Greenfield P, Droettboom M, Bray E, et al. Astropy: A community Python package for astronomy. åp. 2013 Oct;558:A33. _eprint: 1307.6212.

[127] Douglass KM. Modeling noise for image simulations; 2017. Available from: http://kmdouglass.github.io/posts/modeling-noise-for-image-simulations/.

[128] Dibal PY, Onwuka E, Agajo J, Alenoghena C. Analysis of Wavelet Transform Design via Filter Bank Technique. IntechOpen; 2019.

[129] Daubechies I. Ten lectures on wavelets. USA: Society for Industrial and Applied Mathematics; 1992.

[130] Le J, Tian Y, Mendes J, Wilson B, Ibrahim M, DiBella E, et al. Deep learning for radial SMS myocardial perfusion reconstruction using the 3D residual booster U-net. Magnetic Resonance Imaging. 2021 Nov;83:178-88. Available from: https://www.sciencedirect.com/science/article/pii/S0730725X21001375.

[131] Mizusawa S, Sei Y, Orihara R, Ohsuga A. Computed tomography image reconstruction using stacked U-Net. Computerized Medical Imaging and Graphics. 2021 Jun;90:101920. Available from: https://www.sciencedirect.com/science/article/pii/S0895611121000690.

[132] Lu X, Dong W, Wang P, Shi G, Xie X. ConvCSNet: A Convolutional Compressive Sensing Framework Based on Deep Learning. arXiv:180110342 [cs]. 2018 Jan.

[133] Carrasco OC. Gaussian Mixture Models Explained; 2020. Available from: https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95.

[134] Murphy KP. Machine learning: a probabilistic perspective. Adaptive computation and machine learning series. Cambridge, MA: MIT Press; 2012.

[135] Cnossen J, Cui TJ, Joo C, Smith C. Drift correction in localization microscopy using entropy minimization. Optics Express. 2021 Aug;29(18):27961-74. Publisher: Optica Publishing Group. Available from: https://opg.optica.org/oe/abstract.cfm?uri=oe-29-18-27961.

[136] Berberich A, Kurz A, Reinhard S, Paul TJ, Burd PR, Sauer M, et al. Fourier Ring Correlation and Anisotropic Kernel Density Estimation Improve Deep Learning Based SMLM Reconstruction of Microtubules. Frontiers in Bioinformatics. 2021;1:55.

[137] Bell N, Garland M. Efficient Sparse Matrix-Vector Multiplication on CUDA. Nvidia Technical Report NVR-2008-004, Nvidia Corporation. 2008 Dec;2(5):32.

[138] Gregor K, LeCun Y. Learning fast approximations of sparse coding. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. ICML'10. Madison, WI, USA: Omnipress; 2010. p. 399-406.

[139] Dardikman-Yoffe G, Eldar YC. Learned SPARCOM: unfolded deep super-resolution microscopy. Optics Express. 2020 Sep;28(19):27736-63.

[140] Maier AK, Syben C, Stimpel B, Würfl T, Hoffmann M, Schebesch F, et al. Learning with known operators reduces maximum error bounds. Nature Machine Intelligence. 2019 Aug;1(8):373-80. Number: 8 Publisher: Nature Publishing Group. Available from: https://www.nature.com/articles/s42256-019-0077-5.

[141] Wu Y, Rosca M, Lillicrap T. Deep Compressed Sensing. arXiv:190506723 [cs, eess, stat]. 2019 May. ArXiv: 1905.06723. Available from: http://arxiv.org/abs/1905.06723.

[142] Balakrishnan G, Zhao A, Sabuncu MR, Guttag J, Dalca AV. VoxelMorph: A Learning Framework for Deformable Medical Image Registration. IEEE Transactions on Medical Imaging. 2019 Aug;38(8):1788-800. ArXiv:1809.05231 [cs]. Available from: http://arxiv.org/abs/1809.05231.

[143] Solomon O, Mutzafi M, Segev M, Eldar YC. Sparsity-based super-resolution microscopy from correlation information. Optics Express. 2018 Jul;26(14):18238-69.

Acknowledgements

I want to thank Markus Sauer for giving me the opportunity to do my thesis at his lab. Thanks to your support I was able to find my passion for programming and optimize my skills since the days of my bachelor thesis. With the Sauer lab, you created a great place for free and competitive science. Your leadership is an inspiration for the whole department.
For his expertise and all the fruitful discussions as well as his advice on all kinds of topics I want to thank Philip Kollmannsberger. Much of what I know today about artificial intelligence is thanks to your guidance. I especially enjoy the ongoing talks and literature seminars for current and past group members.
I thank Sören Doose for his support during my master's thesis and for helping me out during the early stages of my programming career.
For the great collaborations, I want to thank Fabian Zwettler, Nora Trinks, Dominic Helmerich, Dominik Boras and the remaining members of the Sauer lab. This thesis would not have been possible without your biological know-how and the hours you spend behind the microscope to create magnificent images.

I want to thank my family, especially my parents Heide and Günther Reinhard for their unconditional support in every aspect of my life. You granted me the freedom to develop myself into the person I am today.
I thank my dad for his motto "Fordern statt verwöhnen". I remember him telling me in the context of staying up late "The world is unfair. You're just unlucky to have strict parents." I think he was wrong. There are no better parents and I couldn't have had more luck in this regard. I wish he was still here to celebrate the completion of this thesis with me. You are greatly missed.
I thank my mom for her very own way of supporting me. Your love and kindness always give me fresh energy to further pursue my goals. Thanks for your inspiration and your support in every major event of my life. Be it in sports, work or private life. As long as you are cheering on me I'll always make it to the finish line.
Thank you Sarah for your everlasting support in difficult and good times. Sarah always got me back up on my feet when I was down, especially after the death of my father. During Corona, we spend 24/7 with each other. But instead of getting annoyed, we enriched each other with ideas, sports and projects. I don't think there is another person in this world I would rather spend time with.

Eidesstattliche Erklärungen nach §7 Abs. 2 Satz 3, 4, 5 der Promotionsordnung der Fakultät für Biologie

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, die Dissertation: „Verbesserung von Datenrekonstruktion und -auswertung in der Super-Resolution Mikroskopie durch die Entwicklung von fortgeschrittenen Verarbeitungsalgorithmen und künstlichen neuronalen Netzen“, eigenständig, d. h. insbesondere selbständig und ohne Hilfe eines kommerziellen Promotionsberaters, angefertigt und keine anderen, als die von mir angegebenen Quellen und Hilfsmittel verwendet zu haben.
Ich erkläre außerdem, dass die Dissertation weder in gleicher noch in ähnlicher Form bereits in einem anderen Prüfungsverfahren vorgelegen hat.
Weiterhin erkläre ich, dass bei allen Abbildungen und Texten bei denen die Verwertungsrechte (Copyright) nicht bei mir liegen, diese von den Rechtsinhabern eingeholt wurden und die Textstellen bzw. Abbildungen entsprechend den rechtlichen Vorgaben gekennzeichnet sind sowie bei Abbbildungen, die dem Internet entnommen wurden, der entsprechende Hypertextlink angegeben wurde.

Affidavit

I hereby declare that my thesis entitled: „Improving Super-Resolution Microscopy Data Reconstruction and Evaluation by Developing Advanced processing Algorithms and Artificial Neuronal Networks” is the result of my own work. I did not receive any help or support from commercial consultants. All sources and / or materials applied are listed and specified in the thesis.
Furthermore I verify that the thesis has not been submitted as part of another examination process neither in identical nor in similar form.
Besides I declare that if I do not hold the copyright for figures and paragraphs, I obtained it from the rights holder and that paragraphs and figures have been marked according to law or for figures taken from the internet the hyperlink has been added accordingly.

$Würzburg, den$
$Unterschrift:$