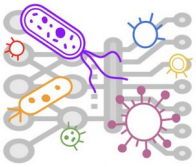
Politecnico di Torino



Politecnico di Torino

Physics of Complex Systems
A.a. 2025/2026
Sessione di Laurea Luglio 2026

Machine Learning and Uncertainty Quantification for Single Particle Tracking

Short-Trajectory Analysis, Diffusion Parameters and Uncertainty

Relatori:
Luca Barbiero
Nataliya Sokolovska
Judith Mine-Hattab

Candidato:
Mario Catella

Abstract

This thesis studies the use of machine learning for the analysis of Single Particle Tracking (SPT) trajectories, with particular attention to the short-trajectory regime. The goal is to infer diffusion parameters, detect local mobility changes and quantify predictive uncertainty from individual molecular trajectories. The first part introduces the SPT experimental pipeline, the physical meaning of Brownian and anomalous diffusion, and the main neural architectures used for sequence analysis. The second part focuses on conformal prediction and on its role as a post-hoc calibration method for neural-network outputs.

The final analysis considers simulated two-dimensional trajectories of length $T = 16$. Trajectories are represented through displacement increments and are processed with CNN and CNN–Transformer models. The CNN–Transformer model is used to predict frame-wise profiles of α_t and $\log_{10}(D_t)$, which are then used for changepoint detection through KernelCPD. The model reaches frame-wise MAE values of 0.16 ± 0.015 for α and 0.12 ± 0.018 for $\log_{10}(D)$, with uncertainties reported as standard deviations. Split conformal prediction applied to the $\log_{10}(D_t)$ profile gives an empirical coverage of 97.60% for a nominal 95% level.

Throughout the thesis, D is treated as the main physical observable because it is directly linked to molecular mobility, binding, confinement and changes in protein-complex state. The anomalous exponent α is used as a complementary descriptor of the transport regime. The results suggest that frame-wise prediction of $\log_{10}(D_t)$, combined with kernel changepoint detection and conformal calibration, is a useful strategy for analyzing short SPT trajectories while avoiding overconfident point predictions.

Contents

Chapter 1

Single Particle Tracking and State of the Art

1.1 Single Particle Tracking in the Context of Super-Resolution Microscopy

Single Particle Tracking (SPT) is a live-cell super-resolution microscopy approach that reconstructs the motion of individual labelled molecules over time. The objective is not only to observe where a molecule is, but to quantify how it explores its cellular environment and how this exploration changes with binding, crowding, confinement or active processes. The review and protocol by Kobayashi et al. emphasizes this point: protein diffusion and molecular interactions are central to cell function, and SPT provides a way to access these dynamics at the level of single molecules rather than only through ensemble averages [5, 1, 2, 3].

In conventional optical microscopy, the spatial resolution is limited by diffraction. A fluorescent molecule is not recorded as a mathematical point but as a diffraction-limited spot whose width is on the order of hundreds of nanometers. A common estimate of the lateral resolution is

$$\Delta r \simeq \frac{0.6\lambda}{\mathrm{NA}}, \tag{1.1}$$

where λ is the emitted wavelength and NA is the numerical aperture of the objective. For visible light and high numerical aperture objectives this gives a scale around 200–300 nm. If many fluorophores are simultaneously emitting inside this distance, their images overlap and they cannot be resolved as individual molecules [6, 5].

Single Molecule Localization Microscopy (SMLM) overcomes this limitation by ensuring that only a sparse subset of fluorophores is visible in each frame. If the emitting molecules are sufficiently separated, each spot can be fitted individually and localized with a precision much better than the diffraction limit. PALM and STORM use this principle mainly to reconstruct structures in fixed samples, whereas SPT applies the same localization logic to living cells to obtain time-resolved trajectories of individual molecules [7, 8, 9, 5]. In two dimensions the final object is a sequence

$$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T, \qquad \mathbf{x}_t = (x_t, y_t), \tag{1.2}$$

sampled at the camera frame rate. Single Particle Tracking follows individual molecules in living cells, with typical resolutions reaching about 20 nm in space and 10 ms in time in optimized experiments. The protocol reports typical live-cell SPT conditions with high temporal and spatial resolution, for example acquisition around 100 Hz and localization on the order of tens of nanometers [5]. Related experiments are also carried out by the team of Judith Mine-Hattab at Sorbonne Université using two microscope setups: a Nikon microscope with a 100×, NA = 1.49 objective and an Andor EMCCD camera, and an Abbelight microscope with a 100×, NA = 1.49 objective and a Hamamatsu Orca Fusion BT camera.

The central advantage of SPT is that it preserves single-molecule heterogeneity. FRAP, FLIP, FLIM and related fluorescence approaches can measure mobility in living cells, but they often return ensemble quantities such as an average diffusion coefficient, a fluorescence recovery or decay time, or a residual immobile fraction. SPT instead produces distributions of trajectories. This makes it possible to identify freely diffusing molecules, confined molecules, transiently bound molecules, and molecules that change state during a recording [10, 11, 5]. Figure 1.1 shows an example of diffusion maps obtained by the team of Judith Mine-Hattab at Sorbonne Université. The maps illustrate how SPT can turn many individual tracks into spatially resolved mobility information: the colors encode values of the diffusion coefficient, ranging from approximately $10^{-3}\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$ in slow regions to approximately $5\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$ in fast regions. Different proteins therefore show visibly different mobility patterns.

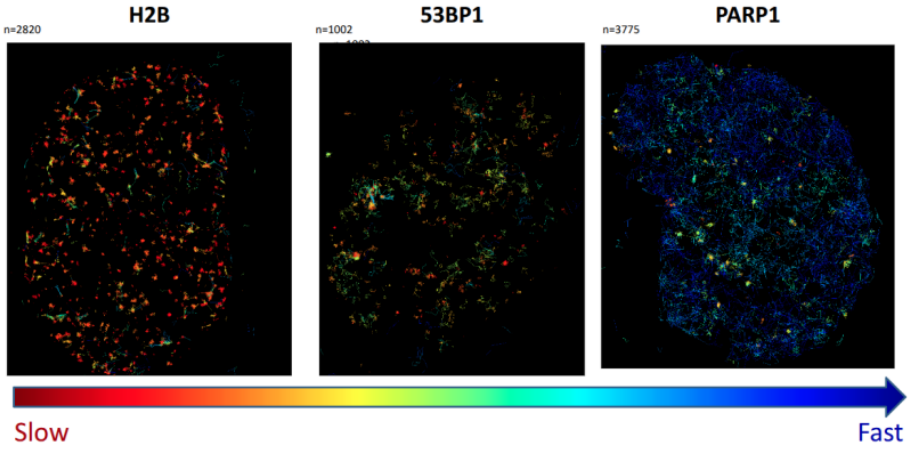


Figure 1.1: Examples of diffusion maps obtained by the team of Judith Mine-Hattab at Sorbonne Université. The panels show H2B, 53BP1 and PARP1. The color scale encodes values of the diffusion coefficient, from slow regions around $10^{-3}\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$ to fast regions around $5\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$. These maps illustrate the type of spatially resolved dynamical information that can be extracted from single-particle trajectories.

Figure 1.2 summarizes the same experimental and computational logic in a compact visual form, from sample preparation and image acquisition to localization, trajectory linking and diffusion analysis.

Table 1.1 summarizes the main fluorescence-microscopy acronyms used in this context.

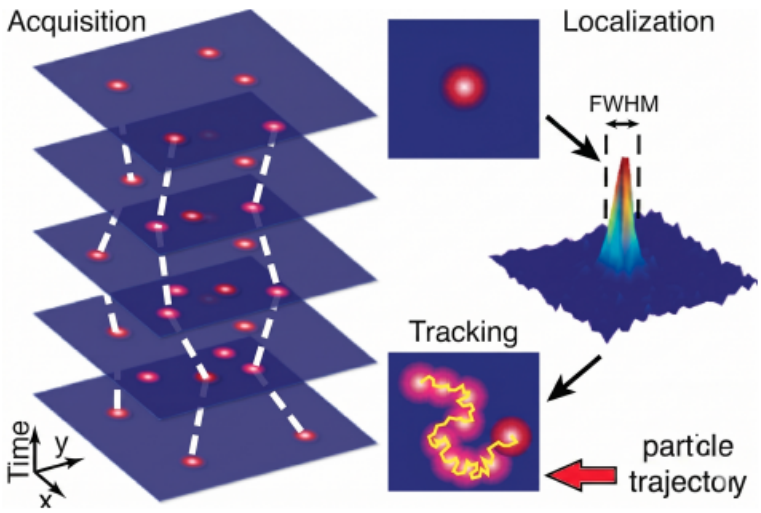


Figure 1.2: Conceptual SPT pipeline: sample preparation, live-cell acquisition, spot detection, sub-pixel localization, trajectory linking and diffusion analysis. The figure is taken from Manley et al. [9].

Acronym	Full name	Main idea
PALM	Photoactivated Localization Microscopy	Sparse photoactivation and localization of individual fluorophores are used to reconstruct a super-resolved molecular map.
STORM	Stochastic Optical Reconstruction Microscopy	Stochastic switching or blinking of fluorophores separates nearby emitters in time and enables super-resolution reconstruction.
FLIP	Fluorescence Loss In Photobleaching	A region is repeatedly photobleached while fluorescence loss elsewhere is monitored to probe molecular mobility and connectivity.
FLAP	Fluorescence Loss After Photoactivation	A selected region is photoactivated and the subsequent loss or redistribution of fluorescence is followed to estimate exchange dynamics.
FLIM	Fluorescence Lifetime Imaging Microscopy	Fluorescence decay times are measured instead of only intensity, providing contrast related to the local molecular environment.

Table 1.1: Brief summary of fluorescence-microscopy techniques relevant to the SPT context. PALM and STORM are localization-based super-resolution methods, while FLIP, FLAP and FLIM are ensemble or local fluorescence assays used to probe mobility, exchange or molecular environment [7, 8, 10].

1.2 Experimental Workflow and Acquisition Constraints

The SPT workflow is both experimental and computational. Following the structure of the protocol by Kobayashi et al., one can separate it into four stages:

$$\text{sample preparation} \longrightarrow \text{image acquisition} \longrightarrow \text{trajectory inference} \longrightarrow \text{diffusion analysis}.$$

In fluorescence-based live-cell SPT, the protein of interest is commonly fused to a tag, for example HaloTag, and labelled with a cell-permeable dye such as a Janelia Fluor dye. Stable or endogenous tagging is preferable when possible, because the labelling strategy itself should not strongly perturb the protein dynamics. Controls are important: a tag that changes protein size, expression level, localization or function can modify the very mobility that the experiment is designed to measure [12, 5].

This point is more than a biological precaution: it affects the statistical meaning of the inferred diffusion coefficient. If the tagged construct is over-expressed, if the fluorescent dye remains nonspecifically bound to the cell, or if the label changes the interaction surface of the protein, the resulting trajectories no longer describe the native protein population. The protocol therefore recommends validating the tagged protein, controlling background fluorescence and using negative controls in which the dye is applied to non-tagged cells. Phenol-red-free imaging media, careful washing, and a first inspection with conventional fluorescence microscopy are practical ways to reduce autofluorescence and false-positive detections before SPT acquisition [5].

During acquisition, the sample must remain alive and stable. Temperature control, focus stabilization and careful illumination are not secondary details: they determine the quality of the trajectories that will later be analyzed. The protocol describes live human-cell imaging at 37°C, the use of high numerical aperture oil objectives, sensitive cameras, photo-activation lasers and highly inclined illumination. HILO microscopy is particularly useful for thick cellular samples because it reduces out-of-focus fluorescence and improves the sample/background contrast compared with standard wide-field illumination [13, 5].

A successful SPT movie requires a delicate balance. If too many molecules are activated in the same frame, individual spots overlap and linking becomes ambiguous. If too few photons are collected, localization precision becomes poor. If laser power is too high, photobleaching and phototoxicity increase. If laser power is too low, the signal-to-background ratio may be insufficient. In practice, the 405 nm activation laser is adjusted so that only a small fraction of fluorophores emits at any instant. For abundant proteins, pulsed or very weak activation may be needed to keep the spot density low enough for single-particle tracking [5].

The main acquisition-quality variables can be summarized as follows:

$$\text{quality} = f(N_\gamma, B, \rho, \sigma_{\text{loc}}, v, \tau_{\text{exp}}, p_{\text{blink}}, p_{\text{bleach}}), \tag{1.3}$$

where N_γ is the number of detected photons, B the background, ρ the density of active emitters, σ_{loc} the localization error, v the typical molecular speed, τ_{exp} the exposure time, and p_{blink}, p_{bleach} represent blinking and photobleaching effects. These variables are coupled. For example, increasing exposure time can increase photon count, but it also increases motion blur for fast molecules and may increase photodamage. Increasing activation improves the number of observed molecules, but also increases spot overlap and mis-linking. Thus, the quality of SPT data is already partly determined before any algorithm is applied.

Several experimental parameters are therefore chosen with the downstream tracking problem in mind. Restricting the region of interest can increase the camera frame rate and reduce unnecessary illumination of the sample. Weak or pulsed 405 nm activation helps maintain sparse single-molecule emission, especially for abundant proteins. Focus stabilization limits axial drift, while fiducial beads or other reference objects can be used to diagnose and correct mechanical or thermal drift. Kobayashi et al. also note simple but useful quality checks, such

as replaying the movie backward at the end of acquisition to reveal systematic drift. For long acquisitions, environmental stability becomes part of the measurement: temperature, humidity and CO_2 control help maintain cell physiology, whereas excessive laser exposure can produce phototoxicity and change the dynamics being measured [5].

Issue	Effect on trajectories	Practical mitigation
High spot density	Overlap and wrong links	Lower or pulse activation
Low photon count	Poor localization precision	Tune exposure and laser power
High background	False detections	HILO, washing, clean medium
Motion blur	Biased short-lag displacements	Shorter exposure or smaller ROI
Drift	Apparent directed motion	Focus lock, beads, drift correction
Blinking/bleaching	Gaps and short tracks	Dye and illumination optimization

Table 1.2: Typical SPT acquisition failure modes and their effect on trajectory analysis, following the practical recommendations discussed by Kobayashi et al. [5].

1.3 Detection and Sub-Pixel Localization

The first computational step is the detection of fluorescent spots in each frame. Classical methods identify local intensity maxima after filtering, while more recent approaches may use convolutional neural networks for detection in challenging images [5]. Once a candidate spot has been detected, its center is estimated at sub-pixel precision. This is possible because the observed spot is a point spread function (PSF), not a single pixel. A simplified image model is

$$I(\mathbf{r}) = B(\mathbf{r}) + \sum_{j=1}^{N_t} A_j h(\mathbf{r} - \mathbf{x}_{j,t}) + \varepsilon(\mathbf{r}), \tag{1.4}$$

where $I(\mathbf{r})$ is the observed intensity, B is the background, h is the PSF, $\mathbf{x}_{j,t}$ is the molecular position, A_j is the signal amplitude and ε represents noise. In two-dimensional localization, the PSF is often approximated by a Gaussian:

$$h(\mathbf{r} - \mathbf{x}_0) \propto \exp\left(-\frac{\|\mathbf{r} - \mathbf{x}_0\|^2}{2\sigma_{\mathrm{PSF}}^2}\right). \tag{1.5}$$

The localization algorithm estimates $\mathbf{x}_0$ by centroid fitting, Gaussian fitting or maximum-likelihood fitting.

The reason sub-pixel localization is possible is that the center of the PSF is estimated from many photons distributed over several pixels. The pixel grid sets the sampling, not the final precision. In ideal conditions, increasing the number of collected photons decreases the uncertainty of the fitted center, but experimental SPT is rarely ideal: background fluorescence, camera noise, overlapping emitters and finite exposure time all broaden the effective uncertainty. For fast molecules, the molecule can move appreciably during one camera exposure, so the recorded PSF is closer to a time average of the path than to an instantaneous position. This motion blur is one reason why increasing exposure time is not always beneficial, even if it increases photon count [5, 23].

Localization precision is limited by photon statistics, camera noise, background fluorescence, pixel size and motion blur. At the level of the trajectory model, the measured position is better written as

$$\mathbf{y}_t = \mathbf{x}_t + \boldsymbol{\eta}_t, \qquad \boldsymbol{\eta}_t \sim \mathcal{N}(0, \sigma_{\mathrm{loc}}^2 I_d), \tag{1.6}$$

where $\mathbf{x}_t$ is the true molecular position and $\mathbf{y}_t$ is the localized position. This distinction is crucial. Diffusion models describe the dynamics of $\mathbf{x}_t$, while algorithms and neural networks

usually receive $\mathbf{y}_t$. If σ_{loc} is not accounted for, the short-lag MSD can be biased upward and the inferred diffusion parameters can be distorted [14, 15, 17, 23].

1.4 Linking Localizations into Trajectories

After localization, the problem becomes temporal. Detections in consecutive frames must be assigned to physical molecules. This step is difficult when spot density is high, molecules diffuse quickly, emitters blink, particles leave the focal plane, or false detections appear in the background. Kobayashi et al. emphasize that molecular density, signal-to-noise ratio, defocalization, blinking and the number of user-defined parameters should influence the choice of tracking software [5].

Formally, if frame t contains localizations $\{\mathbf{y}_{i,t}\}_{i=1}^{n_t}$ and frame $t+1$ contains $\{\mathbf{y}_{j,t+1}\}_{j=1}^{n_{t+1}}$, linking can be written as an assignment problem. A simple Brownian cost is

$$c_{ij}^{(t)} = \frac{\|\mathbf{y}_{j,t+1} - \mathbf{y}_{i,t}\|^2}{4D\Delta t}, \tag{1.7}$$

possibly with additional penalties for gaps, blinking, track birth and track termination. The algorithm searches for an assignment a minimizing

$$\min_a \sum_i c_{i,a(i)}^{(t)} \tag{1.8}$$

under uniqueness and maximum-displacement constraints. Nearest-neighbor tracking is fast and simple, but becomes fragile at high density. More robust methods use global optimization, Kalman filtering or likelihood-based combinatorial reconnection [18, 19, 20].

Linking errors are not harmless preprocessing noise. A wrong link creates an artificial displacement, while a missed link shortens the trajectory. Both effects can bias the estimated diffusion coefficient, anomalous exponent and changepoint position. This is why SPT analysis should be viewed as a full pipeline: sample preparation, acquisition, localization, linking and physical inference are statistically connected.

1.5 From Trajectories to Diffusion Observables

Once a trajectory is reconstructed, the analysis can be performed at ensemble level or individual level. Ensemble-level analysis groups trajectories and estimates population properties. It is robust when trajectories are very short, but it can hide heterogeneous subpopulations. Individual-level analysis assigns parameters to each trajectory or to each segment of a trajectory. It is more informative, but also harder because each estimate is based on few observations [5].

The main mathematical representations are

$$\{\mathbf{x}_t\}_{t=1}^{T}, \qquad \{\Delta\mathbf{x}_t\}_{t=1}^{T-1}, \qquad \{\|\Delta\mathbf{x}_t\|^2\}_{t=1}^{T-1}, \tag{1.9}$$

with

$$\Delta\mathbf{x}_t = \mathbf{x}_{t+1} - \mathbf{x}_t. \tag{1.10}$$

Absolute coordinates are useful for spatial maps and density analysis, increments are useful for local dynamics, and squared displacements enter MSD estimators. In the computational work described in this thesis, neural networks mainly receive increments, because the initial absolute position is often irrelevant for estimating D, α or a changepoint.

For a set of trajectories, the ensemble MSD in one dimension can be written as

$$\text{MSD}(t) = \frac{1}{N}\sum_{i=1}^{N} \left(x_i(t) - x_i(0)\right)^2. \tag{1.11}$$

For one trajectory, the time-averaged MSD is

$$\overline{\delta_i^2(\tau)} = \frac{1}{T-\tau}\sum_{t=1}^{T-\tau} \left\|\mathbf{x}_i(t+\tau) - \mathbf{x}_i(t)\right\|^2 . \tag{1.12}$$

The ensemble average of the TAMSD is then

$$\left\langle \overline{\delta^2(\tau)} \right\rangle = \frac{1}{N}\sum_{i=1}^{N} \overline{\delta_i^2(\tau)}. \tag{1.13}$$

For Brownian motion in d dimensions,

$$\mathrm{MSD}(\tau) = 2dD\tau. \tag{1.14}$$

For anomalous diffusion,

$$\mathrm{MSD}(\tau) \sim 2dK_\alpha \tau^\alpha, \tag{1.15}$$

where K_α is a generalized diffusion coefficient and α is the anomalous exponent. The protocol notes that MSD-based estimates are reliable for Brownian particles, but can become biased or insufficient for anomalous and heterogeneous trajectories, especially if different dynamical states are mixed before fitting [1, 23, 5].

For short trajectories, Kobayashi et al. discuss an alternative ensemble strategy based on the empirical distribution of displacements. If a Brownian particle in n spatial dimensions has diffusion coefficient D, then the cumulative distribution function of the squared displacement r^2 at lag τ can be written as

$$F(r^2;\tau) = 1 - \exp\left(-\frac{r^2}{2nD\tau}\right). \tag{1.16}$$

In two dimensions this reduces to the familiar denominator $4D\tau$. If two Brownian subpopulations are present, a simple mixture model is

$$F(r^2;\tau) = 1 - \left[a\exp\left(-\frac{r^2}{2nD_1\tau}\right) + (1-a)\exp\left(-\frac{r^2}{2nD_2\tau}\right)\right], \tag{1.17}$$

where a is the fraction of the first population. Fitting the CDF can be numerically more stable than fitting a noisy displacement histogram, and it is useful when individual trajectories are too short for reliable per-trajectory MSD fitting [2, 22, 5]. Its limitation is equally important: this model assumes Brownian populations. It does not estimate an anomalous exponent, a confinement length or a changepoint by itself, and it can hide rare subpopulations if the mixture model is chosen too coarsely [5].

Another simple summary statistic is the mean jump-distance of a trajectory,

$$m_i = \frac{1}{T_i - 1}\sum_{t=1}^{T_i-1} \left\|\mathbf{x}_i(t+1) - \mathbf{x}_i(t)\right\| . \tag{1.18}$$

Compared with pooling all displacements together, m_i preserves one value per trajectory and can separate slow and fast populations more clearly when trajectories are internally homogeneous. However, if a single trajectory contains a transition between states, the mean jump-distance averages over the transition. In that case segmentation should precede population analysis. Angle distributions between consecutive displacements,

$$\cos\theta_t = \frac{\Delta\mathbf{x}_t \cdot \Delta\mathbf{x}_{t+1}}{\|\Delta\mathbf{x}_t\|\,\|\Delta\mathbf{x}_{t+1}\|}, \tag{1.19}$$

can also reveal persistence, backtracking or directed motion, although in many live-cell SPT datasets the signal is weak unless motion is strongly directed [5].

1.6 Software and Practical State of the Art

Many public tools implement parts of the SPT pipeline. TrackMate provides an interactive ImageJ/Fiji environment for detection, linking and visual inspection; u-track and related likelihood-based methods perform robust reconnection in live-cell sequences; Trackpy implements Crocker–Grier style particle tracking in Python; Spot-On fits displacement distributions and corrects for defocalization bias; FreeTrace combines classical spot detection with tracking strategies adapted to particle density; DeepTrack and DeepSPT use neural networks for image or trajectory analysis; BI-ADD focuses on the estimation of anomalous-diffusion parameters and changepoints [18, 19, 21, 22, 28, 5].

There is no universally best SPT software. The correct choice depends on the data: spot density, background, localization precision, blinking, defocalization frequency, expected diffusivity, whether 2D or 3D tracking is needed, and whether a graphical interface or a scriptable pipeline is preferable. A method with many manual parameters can be flexible but may introduce user bias; a method with strong Brownian assumptions can be efficient but may fail for anomalous trajectories. This practical point is important for the present work: the quality of the final machine-learning estimates is limited by the quality of the trajectories and labels used to train, calibrate and evaluate the models. In other words, software selection is not a purely technical preference but a modeling decision. A displacement-distribution method such as Spot-On is appropriate when the goal is to estimate population diffusion fractions under a Brownian model with defocalization correction; a trajectory classifier or segmenter is preferable when individual trajectories may switch between dynamical states; and a neural method becomes attractive when the trajectories are too short or noisy for classical per-trajectory estimators [22, 28, 5].

1.7 Machine Learning for Trajectory Analysis

Machine learning enters SPT because it can learn directly from simulated or curated trajectories with known ground truth. In simulations one can generate trajectories with controlled values of D, α, diffusion model and changepoint positions. A supervised model can then learn a map of the form

$$(\Delta x_1, \Delta y_1, \ldots, \Delta x_{T-1}, \Delta y_{T-1}) \longmapsto (D, \alpha, t_c). \tag{1.20}$$

Unlike a hand-crafted MSD fit, a neural network can combine displacement magnitudes, local correlations, multi-lag information, changes in scale and sequence context. This is particularly useful for short trajectories, where large-lag TAMSD estimates have high variance because only $T - \tau$ terms are available.

The Anomalous Diffusion Challenge (AnDi) formalized this problem by proposing tasks for anomalous-exponent inference, diffusion-model classification and trajectory segmentation in 1D, 2D and 3D [24, 25]. More recent methods combine parameter estimation and changepoint detection at the single-trajectory level, for example DeepSPT and BI-ADD [26, 27, 28, 65]. The main limitation remains the simulation-to-real gap: a model trained on idealized synthetic trajectories may fail on experimental data if the simulations do not reproduce localization noise, motion blur, blinking, defocalization, tracking errors, biological heterogeneity and acquisition-specific artifacts. Therefore, experimental SPT knowledge and machine learning should not be separated; the experimental pipeline defines the statistical problem that the model must solve.

1.8 Motivation and Scope of This Thesis

The previous sections show that SPT analysis is limited by two connected problems: the trajectories are often short, and the quantities of biological interest are local rather than purely global. Classical MSD-based estimators are interpretable, but they are statistically unstable when only a few displacements are available. Population methods are robust, but they may

hide molecule-to- molecule heterogeneity and state changes inside individual trajectories. This motivates the central question of the thesis: whether a neural sequence model, combined with calibrated uncertainty, can extract useful diffusion profiles and changepoint information from trajectories as short as $T = 16$ frames.

Method	Main output	Strength	Limitation
MSD/TAMSD	Global D, α	Physically interpretable	Unstable for short or heterogeneous tracks
Spot-On	Population diffusion states	Robust ensemble analysis	Not designed for local trajectory segmentation
DeepSPT	Trajectory-level dynamical labels	Learns complex simulated patterns	Sensitive to the simulation-to-real gap
BI-ADD	Changepoints, K, α	Interpretable anomalous-diffusion pipeline	Segment estimates become difficult for very short tracks
This thesis	Frame-wise $\log_{10}(D_t)$, α_t, uncertainty	Focused on short trajectories and calibrated outputs	Still requires validation on experimental controls

Table 1.3: Position of this thesis with respect to common SPT analysis strategies.

Chapter 2

Mathematical Description of Brownian Motion and Anomalous Diffusion

2.1 Brownian Motion as a Limit of Random Walks

Brownian motion describes the irregular motion of a particle subject to many microscopic collisions. An intuitive construction starts from a discrete-time random walk:

$$X_n = \sum_{i=1}^{n} \xi_i, \tag{2.1}$$

where the increments ξ_i are independent and identically distributed with zero mean and finite variance. In the limit of small time steps and many steps, the process converges to a Wiener process $W(t)$. In d dimensions, Brownian motion with diffusion coefficient D can be written as

$$\mathbf{X}(t) = \mathbf{X}(0) + \sqrt{2D}\,\mathbf{W}(t). \tag{2.2}$$

Its fundamental properties are:

- independent increments;
- Gaussian increments;
- variance proportional to time;
- continuous but nowhere differentiable sample paths.

Formally, a standard Wiener process $W(t)$ satisfies

$$W(0) = 0, \qquad W(t) - W(s) \sim \mathcal{N}(0, t-s) \quad \text{for } 0 \le s < t, \tag{2.3}$$

with independent increments on disjoint intervals. Its covariance is

$$\mathbb{E}[W(t)W(s)] = \min(t, s). \tag{2.4}$$

In d dimensions, the components are independent Wiener processes. If $\mathbf{X}(t) = \sqrt{2D}\mathbf{W}(t)$, then

$$\mathrm{Cov}(X_i(t), X_j(s)) = 2D \min(t, s)\delta_{ij}. \tag{2.5}$$

This covariance already contains the structure of Brownian motion: the variance grows linearly with time and different spatial axes are uncorrelated.

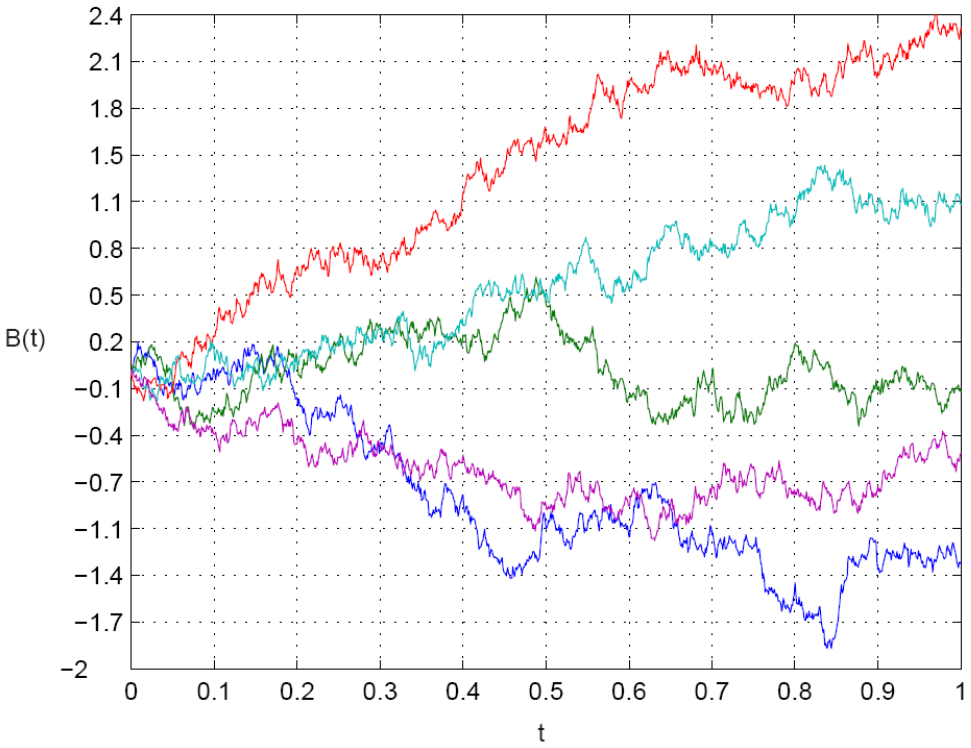


Figure 2.1: Examples of simulated one-dimensional Brownian trajectories $B(t)$. The different paths illustrate the stochastic variability of independent realizations of the same Brownian process.

2.2 Langevin Equation and Fluctuation-Dissipation

The random-walk construction explains Brownian motion as a scaling limit. A more physical derivation starts from the motion of a particle immersed in a thermal bath. At mesoscopic time scales, the many microscopic collisions with the molecules of the fluid are replaced by two effective terms: a friction force and a fluctuating force. In one spatial dimension, the inertial Langevin equation is

$$m\frac{dv(t)}{dt} = -\gamma v(t) + \xi(t), \qquad \frac{dx(t)}{dt} = v(t), \tag{2.6}$$

where m is the particle mass, γ is the friction coefficient and $\xi(t)$ is the Langevin force. For a spherical particle of radius R in a fluid of viscosity η, Stokes' law gives

$$\gamma = 6\pi\eta R. \tag{2.7}$$

The Langevin force is modeled as a zero-mean white noise,

$$\langle \xi(t) \rangle = 0, \qquad \langle \xi(t)\xi(t') \rangle = C\delta(t - t'). \tag{2.8}$$

The coefficient C cannot be chosen independently of γ. At thermal equilibrium, equipartition requires

$$\frac{1}{2}m\langle v^2 \rangle = \frac{1}{2}k_B T. \tag{2.9}$$

Consistency with this condition gives the fluctuation-dissipation relation

$$C = 2\gamma k_B T, \qquad \langle \xi(t)\xi(t') \rangle = 2\gamma k_B T\, \delta(t - t'). \tag{2.10}$$

This relation expresses the fact that friction and noise have the same microscopic origin: collisions with the surrounding thermal bath [31, 32].

The velocity relaxation time is

$$\tau_v = \frac{m}{\gamma}. \tag{2.11}$$

Solving the linear Langevin equation gives

$$\langle v(t) \rangle = v(0) e^{-t/\tau_v}. \tag{2.12}$$

In the stationary regime, the velocity autocorrelation is

$$\langle v(t) v(0) \rangle = \frac{k_B T}{m} e^{-|t|/\tau_v}. \tag{2.13}$$

The diffusion coefficient is the integral of this velocity autocorrelation, also called a Green–Kubo relation:

$$D = \int_0^\infty \langle v(t) v(0) \rangle \, dt = \frac{k_B T}{\gamma}. \tag{2.14}$$

Combining this with Stokes' law gives the Stokes–Einstein relation

$$D = \frac{k_B T}{6\pi\eta R}. \tag{2.15}$$

This formula is idealized, but it gives useful intuition for protein dynamics: larger hydrodynamic radius or higher effective viscosity lowers D.

The same model also shows the crossover between ballistic and diffusive motion. In one dimension,

$$\left\langle [x(t) - x(0)]^2 \right\rangle = 2D \left[t - \tau_v \left(1 - e^{-t/\tau_v} \right) \right]. \tag{2.16}$$

For $t \ll \tau_v$, the MSD grows approximately as

$$\left\langle [x(t) - x(0)]^2 \right\rangle \simeq \frac{k_B T}{m} t^2, \tag{2.17}$$

which is ballistic motion. For $t \gg \tau_v$, the exponential term is negligible and one recovers ordinary diffusion,

$$\left\langle [x(t) - x(0)]^2 \right\rangle \simeq 2Dt. \tag{2.18}$$

Most SPT experiments sample the overdamped regime, where the inertial relaxation time is much shorter than the camera exposure time. In this limit, velocity is not explicitly resolved and the position obeys

$$dX(t) = \sqrt{2D}\, dW(t) \tag{2.19}$$

in the absence of external forces. With a force $F(x)$, the overdamped equation becomes

$$dX(t) = \frac{F(X(t))}{\gamma}\, dt + \sqrt{2D}\, dW(t). \tag{2.20}$$

Thus Brownian trajectories can be seen either as limits of random walks or as solutions of stochastic differential equations driven by white noise [31, 32].

2.3 Diffusion Equation and MSD

Before moving fully to trajectory-based descriptions, it is useful to recall the classical continuum description of normal diffusion. Fick's law states that the particle flux is proportional to the negative concentration gradient,

$$\mathbf{J}(\mathbf{x}, t) = -D \nabla c(\mathbf{x}, t), \tag{2.21}$$

where $c(\mathbf{x}, t)$ is the concentration field and D is the diffusion coefficient. Together with mass conservation, $\partial_t c + \nabla \cdot \mathbf{J} = 0$, this gives the diffusion equation

$$\partial_t c(\mathbf{x}, t) = D\nabla^2 c(\mathbf{x}, t). \tag{2.22}$$

This macroscopic Fickian description is consistent with normal Brownian diffusion, while SPT observes diffusion through individual stochastic trajectories. This motivates the trajectory-based MSD and anomalous-diffusion descriptions introduced below.

The probability density $p(\mathbf{x}, t)$ of finding the particle at $\mathbf{x}$ at time t satisfies the diffusion equation

$$\frac{\partial p}{\partial t} = D\nabla^2 p. \tag{2.23}$$

Starting from a point-like initial condition, the solution is Gaussian:

$$p(\mathbf{x}, t) = \frac{1}{(4\pi D t)^{d/2}} \exp\left(-\frac{|\mathbf{x}|^2}{4Dt}\right). \tag{2.24}$$

The mean squared displacement is

$$\left\langle |\mathbf{X}(t) - \mathbf{X}(0)|^2 \right\rangle = 2dDt. \tag{2.25}$$

In two dimensions this becomes $\mathrm{MSD}(t) = 4Dt$.

For discrete data, with frames separated by Δt, Brownian increments are distributed as

$$\Delta \mathbf{x}_t \sim \mathcal{N}\left(0, 2D\Delta t\, I_d\right). \tag{2.26}$$

This relation explains why increments contain direct information about D.

Brownian motion can also be characterized through its infinitesimal generator. For a sufficiently regular function f,

$$\mathcal{L}f = D\nabla^2 f. \tag{2.27}$$

The diffusion equation is therefore the Kolmogorov forward equation associated with the process. This equivalence between probabilistic and partial differential equation descriptions connects SPT, statistical physics and parameter inference [29, 30, 31].

More generally, if the overdamped dynamics contains a drift field $\mathbf{b}(\mathbf{x})$, the stochastic differential equation

$$d\mathbf{X}_t = \mathbf{b}(\mathbf{X}_t)\, dt + \sqrt{2D}\, d\mathbf{W}_t \tag{2.28}$$

leads to the Fokker–Planck equation

$$\frac{\partial p}{\partial t} = -\nabla \cdot (\mathbf{b}p) + D\nabla^2 p. \tag{2.29}$$

This can be written as a conservation law,

$$\frac{\partial p}{\partial t} = -\nabla \cdot \mathbf{J}, \qquad \mathbf{J} = \mathbf{b}p - D\nabla p. \tag{2.30}$$

The first contribution is a drift current and the second is a diffusion current from regions of high density to regions of low density. For a constant drift $\mathbf{b} = 0$, one recovers pure diffusion. For a physical force $\mathbf{F}$, the drift is $\mathbf{b} = \mathbf{F}/\gamma$ in the overdamped limit [31, 32].

2.4 From the Master Equation to the Fokker–Planck Limit

The Fokker–Planck equation can also be understood as the continuum limit of a more general Markov jump process. This point is useful because experimental trajectories are always sampled at discrete frames: the observed motion is a sequence of increments, while Brownian motion is a continuous idealization. Following the standard notation for a one-dimensional process, let $P_t(x)$ be the probability density and let $W(x|x')$ be the transition rate from position x' to position x. The master equation is

$$\frac{\partial P_t(x)}{\partial t} = \int W(x|x')P_t(x')\,dx', \qquad \int W(x|x')\,dx = 0. \tag{2.31}$$

The second condition expresses conservation of total probability. It means that the gain of probability at some positions must be balanced by loss somewhere else.

It is convenient to write the jump amplitude as

$$\eta = x - x'. \tag{2.32}$$

If $\widetilde{w}(x;\eta)$ is the rate density for making a jump of amplitude η from x, the master equation can be written schematically as

$$\frac{\partial P_t(x)}{\partial t} = \int \widetilde{w}(x-\eta;\eta)P_t(x-\eta)\,d\eta - P_t(x)\int \widetilde{w}(x;\eta)\,d\eta. \tag{2.33}$$

This equation is more general than a diffusion equation: it allows finite jumps, non-Gaussian increments and processes that are not continuous.

The Fokker–Planck equation emerges when jumps are small and frequent. If $\widetilde{w}(x;\eta)$ is sharply concentrated near $\eta = 0$, one can expand the gain term in powers of η. Defining the jump moments

$$a_n(x) = \int \eta^n \widetilde{w}(x;\eta)\,d\eta, \tag{2.34}$$

one obtains the Kramers–Moyal expansion

$$\frac{\partial P_t(x)}{\partial t} = \sum_{n=1}^{\infty} \frac{(-1)^n}{n!}\frac{\partial^n}{\partial x^n}\left[a_n(x)P_t(x)\right]. \tag{2.35}$$

Keeping only the first two terms gives

$$\frac{\partial P_t(x)}{\partial t} = -\frac{\partial}{\partial x}\left[a_1(x)P_t(x)\right] + \frac{1}{2}\frac{\partial^2}{\partial x^2}\left[a_2(x)P_t(x)\right]. \tag{2.36}$$

This is the Fokker–Planck equation with drift

$$b(x) = a_1(x) \tag{2.37}$$

and position-dependent diffusion coefficient

$$D(x) = \frac{a_2(x)}{2}. \tag{2.38}$$

Thus drift is the first local moment of the jumps per unit time, and diffusion is half of the second local moment per unit time. This gives a precise meaning to the idea that diffusion is the large-scale limit of many microscopic random kicks [31, 32].

This derivation also clarifies the assumptions behind Brownian modeling. If jump distributions have finite second moment and the observed dynamics is coarse grained over many small kicks, the second-order Fokker–Planck approximation is natural. If the process has heavy-tailed

jumps, long waiting times, or memory, higher-order or nonlocal descriptions may be needed. This is one mathematical reason why anomalous diffusion cannot be identified from the MSD exponent alone: different microscopic mechanisms can share the same scaling law but correspond to different stochastic equations.

For a lattice random walk, the same continuum limit can be seen explicitly. In d dimensions, suppose a walker jumps every Δt to one of its nearest neighbors on a lattice of spacing a. The continuum Brownian limit is obtained by sending

$$a \to 0, \qquad \Delta t \to 0, \qquad \frac{a^2}{2d\Delta t} \to D. \tag{2.39}$$

In this limit, the propagator becomes Gaussian and satisfies the diffusion equation. At the same time, the limiting path is continuous but nowhere differentiable. Indeed,

$$\mathbb{E}\left[(X(t+\Delta t) - X(t))^2\right] = 2D\Delta t, \tag{2.40}$$

so the squared finite-difference velocity scales as

$$\mathbb{E}\left[\left(\frac{X(t+\Delta t) - X(t)}{\Delta t}\right)^2\right] = \frac{2D}{\Delta t} \to \infty \qquad \text{as } \Delta t \to 0. \tag{2.41}$$

This is why the Langevin equation is written in stochastic differential form: the white-noise force is not an ordinary function of time, and Brownian paths do not have ordinary velocities [32].

Boundary conditions also have a direct probabilistic meaning. For a reflecting boundary, the normal probability current vanishes,

$$\mathbf{J} \cdot \mathbf{n} = 0, \tag{2.42}$$

so probability cannot leave the domain. For an absorbing boundary, the density is set to zero at the boundary,

$$p = 0, \tag{2.43}$$

which represents removal of trajectories that hit that boundary. These conditions are important conceptually for cellular SPT, where apparent confinement, membrane domains and escape from compartments can change the statistics of observed trajectories.

2.5 Anomalous Diffusion

Many biological systems do not show a linear MSD. In such cases one speaks of anomalous diffusion, described by

$$\mathrm{MSD}(\tau) \sim 2dD\tau^{\alpha}. \tag{2.44}$$

If $\alpha = 1$, the motion is Brownian. If $0 < \alpha < 1$, the motion is subdiffusive; if $\alpha > 1$, it is superdiffusive. Subdiffusion can arise from crowding, confinement, obstacles, viscoelasticity or transient binding. Superdiffusion can instead be associated with active transport or persistent motion.

Anomalous diffusion does not identify a unique mechanism. Different processes can produce the same value of α while having different statistical properties. Therefore, in addition to estimating α, one must consider the generative model and possible changes along the trajectory [33, 34, 35, 43].

In logarithmic scale, the power law becomes

$$\log \mathrm{MSD}(\tau) = \log(2dD) + \alpha \log \tau. \tag{2.45}$$

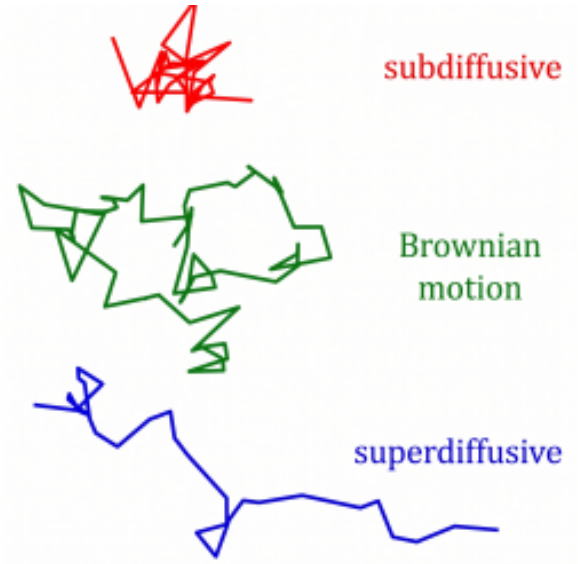


Figure 2.2: Representative trajectories for three transport regimes. The red trajectory illustrates subdiffusive motion ($0 < \alpha < 1$), the green trajectory illustrates Brownian motion ($\alpha = 1$), and the blue trajectory illustrates superdiffusive motion ($\alpha > 1$). The figure is meant as a qualitative visualization of how the anomalous exponent changes the geometry of the path: subdiffusive trajectories remain more spatially confined, Brownian trajectories spread diffusively, and superdiffusive trajectories show more persistent exploration.

Thus α is the slope of the curve in a log-log plot. However, in anomalous diffusion the parameter D does not always have the same physical dimension as the Brownian diffusion coefficient. Since

$$[\text{MSD}] = L^2, \qquad [\tau] = T, \tag{2.46}$$

one has

$$[D] = L^2 T^{-\alpha}. \tag{2.47}$$

For this reason, D is often called a generalized diffusion coefficient. More precisely, when $\alpha \neq 1$, the prefactor is often denoted K_α to emphasize that its physical dimension depends on α. In the numerical part of this thesis, D or $\log_{10}(D)$ denotes the diffusion-scale parameter predicted by the neural model. This notation is used because the central task is to detect changes in mobility scale, while the anomalous exponent is treated separately.

2.6 Fractional Brownian Motion

An important model is fractional Brownian motion (fBm), denoted by $B_H(t)$, where $H \in (0, 1)$ is the Hurst exponent. The process is Gaussian, has stationary but non-independent increments, and satisfies

$$\mathbb{E}\left[(B_H(t+\tau) - B_H(t))^2\right] \propto \tau^{2H}. \tag{2.48}$$

The anomalous exponent is

$$\alpha = 2H. \tag{2.49}$$

For $H = 1/2$, one recovers standard Brownian motion. If $H < 1/2$, increments are anticorrelated and the motion is subdiffusive. If $H > 1/2$, increments are positively correlated and the motion is superdiffusive.

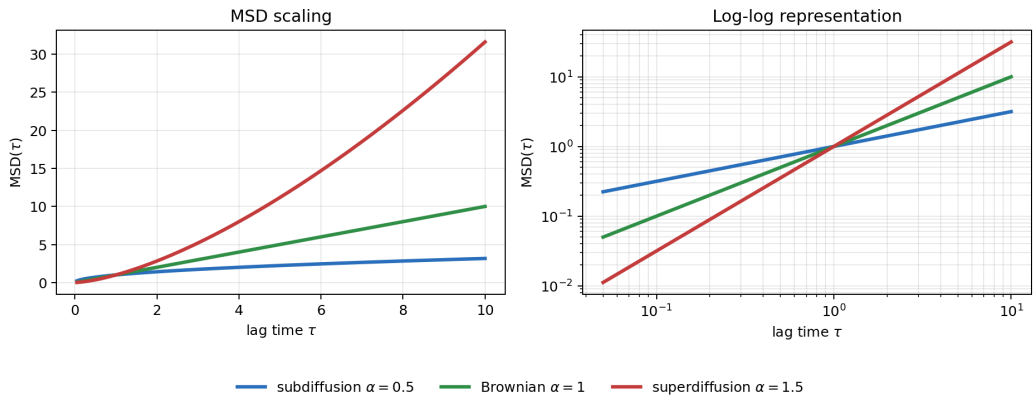


Figure 2.3: Comparison between subdiffusive, Brownian and superdiffusive MSD curves in linear and log-log scale.

The full covariance of fractional Brownian motion is

$$\mathbb{E}[B_H(t)B_H(s)] = \frac{\sigma^2}{2}\left(t^{2H} + s^{2H} - |t-s|^{2H}\right). \tag{2.50}$$

The increments

$$G_H(t) = B_H(t+1) - B_H(t) \tag{2.51}$$

are called fractional Gaussian noise. Their autocovariance is

$$\gamma(k) = \frac{\sigma^2}{2}\left(|k+1|^{2H} - 2|k|^{2H} + |k-1|^{2H}\right). \tag{2.52}$$

For $H < 1/2$, $\gamma(k) < 0$ at small lags: a displacement in one direction tends to be followed by a displacement in the opposite direction. For $H > 1/2$, $\gamma(k) > 0$: the motion is persistent. This correlation structure is one reason why sequence models are useful: they can learn temporal patterns beyond displacement amplitude alone [45, 46].

2.7 Why Fractional Brownian Motion Appears in Cells

Fractional Brownian motion is not just a convenient mathematical model. It is particularly relevant for intracellular motion because the cytoplasm and the nucleoplasm are not dilute Newtonian liquids. They are crowded, structured and viscoelastic media containing proteins, nucleic acids, organelles, membranes and cytoskeletal filaments. A particle moving in such an environment does not experience independent kicks as in ideal Brownian motion. Instead, the medium has memory: a displacement deforms or rearranges the surrounding material, and the relaxation of that material affects subsequent displacements [36, 39, 40].

This physical picture explains why subdiffusive fBm often corresponds to anticorrelated increments. If a protein or tracer particle moves in one direction, the elastic or viscoelastic response of the surrounding medium can partially pull it back. Mathematically, for $H < 1/2$,

$$\mathrm{Cov}\left(B_H(t+\Delta t) - B_H(t), B_H(t+2\Delta t) - B_H(t+\Delta t)\right) < 0. \tag{2.53}$$

Thus a positive increment tends to be followed by a negative increment. This is different from a CTRW interpretation, where subdiffusion is often caused by long waiting times or trapping events. Both models can yield $\mathrm{MSD}(\tau) \propto \tau^\alpha$, but they imply different microscopic mechanisms: fBm suggests correlated motion in a viscoelastic environment, whereas CTRW suggests intermittent immobilization or broadly distributed residence times [33, 35, 40].

The connection between fBm and cellular material properties can also be stated in rheological language. If a passive tracer is embedded in a viscoelastic medium, its MSD can be related to the mechanical response of the medium. In experiments on living cells, anomalous scaling of tracer motion has been used to infer viscoelastic behavior of the cytoplasm and nucleoplasm [36]. A smaller α indicates stronger subdiffusion and is often associated with a more elastic, crowded or constrained environment, whereas α closer to one indicates motion closer to ordinary diffusion. However, the same α can arise from different mechanisms; therefore model selection and experimental context are essential.

2.8 Interpreting D and α for Protein Dynamics

For proteins, D and α should be interpreted as effective dynamical observables. The diffusion coefficient D primarily reflects the scale of motion over the observation window. In a dilute fluid, the Stokes–Einstein relation suggests

$$D = \frac{k_B T}{6\pi\eta R_h}, \tag{2.54}$$

where k_B is Boltzmann's constant, T is temperature, η is the viscosity and R_h is the hydrodynamic radius. This relation gives useful intuition: larger proteins or complexes, or proteins moving in a more viscous environment, tend to have smaller D. Inside cells, however, this simple formula is only an approximation because proteins interact with many partners, encounter obstacles and may switch between dynamical states [4, 3, 40].

A high apparent D is typically consistent with a freely diffusing or weakly interacting protein, especially if $\alpha \approx 1$. A low D may indicate a larger molecular complex, increased local viscosity, transient binding, confinement, chromatin association, membrane association or localization inside a dense biomolecular environment. If D becomes close to zero over the measurement time, the molecule may appear immobile, but this should be interpreted carefully: true immobilization, long residence times, tracking limitations and localization noise can all produce very small apparent displacements.

The anomalous exponent α gives complementary information. Values close to one suggest Brownian-like diffusion on the measured time and length scales. Values below one suggest subdiffusion, which can be caused by molecular crowding, viscoelasticity, confinement, nonspecific interactions or transient binding. Values above one suggest persistent or directed components, such as active transport, motor-driven motion, flow, or a persistent search phase. For soluble proteins, strong superdiffusion is less commonly interpreted as passive thermal motion alone and usually requires considering active or directed mechanisms [35, 40, 4].

This distinction motivates the emphasis placed later on D. In SPT of proteins, changes in the diffusion coefficient are often the most actionable signal: they can indicate that a molecule has entered a slower diffusive state, joined a larger complex, become transiently bound, or moved into a cellular region with different effective viscosity or obstruction. This is why several protein-dynamics analyses explicitly model populations through diffusion coefficients and state fractions [22, 41]. The anomalous exponent is essential for describing the type of motion, but it is not by itself a unique biochemical label. Two trajectories with similar α may come from different microscopic mechanisms, while a strong change in D directly changes the spatial scale explored by the protein over the experimental window.

It is therefore better to treat D and α as quantitative signatures of mobility rather than direct labels of biological state, with D carrying the primary information about local mobility scale. For example, a decrease in D can support a binding or confinement hypothesis, but it does not prove binding by itself. Stronger interpretation requires additional evidence: colocalization with a cellular structure, residence-time analysis, perturbation experiments, comparison with controls, and uncertainty estimates for the fitted or predicted parameters.

Observed behavior	Possible protein-level interpretation	Caution
Large D, $\alpha \simeq 1$	Freely diffusing, weakly interacting protein	Depends on viscosity and protein size
Small D, $\alpha \simeq 1$	Large complex or high local viscosity	May also reflect short tracks or noise
Small D, $\alpha < 1$	Crowding, confinement, transient binding, chromatin association	Mechanisms are not uniquely identifiable
Very small D	Long residence time or apparent immobilization	May be limited by localization precision
$\alpha > 1$	Persistent or active motion, directed transport, flow	Not typical of purely passive diffusion

Table 2.1: Qualitative interpretation of D and α for protein dynamics. These interpretations are hypotheses that must be checked against biological context, controls and uncertainty estimates [3, 4, 40].

2.9 Other Models of Anomalous Diffusion

Other common anomalous-diffusion models include:

- continuous-time random walks (CTRW), where heavy-tailed waiting times can produce subdiffusion and non-ergodicity;
- scaled Brownian motion (SBM), where the diffusion coefficient depends on time;
- Levy walks, used for some superdiffusive processes;
- multi-state models, where D and α change along the trajectory.

Model	Main mechanism	Increment structure	Typical biological interpretation
fBm	Viscoelastic memory	Correlated increments; anticorrelated for $\alpha < 1$	Crowded or elastic intracellular medium
CTRW	Broad waiting times or trapping	Jumps separated by random immobilization periods	Binding, trapping or intermittent immobilization
SBM	Time-dependent diffusivity	Non-stationary variance scale	Slowly changing effective environment
Multi-state model	Switching between parameter regimes	Piecewise dynamics	Transitions between mobility states

Table 2.2: Qualitative comparison of common anomalous-diffusion models. Similar MSD scaling can arise from different mechanisms, so α alone should not be interpreted as a unique biological label.

In a CTRW, motion is defined by spatial jumps separated by random waiting times. If the waiting-time distribution has a heavy tail,

$$\psi(t) \sim t^{-1-\alpha}, \qquad 0 < \alpha < 1, \tag{2.55}$$

the average number of jumps grows sublinearly in time and the process becomes subdiffusive. Unlike fBm, CTRW can show aging and weak ergodicity breaking: time averages and ensemble averages do not coincide even at long times [33, 35].

In scaled Brownian motion, the process satisfies a diffusion equation with time-dependent diffusivity,

$$\frac{\partial p}{\partial t} = D(t)\nabla^2 p, \qquad D(t) = D_0 \alpha t^{\alpha-1}. \tag{2.56}$$

Here too the MSD scales as t^α, but the mechanism is different: increments may remain Gaussian while the diffusion scale changes in time.

The presence of multiple states naturally leads to segmentation. If a trajectory changes regime at time t_c, one can write

$$(D, \alpha) = \begin{cases} (D_1, \alpha_1), & t < t_c, \\ (D_2, \alpha_2), & t \geq t_c. \end{cases} \tag{2.57}$$

Change point detection aims to estimate t_c from the observed data.

Chapter 3

Convolutional Neural Networks and Transformers

3.1 Supervised Neural Networks

A neural-network model is useful in this project because the relevant map is an inverse statistical problem: from a short noisy sequence of displacements one wants to infer the parameters of an underlying stochastic process. In the language of statistical learning, the dataset is a collection

$$\mathcal{D} = \{(X_i, y_i)\}_{i=1}^{N}, \tag{3.1}$$

where X_i is the input trajectory, usually represented by increments, and y_i is a label such as D, α, a pair (D, α), or a time-dependent profile. A supervised neural network approximates a function

$$f_\theta : X \mapsto y, \tag{3.2}$$

where θ denotes the trainable parameters. For trajectories, X can be the sequence of increments and y can be D, α, a sequence of local parameter values, or the position of a change point. Training consists of minimizing a loss function, for example

$$\widehat{\mathcal{R}}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell\left(y_i, f_\theta(X_i)\right), \tag{3.3}$$

where $\widehat{\mathcal{R}}$ is the empirical risk. For regression, common choices are the L_1 loss

$$\ell_1(y, \hat{y}) = |y - \hat{y}| \tag{3.4}$$

and the squared loss

$$\ell_2(y, \hat{y}) = (y - \hat{y})^2. \tag{3.5}$$

The central difficulty is that minimizing the training loss is not the same as making accurate predictions on new trajectories. Mehta et al. emphasize this distinction through the bias–variance tradeoff: a very flexible model can fit training noise, whereas a model that is too simple may miss the structure of the data [49].

A basic feed-forward network is composed of affine maps and nonlinearities:

$$\mathbf{h}^{(0)} = \mathbf{x}, \qquad \mathbf{h}^{(\ell)} = \phi\left(W^{(\ell)}\mathbf{h}^{(\ell-1)} + \mathbf{b}^{(\ell)}\right), \tag{3.6}$$

where ϕ may be a ReLU,

$$\phi(z) = \max(0, z). \tag{3.7}$$

Training is performed by backpropagation. If $\mathcal{L}$ is the loss, the parameters can be updated by gradient descent:

$$\theta_{m+1} = \theta_m - \eta \nabla_\theta \mathcal{L}(\theta_m), \tag{3.8}$$

where η is the learning rate. In practice, stochastic or mini-batch versions are almost always used.

The dataset is therefore split into training, validation and test sets. The training set updates θ; the validation set is used to choose hyperparameters such as learning rate, number of filters, dropout rate and training time; the test set estimates final predictive performance. This split is essential for SPT because simulated trajectories can be abundant but experimental trajectories are noisy and heterogeneous. A small training error with a much larger validation error indicates overfitting. A large training error indicates underfitting or an optimization problem. Regularization, dropout, early stopping and data augmentation are practical ways to control this tradeoff [49, 48, 51].

For trajectory regression, the loss also has a statistical interpretation. A quadratic loss penalizes outliers strongly and corresponds to a Gaussian-error assumption. An L_1 loss is more robust and corresponds to a Laplace-error assumption. In the implementation used in this work, the CNN–Transformer model is trained with an L_1 loss, which is reasonable when errors may have non-negligible tails.

3.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) exploit two assumptions that are also familiar in physics: locality and translation equivariance. Locality means that nearby elements of the input contain important information. Translation equivariance means that the same pattern should be recognized independently of where it appears. Mehta et al. describe CNNs as neural networks that build these structures directly into the architecture through local receptive fields and shared weights [49]. For SPT, this is natural: a local burst of large displacements, an anticorrelation pattern, or a change in displacement scale should be detectable wherever it occurs along the trajectory.

For a one-dimensional sequence, a convolution computes

$$z_t^{(k)} = \sum_{c=1}^{C} \sum_{j=-m}^{m} w_{c,j}^{(k)} x_{c,t+j} + b^{(k)}, \tag{3.9}$$

where c indexes the input channel, k indexes the filter and $2m+1$ is the kernel size. For SPT trajectories, the natural channels are dx and dy.

More generally, with kernel size F, stride S, padding P and input length T, the output length is

$$T_{\text{out}} = \left\lfloor \frac{T - F + 2P}{S} \right\rfloor + 1. \tag{3.10}$$

The same filter weights are used at every temporal position. This parameter sharing strongly reduces the number of trainable parameters compared with a fully connected layer, and it makes the model easier to train on finite datasets. In the SPT setting, a filter may learn something analogous to a local finite-difference operator, a local energy estimate $\|\Delta \mathbf{x}_t\|^2$, or a short-range correlation detector [47, 49].

Figure 3.1 shows the CNN baseline used for trajectory regression and makes explicit how local convolutional features are converted into a scalar diffusion-parameter estimate.

The CNN used for the homogeneous-trajectory baseline contains three convolutional layers, pooling, dropout and fully connected layers. This architecture is suitable for global estimates of D or α, because it compresses the sequence into a final representation from which a scalar value is predicted.

```python
class CNN(nn.Module):
    def __init__(self, seq_len):
        super().__init__()

        self.conv1 = nn.Conv1d(2, 64, kernel_size=5, padding=2)
        self.conv2 = nn.Conv1d(64, 128, kernel_size=3, padding=1)
        self.conv3 = nn.Conv1d(128, 256, kernel_size=3, padding=1)

        self.pool = nn.MaxPool1d(kernel_size=2)
        self.dropout = nn.Dropout(0.15)

        final_seq_len = seq_len // 8   # tre pooling
        self.fc1 = nn.Linear(256 * final_seq_len, 128)
        self.fc2 = nn.Linear(128, 64)
        self.out = nn.Linear(64, 1)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = self.pool(x)

        x = F.relu(self.conv2(x))
        x = self.pool(x)

        x = F.relu(self.conv3(x))
        x = self.pool(x)

        x = x.view(x.size(0), -1)
        x = F.relu(self.fc1(x))
        x = self.dropout(x)
        x = F.relu(self.fc2(x))
        x = self.out(x)
        return x
```

Figure 3.1: Example implementation of a one-dimensional CNN for trajectory regression. The network receives a two-channel temporal input, applies convolution, pooling and dropout layers, and ends with fully connected regression layers.

Pooling performs a coarse graining of the feature map. For max pooling,

$$z'_t = \max_{j \in \mathcal{W}_t} z_j \tag{3.11}$$

where $\mathcal{W}_t$ is a local temporal window. This is useful for global parameter estimation because the exact position of a local fluctuation may not matter. It is less innocent for change point detection: if pooling is too aggressive, temporal resolution is lost and the predicted transition can become blurred. Thus, a CNN used for frame-wise SPT prediction must preserve enough temporal structure to localize changes.

Dropout introduces noise during training:

$$\tilde{h}_i = m_i h_i, \qquad m_i \sim \text{Bernoulli}(p), \tag{3.12}$$

reducing co-adaptation of neurons and improving generalization [51].

3.3 Limits of CNNs on Sequences

A CNN initially sees only local windows. To capture long-range dependencies, one needs many layers, large kernels or dilations. This can be a limitation when the quantity to infer depends on correlations across multiple temporal scales, as in fBm or in processes with change points. Moreover, if frame-wise prediction is needed, temporal information must be preserved throughout the network.

3.4 Embeddings

Before introducing Transformers, it is useful to define what an embedding is. In machine learning, an embedding is a vector representation of an object. The object may be a word, an image patch, a molecule, a graph node, or a time point in a trajectory. The purpose of the embedding is to map the raw object into a continuous vector space where the neural network can compare, combine and transform it:

$$e : \mathcal{X} \to \mathbb{R}^{d_{\text{model}}}. \tag{3.13}$$

The dimension d_{model} is not necessarily the physical dimension of the data. It is the dimension of the internal representation used by the model.

The elementary object given to a sequence model is usually called a *token*. A token is not necessarily a word. It is the unit into which the input sequence is decomposed before being processed by the network. In natural language, a token can be a word, a subword fragment, a punctuation mark or a special symbol. In vision Transformers, a token can be an image patch. In SPT, a natural token is one time step of the trajectory, for example the displacement $\Delta\mathbf{x}_t = (dx_t, dy_t)$, or a small local window of consecutive displacements. Thus, a trajectory can be viewed abstractly as a token sequence

$$X = (x_1, x_2, \dots, x_T), \tag{3.14}$$

where each x_t is the information observed at time t. The word *token* therefore means "one element of the sequence", while the word *embedding* means the vector representation of that element used inside the model.

There are two important cases. In language models, tokens are discrete objects chosen from a finite vocabulary $\mathcal{V}$. A tokenizer first maps the raw text into a sequence of token identifiers,

$$\text{text} \longmapsto (w_1, w_2, \dots, w_T), \qquad w_t \in \mathcal{V}. \tag{3.15}$$

If token w_t has vocabulary index j, it can be represented as a one-hot vector $\mathbf{u}_j$. The embedding layer is then a learned matrix

$$E \in \mathbb{R}^{|\mathcal{V}| \times d_{\text{model}}}, \tag{3.16}$$

and the embedded token is

$$\mathbf{e}_t = E^\top \mathbf{u}_j. \tag{3.17}$$

Equivalently, $\mathbf{e}_t$ is the j-th row of the embedding matrix. The network learns this vector during training. Tokens that play similar roles in similar contexts can end up with similar embeddings, because the training objective pushes their internal representations in similar directions.

For continuous scientific data, such as SPT trajectories, there is usually no vocabulary and no one-hot vector. The token is already numerical. The embedding layer is then a learned transformation from the physical coordinates of the token to the internal feature space of the network. This distinction is useful: the sequence length T counts how many tokens are present, whereas d_{model} counts how many learned features are used to represent each token.

A classical example is word2vec. In the skip-gram formulation, the model sees a center word w and tries to distinguish true context words w' from randomly sampled negative examples. One learns a word embedding

$$e(w) \in \mathbb{R}^p \tag{3.18}$$

and a context embedding

$$c(w') \in \mathbb{R}^p. \tag{3.19}$$

The probability that the pair (w, w') is a true word–context pair can be modeled as

$$p(y = 1 \mid w, w') = \sigma\left(e(w) \cdot c(w')\right), \tag{3.20}$$

where σ is the logistic sigmoid. Training with cross-entropy pushes embeddings of words that appear in similar contexts to have compatible vector representations. This is why semantic relations can appear geometrically in the embedding space: the vectors are not manually assigned meanings, but the training objective organizes them according to contextual similarity [50].

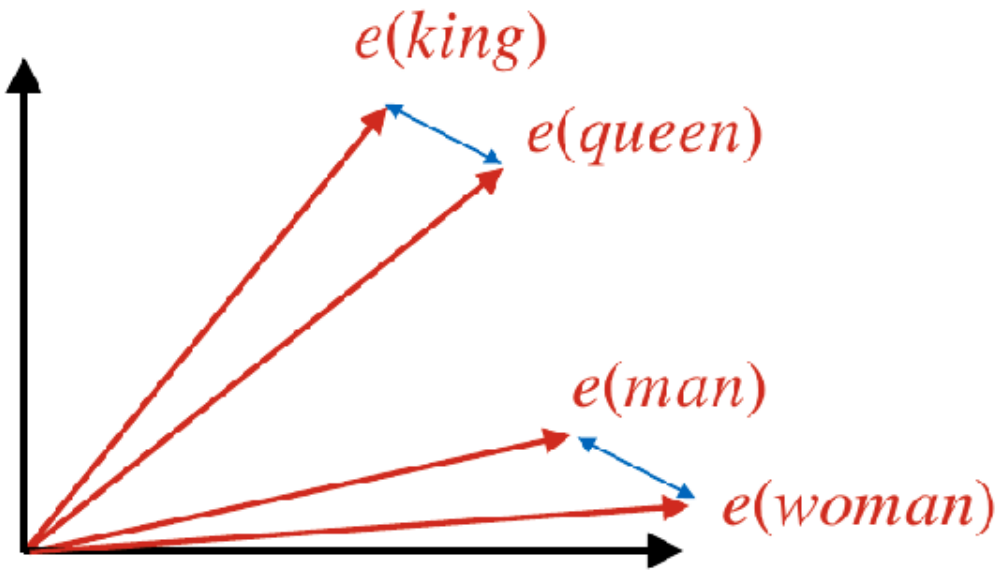


Figure 3.2: Word2vec-style embedding geometry. Words are represented as vectors in a learned continuous space; relations between words can correspond to directions or relative positions in that space.

Figure 3.2 gives a visual example of the general idea of an embedding: objects are mapped to vectors in a continuous space where geometric relations can encode useful learned similarities.

For SPT trajectories the input is already numerical, not a discrete vocabulary. At time t, the raw displacement can be written as

$$\Delta\mathbf{x}_t = (dx_t, dy_t) \in \mathbb{R}^2. \tag{3.21}$$

An embedding layer maps this two-dimensional physical input into a higher dimensional latent vector:

$$\mathbf{e}_t = W_{\text{emb}}\Delta\mathbf{x}_t + \mathbf{b}_{\text{emb}}, \qquad \mathbf{e}_t \in \mathbb{R}^{d_{\text{model}}}. \tag{3.22}$$

This operation does not create new measurements; it creates a representation in which the network can store several learned features of the same displacement. Different coordinates of $\mathbf{e}_t$ may encode displacement magnitude, direction, local scale, or combinations that are useful for predicting D, α, or a change point.

An embedding is therefore not the final prediction. It is an intermediate coordinate system learned for the task. In a Transformer, the sequence of embeddings

$$E_X = (\mathbf{e}_1, \ldots, \mathbf{e}_T) \in \mathbb{R}^{T \times d_{\text{model}}} \tag{3.23}$$

is the object on which attention operates. Positional encodings are then added to these vectors so that the model also knows where each element occurs in the trajectory [54].

3.5 Transformers and Self-Attention

The Transformer, introduced by Vaswani et al. [54], was designed to process sequences without recurrent layers. The original architecture is an encoder–decoder model for machine translation, but the component most relevant for trajectory analysis is the encoder block: it maps an input sequence into a new sequence of contextual representations. For SPT, this means that each displacement can be represented after comparing it with all the other displacements in the trajectory. This is useful when the meaning of a local step depends on nonlocal context, for example when one compares the dynamics before and after a possible change point.

The basic operation is scaled dot-product attention. Given the embedded sequence

$$X \in \mathbb{R}^{T \times d_{\text{model}}}, \tag{3.24}$$

three learned linear maps produce queries, keys and values:

$$Q = XW_Q, \qquad K = XW_K, \qquad V = XW_V, \tag{3.25}$$

with

$$Q, K \in \mathbb{R}^{T \times d_k}, \qquad V \in \mathbb{R}^{T \times d_v}. \tag{3.26}$$

Although they are all obtained from the same input sequence, queries, keys and values have different roles. For each position i, the query vector q_i represents what that position is looking for in the rest of the sequence. For each position j, the key vector k_j represents the information by which that position can be matched or selected. The value vector v_j contains the content that will actually be passed to other positions if position j is considered relevant. In this sense, attention separates the act of *matching* from the act of *reading*: queries and keys decide the weights, while values provide the vectors that are averaged.

This distinction is useful in SPT. Suppose position i corresponds to a small displacement inside a possibly confined segment. Its query can learn to ask whether other time points show similar local dynamics. Positions j with compatible keys receive large attention weights, and their values contribute to the new representation of position i. If the trajectory contains a changepoint, positions before and after the transition may produce different key–query similarities, allowing the model to compare regimes without scanning the sequence recurrently. The vectors q_i, k_j and v_j are not fixed hand-designed physical quantities; they are learned projections optimized for the prediction task [54].

The score between time points i and j is the dot product

$$e_{ij} = \frac{q_i^\top k_j}{\sqrt{d_k}}. \tag{3.27}$$

After applying a row-wise softmax, one obtains attention weights

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{\ell=1}^{T} \exp(e_{i\ell})}. \tag{3.28}$$

The output representation at position i is the weighted average

$$o_i = \sum_{j=1}^{T} a_{ij} v_j. \tag{3.29}$$

Thus o_i is not simply the original embedding of time i. It is a new contextual vector that mixes information from the whole trajectory according to the attention weights in row i. If a_{ij} is close to zero, position j contributes almost nothing to o_i; if a_{ij} is large, the value v_j becomes important for the representation of position i. In matrix form,

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V. \tag{3.30}$$

The factor $\sqrt{d_k}$ is important: without it, dot products grow in magnitude when d_k is large, pushing the softmax into saturated regions with small gradients [54].

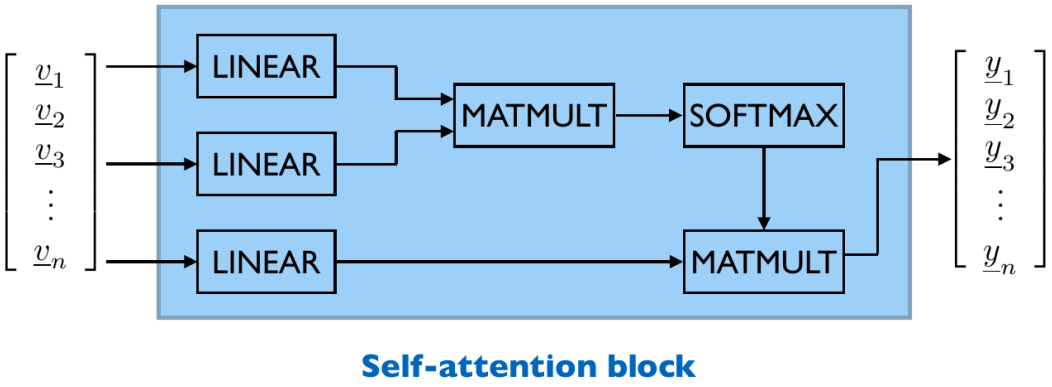


Figure 3.3: Self-attention block: the input sequence is projected into query, key and value representations; query–key products define attention weights, and these weights are applied to the values to produce contextualized output vectors.

Figure 3.3 summarizes the query-key-value mechanism: the attention weights are computed from query–key similarities and then used to mix the value vectors.

Multi-head attention performs this operation several times in parallel:

$$\text{head}_h = \text{Attention}(XW_Q^{(h)}, XW_K^{(h)}, XW_V^{(h)}), \tag{3.31}$$

$$\text{MultiHead}(X) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)W_O. \tag{3.32}$$

Each head has its own projections, so different heads can represent different relations. In a trajectory, one head may compare neighboring increments, another may relate early and late portions of the sequence, and another may focus on changes in displacement scale. The important point is not that these roles are manually imposed, but that the architecture gives the model several learned ways to compare positions.

Self-attention alone does not encode temporal order. If the rows of X are permuted, the attention computation is permuted in the same way. A Transformer therefore adds positional encodings to the input embeddings. Vaswani et al. use sinusoidal encodings:

$$\text{PE}_{(t,2i)} = \sin\left(\frac{t}{10000^{2i/d_{\text{model}}}}\right), \qquad \text{PE}_{(t,2i+1)} = \cos\left(\frac{t}{10000^{2i/d_{\text{model}}}}\right). \tag{3.33}$$

These encodings give the model access to absolute and relative position. For SPT, this is essential because the same local displacement pattern has a different meaning if it appears before or after a candidate transition.

A Transformer encoder layer then combines multi-head self-attention with a position-wise feed-forward network. In simplified notation,

$$Z = \text{LayerNorm}(X + \text{MultiHead}(X)), \tag{3.34}$$

$$Y = \mathrm{LayerNorm}(Z + \mathrm{FFN}(Z)), \tag{3.35}$$

where the feed-forward network is applied independently at each position:

$$\mathrm{FFN}(z) = \max(0, zW_1 + b_1)W_2 + b_2. \tag{3.36}$$

The residual connections preserve information from the previous representation, while layer normalization stabilizes the scale of hidden activations. Stacking several encoder layers lets the model build increasingly contextual sequence representations [54].

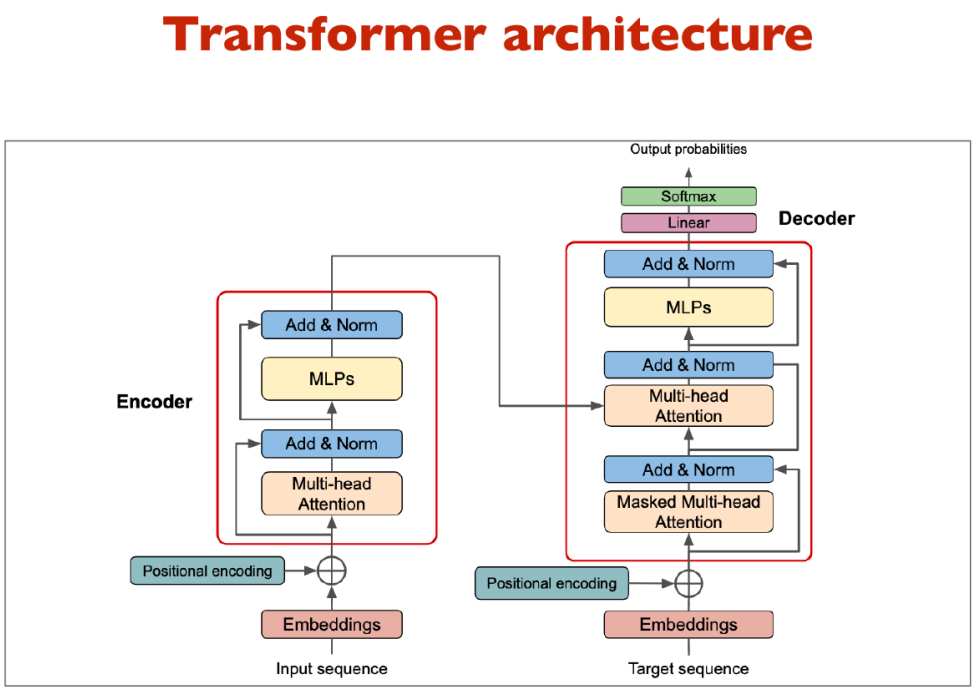


Figure 3.4: Transformer architecture: embeddings and positional encodings are processed by encoder and decoder blocks built from multi-head attention, feed-forward networks, residual connections and normalization. For the SPT setting discussed here, the encoder-style block is the most relevant part.

Figure 3.4 places the attention block inside the full Transformer architecture and shows how attention, feed-forward layers, residual connections and normalization are composed. Figures 3.2, 3.3 and 3.4 are taken from the Computational Science course slides by M. Weigt, J. F. De Cossio Diaz and F. Calvanese.

The advantage over recurrent models is that all positions can be processed in parallel. The cost is that standard self-attention computes all pairwise interactions, giving complexity

$$\mathcal{O}(T^2 d), \tag{3.37}$$

where T is the sequence length and d is the representation dimension. For the short SPT trajectories used here, with $T = 16$, this cost is small, while the ability to compare every pair of time points is valuable. For much longer trajectories, one may need downsampling, local attention or hybrid CNN–Transformer architectures.

The original Transformer also uses masking in the decoder so that an autoregressive prediction cannot see future tokens. For offline SPT analysis, the full trajectory is available. Therefore,

an encoder-style, non-causal self-attention block is more appropriate: the representation at time t can use both past and future displacements. This is particularly important for changepoint detection, where the evidence for a transition comes from comparing the regimes on both sides.

3.6 Why LSTM Models Are Less Convenient Here

Long short-term memory networks (LSTMs) are recurrent neural networks designed to store information over time and reduce the vanishing-gradient problem of classical RNNs [55]. For an input sequence $x_1, \dots, x_T$, an LSTM updates a hidden state and a cell state recursively. A standard form is

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \qquad f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \tag{3.38}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \qquad \tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c), \tag{3.39}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \qquad h_t = o_t \odot \tanh(c_t). \tag{3.40}$$

The gates i_t, f_t and o_t regulate writing, forgetting and reading from memory. This makes LSTMs expressive for temporal data, and variants such as ConvLSTM replace dense transformations with convolutions in order to process spatiotemporal signals [56].

The practical drawback is the recurrence itself. The hidden state h_t depends on h_{t-1}, so the computation inside one trajectory has a sequential dependency chain of length T. Even if many trajectories can be batched, the time dimension cannot be parallelized as directly as in a CNN or a Transformer. In contrast, a convolutional layer can apply filters to all positions at once, and a self-attention layer can compute all pairwise temporal relations in a small number of matrix operations. This is one of the main motivations of the Transformer architecture, which was introduced as a more parallelizable alternative to recurrent sequence models [54].

For the present SPT problem this matters for two reasons. First, the analysis is performed on many simulated trajectories, so training speed and GPU throughput are important. Second, the relevant information is often not strictly causal: to estimate a change point one wants to compare the region before and after the candidate transition. A bidirectional LSTM can use both sides, but it still pays the recurrent sequential cost. A CNN–Transformer model is therefore often a more convenient choice: convolutions extract local displacement patterns, while attention compares distant regions without scanning the sequence step by step. This does not mean that LSTMs are theoretically wrong for SPT; rather, they are less attractive when the goal is fast large-scale inference on many short trajectories.

3.7 Why Combine CNNs and Transformers

For SPT trajectories, a hybrid CNN–Transformer architecture is natural. Convolutions extract local displacement patterns, such as changes in amplitude or short-scale correlations. The Transformer integrates information along the whole trajectory, making it possible to compare distant regions and identify dynamical changes. The architecture used in this work follows this logic: convolutional/residual blocks followed by an attention-based encoder.

From the point of view of physical inference, this combination separates two scales. Convolutions are sensitive to local patterns such as

$$\|\Delta \mathbf{x}_t\|, \qquad \Delta \mathbf{x}_t \cdot \Delta \mathbf{x}_{t+1}, \qquad \|\Delta \mathbf{x}_{t+1} - \Delta \mathbf{x}_t\|. \tag{3.41}$$

Attention can compare local statistics that are far apart in time, for example before and after a possible change point. In a multi-state trajectory, this is crucial: the model must decide whether two regions belong to the same diffusive regime.

This division of labor mirrors the sources used in this chapter. From the CNN perspective of Mehta et al., the convolutional part encodes locality and weight sharing. From the attention perspective of Vaswani et al., the attention part lets every time point query the rest of the sequence for relevant context [49, 54].

Block	Role in the trajectory model	Output type
Input increments	Provide translation-invariant displacement information	$T \times 2$ sequence
CNN/XRes blocks	Extract local amplitude and short-range correlation features	Temporal feature map
Transformer encoder	Compare distant parts of the trajectory through self-attention	Contextual temporal representation
Linear regression head	Produce a value at each frame	α_t or $\log_{10}(D_t)$ profile
KernelCPD post-processing	Segment the predicted profile into mobility states	Estimated changepoint

Table 3.1: Functional role of the CNN–Transformer pipeline used in the results.

3.8 How Neural Networks Estimate D and α

The previous sections describe CNNs and Transformers as general sequence models. In SPT, however, their role is more specific: they approximate an inverse statistical map from a finite, noisy trajectory to the parameters of the underlying stochastic process [26, 25, 27]. Given a trajectory represented by increments,

$$X = \begin{pmatrix} dx_1 & dx_2 & \cdots & dx_{T-1} \\ dy_1 & dy_2 & \cdots & dy_{T-1} \end{pmatrix} \in \mathbb{R}^{2\times(T-1)}, \tag{3.42}$$

the network learns a function

$$f_\theta(X) \approx y, \qquad y \in \{D, \alpha, (\log_{10} D_t)_{t=1}^{T}\}. \tag{3.43}$$

For a global regression task, the output is a scalar, for example $\hat{\alpha}$ or $\widehat{\log_{10} D}$. For a local or segmentation task, the output is a sequence

$$f_\theta(X) = (\hat{z}_1, \ldots, \hat{z}_T), \qquad \hat{z}_t \approx \log_{10}(D_t). \tag{3.44}$$

This can be compared with classical Brownian estimation. In the Brownian case, D is controlled by the scale of the increments:

$$\mathbb{E}\|\Delta \mathbf{x}_t\|^2 = 2dD\Delta t. \tag{3.45}$$

A simple estimator therefore averages squared displacements. A CNN can learn a similar operation implicitly. For example, early convolutional filters can respond to local displacement magnitude,

$$r_t^2 = dx_t^2 + dy_t^2, \tag{3.46}$$

while deeper layers aggregate this information across time. Pooling and fully connected layers then convert local evidence into a global estimate. In this sense, a CNN can be viewed as a flexible, data-driven generalization of hand-crafted MSD features [1, 23, 26].

Estimating α is more subtle. The anomalous exponent is not determined only by the variance of one-step increments; it is related to how displacement statistics scale with lag:

$$\mathbb{E}\|\mathbf{x}_{t+\tau} - \mathbf{x}_t\|^2 \propto \tau^\alpha. \tag{3.47}$$

Therefore, a model must compare information across multiple temporal scales. A CNN can do this by stacking layers: the receptive field of a unit in a deeper layer covers several adjacent increments. If the network contains L layers with kernel size k and no dilation, the approximate receptive field is

$$R \simeq 1 + L(k-1). \tag{3.48}$$

With pooling, the effective receptive field grows even faster. This allows the CNN to learn features related to local variance, increment correlations and multi-lag scaling without explicitly computing the TAMSD [47, 48, 25].

Transformers approach the same problem differently. Self-attention computes pairwise interactions between all time points:

$$A_{ij} = \frac{\exp(q_i^\top k_j/\sqrt{d_k})}{\sum_{\ell=1}^{T} \exp(q_i^\top k_\ell/\sqrt{d_k})}. \tag{3.49}$$

The output at time i is a weighted combination of value vectors:

$$\tilde{v}_i = \sum_{j=1}^{T} A_{ij} v_j. \tag{3.50}$$

In trajectory analysis, this means that the representation at one frame can directly incorporate information from distant frames. This is useful for anomalous diffusion because long-range correlations are central in processes such as fBm, and it is useful for change point detection because the model can compare the dynamics before and after a candidate transition [45, 54, 25].

3.9 Input Scaling and Target Transformations

The numerical scale of the target matters. Diffusion coefficients can vary over several orders of magnitude, and in simulated SPT datasets they are often sampled over wide ranges. Regressing directly on D can make the loss dominated by large values of D. A common solution is to predict

$$z = \log_{10} D \tag{3.51}$$

instead of D. The inverse transformation is

$$\widehat{D} = 10^{\hat{z}}. \tag{3.52}$$

This makes multiplicative errors in D closer to additive errors in the training target. For example, predicting $D = 0.1$ instead of $D = 0.01$ and predicting $D = 10$ instead of $D = 1$ both correspond to an error of one decade in $\log_{10} D$. Log-scale parameterization is common when diffusion coefficients span orders of magnitude [24, 25].

3.10 BI-ADD: Bottom-Up Iterative Anomalous Diffusion Detector

BI-ADD is a recent state-of-the-art method designed specifically for molecular trajectories whose diffusive properties change over time [65]. It is important in the context of this thesis because it attacks the same physical problem as neural trajectory analysis: from a finite, noisy SPT trajectory, one wants to infer where the motion changes and what diffusion regime is present in each segment. The output is therefore not only a list of changepoints, but a piecewise physical description of the trajectory.

The model assumed in BI-ADD is fractional Brownian motion. If $B_H(t)$ is an fBm with Hurst exponent H, the covariance is

$$\mathbb{E}[B_H(t)B_H(s)] = \frac{1}{2}\left(|t|^\alpha + |s|^\alpha - |t-s|^\alpha\right), \qquad \alpha = 2H. \tag{3.53}$$

For an n-dimensional trajectory, the corresponding mean squared displacement scales as

$$\mathrm{MSD}(t) = 2nKt^\alpha, \tag{3.54}$$

where K is the generalized diffusion coefficient. Thus K fixes the overall spatial scale of the motion, while α fixes its temporal scaling and correlation structure. In BI-ADD, each segment m of a trajectory is described by a state

$$\theta_m = (\alpha_m, K_m). \tag{3.55}$$

A changepoint is a time index at which α, K, or both change:

$$(\alpha_t, K_t) = \begin{cases} (\alpha_1, K_1), & t < t_c, \\ (\alpha_2, K_2), & t \geq t_c. \end{cases} \tag{3.56}$$

Biologically, such a transition can correspond to a protein entering a more crowded region, binding to a substrate, leaving a membrane domain, or changing its interaction state.

The method is called bottom-up because it first deliberately over-segments a trajectory into candidate sub-trajectories, and then merges neighboring pieces that appear to belong to the same diffusive state. The key idea is that it is usually safer to detect too many possible changepoints at first and then remove the false positives using physical parameter estimates.

The first step is candidate changepoint generation. Given the observed coordinate sequence

$$\mathbf{x}_1, \ldots, \mathbf{x}_T, \qquad \mathbf{x}_t = (x_t, y_t), \tag{3.57}$$

BI-ADD extends the trajectory at both ends by reflecting the first and last coordinates. This reduces boundary artifacts and makes changepoints near the beginning or the end less likely to be missed. The trajectory is then examined with multiple sliding-window sizes, for example

$$w \in \{20, 22, \ldots, 40\}. \tag{3.58}$$

For a window centered around time i, the method compares the behavior of the two half-windows of size $w/2$. In simplified notation, one can write a local contrast

$$v_i^{(w)} = \Phi\big(\mathbf{x}_{i-w/2}, \ldots, \mathbf{x}_{i-1}\big) - \Phi\big(\mathbf{x}_i, \ldots, \mathbf{x}_{i+w/2-1}\big), \tag{3.59}$$

where Φ denotes a feature extracted from local Euclidean displacements. The exact signal used by BI-ADD is designed to respond to changes in the relative diffusive behavior of the two sides of the window, including changes in both α and K. The multi-scale changepoint signal is then obtained by aggregating the window-specific contrasts,

$$S_i = \frac{1}{|W|} \sum_{w \in W} v_i^{(w)}. \tag{3.60}$$

Local maxima of S_i above a threshold λ are selected as candidate changepoints [65].

The second step is local parameter estimation. If the candidate set contains

$$c_1 < c_2 < \cdots < c_N, \tag{3.61}$$

the trajectory is split into $N+1$ sub-trajectories. For each sub-trajectory $\mathcal{T}_m$, BI-ADD estimates

$$\widehat{\theta}_m = (\hat{\alpha}_m, \hat{K}_m). \tag{3.62}$$

The original implementation uses a ConvLSTM-based model for α and a separate neural network for K [65]. This separation is natural: K is mainly reflected in the magnitude of displacements, while α depends on temporal scaling and increment correlations across lags. In finite trajectories, especially short ones, α is therefore harder to estimate than K, because its signal is distributed over the temporal structure rather than being visible only from the instantaneous displacement scale.

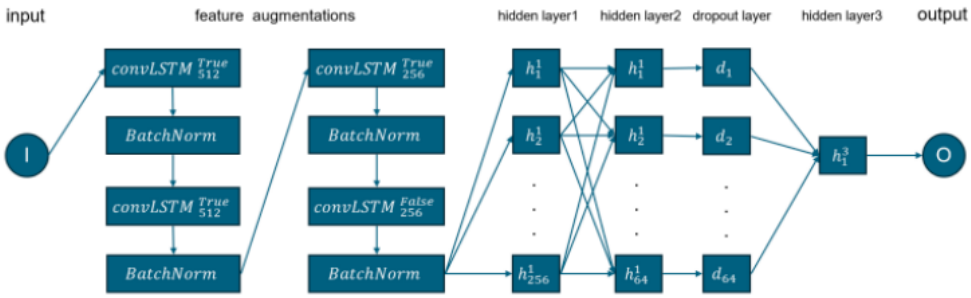


Figure 3.5: Architecture used in the BI-ADD article for local parameter estimation. This is the architecture reported in the original BI-ADD article and includes ConvLSTM blocks, batch-normalization layers, fully connected hidden layers, and dropout before the final output.

Figure 3.5 shows the neural architecture used in the BI-ADD article for the local estimation stage. It makes clear that BI-ADD relies on a dedicated ConvLSTM-based architecture to estimate local diffusion quantities on sub-trajectories before the subsequent clustering and bottom-up merging steps.

The third step is unsupervised state grouping. The estimated segment descriptors are embedded in the physical parameter space

$$\widehat{\mathbf{u}}_m = (\hat{\alpha}_m, \log \hat{K}_m). \tag{3.63}$$

BI-ADD clusters these points using a Gaussian mixture model. The number of diffusive states can be selected using the Bayesian information criterion or set manually when prior biological knowledge is available [65]. This clustering stage is important because a candidate changepoint is meaningful only if the two neighboring segments are assigned to distinct physical states.

The final step is the bottom-up merge. Suppose that a candidate changepoint separates two neighboring sub-trajectories $\mathcal{T}_m$ and $\mathcal{T}_{m+1}$. If their estimated parameters belong to the same GMM cluster, or are sufficiently close according to the BI-ADD decision rule, the changepoint is rejected and the two sub-trajectories are merged:

$$\mathcal{T}_{m,m+1} = \mathcal{T}_m \cup \mathcal{T}_{m+1}. \tag{3.64}$$

After a merge, α and K are re-estimated on the new longer segment. This matters because parameter estimates on short segments are noisy; merging can increase statistical stability and change the subsequent clustering decision. The algorithm repeats this procedure until no additional merge is accepted. In this sense, BI-ADD combines a classical signal-processing step, a supervised regression step and an unsupervised clustering step in a single iterative pipeline [65].

Figure 3.6 summarizes the complete BI-ADD workflow, from candidate changepoint generation to local parameter estimation, clustering and iterative merging.

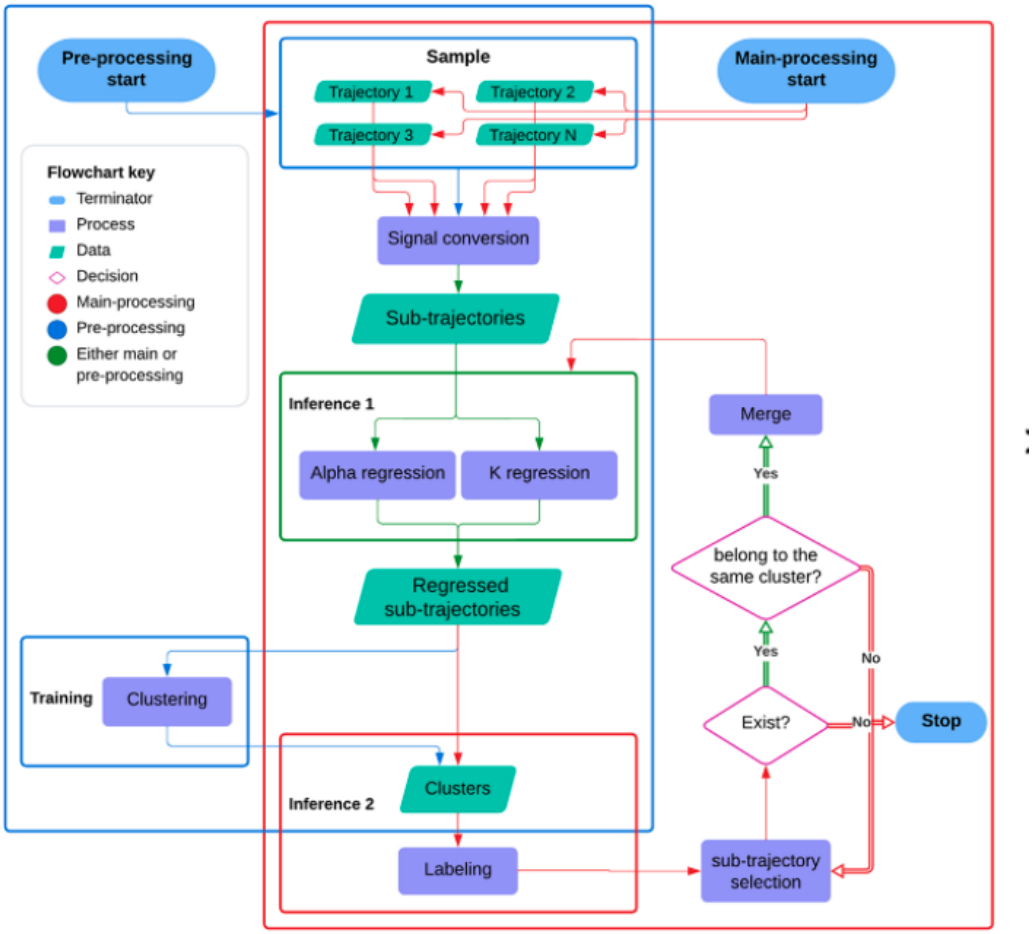


Figure 3.6: BI-ADD workflow: candidate changepoint detection, local estimation of α and K, GMM clustering and iterative merging of neighboring sub-trajectories.

This iterative design is useful because the first candidate-detection step is sensitive and may over-detect transitions, while the clustering and merging steps remove candidates that do not correspond to a statistically distinct diffusive state. It also gives a physically interpretable output: not only the changepoint positions, but also the estimated (α, K) values of each state.

The paper also makes clear why this problem is difficult. Changepoints caused by a large jump in K are easier to detect than changepoints caused only by a change in α, because changes in K immediately affect the amplitude of displacements, whereas changes in α require reliable information across time scales. Short trajectories and short sub-trajectories therefore increase uncertainty and can produce both false positives and false negatives [65].

There is also a computational cost. BI-ADD uses a ConvLSTM component for α estimation, and the merge decisions are performed sequentially along each trajectory. The authors report that memory and computation are mostly consumed by the α-estimation model and that the sequential merging mechanism limits parallelization [65]. This supports the choice, for the present thesis, to prefer architectures based on CNNs and Transformers when the goal is a fast predictor over many simulated trajectories. CNNs and Transformers are still expressive enough to learn local displacement features and long-range temporal dependencies, but most of their expensive operations can be batched as matrix operations on a GPU. BI-ADD is therefore best viewed here as an important methodological reference for heterogeneous anomalous diffusion, while the present approach implements a lighter direct regression strategy.

Chapter 4

Conformal Prediction

4.1 Motivation

A neural network usually returns a point prediction. For SPT, a point prediction is often insufficient: short trajectories, localization noise and imperfect simulation models make the estimate uncertain. Conformal prediction constructs prediction sets or intervals with finite-sample guarantees under weak assumptions, especially exchangeability [59, 60, 61, 64]. In this chapter the miscoverage level is denoted by ε, so that it is not confused with the anomalous diffusion exponent α.

The main appeal of conformal prediction is that it can be used as a wrapper around an already trained black-box model. The method does not require the model to be Bayesian, the residuals to be Gaussian, or the neural network to estimate the correct posterior distribution. Instead, it uses a held-out calibration set to transform model errors observed on past examples into prediction intervals for future examples [64]. This is why conformal prediction is often described as distribution-free uncertainty quantification: the validity comes from the calibration procedure and the exchangeability assumption, not from a parametric model of the data.

4.2 Prediction Sets, Scores and p-values

Let

$$Z_i = (X_i, Y_i), \qquad i = 1, \dots, N, \tag{4.1}$$

be examples drawn from the same experimental or simulated data-generating process. Here X_i is the input trajectory and Y_i is the target: it may be $\log_{10} D_i$, α_i, a pair $(\log_{10} D_i, \alpha_i)$, a whole temporal profile $(Y_{i,1}, \dots, Y_{i,T})$, or a change point $t_{c,i}$. Conformal prediction does not prescribe the neural architecture. It only needs a nonconformity score

$$S : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}, \tag{4.2}$$

where large values mean that the candidate label y is unusual for the input x according to the fitted model [60, 64]. The score is the main design choice. If the score ranks examples well, easy trajectories receive small intervals and difficult trajectories receive larger ones. If the score is poorly chosen, conformal prediction can still satisfy marginal coverage, but the resulting intervals may be too wide to be useful [64].

In the original full conformal construction, for a test trajectory x_{N+1} and a candidate label y, one forms the augmented dataset

$$\{(X_1, Y_1), \dots, (X_N, Y_N), (x_{N+1}, y)\} \tag{4.3}$$

and computes scores for all examples. The conformal p-value is

$$p(y) = \frac{1}{N+1} \left| \{ j : S_j \geq S_{N+1} \} \right|. \tag{4.4}$$

The prediction set at confidence $1-\varepsilon$ is then

$$C_\varepsilon(x_{N+1}) = \{y \in \mathcal{Y} : p(y) > \varepsilon\}. \tag{4.5}$$

This definition is conceptually important because it shows the core idea: labels are accepted if they do not make the new example look too nonconforming relative to previous examples. However, full conformal prediction can be computationally expensive for neural networks, because the model may need to be retrained or re-evaluated for many candidate labels. For this reason, practical deep-learning pipelines usually use split conformal prediction [61, 64].

4.3 Split Conformal Prediction

Split conformal prediction separates model fitting from calibration. The data are divided into a proper training set, a calibration set and a test set:

$$\mathcal{D} = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{cal}} \cup \mathcal{D}_{\text{test}}. \tag{4.6}$$

First, the neural network f_θ is trained only on $\mathcal{D}_{\text{train}}$. Second, predictions are computed on the calibration set and converted into scores. Third, the empirical quantile of these scores is used to inflate the prediction on new test trajectories. In practical terms, the procedure is:

$$\begin{aligned}&\text{train model} \longrightarrow \text{compute calibration scores} \longrightarrow \text{take a quantile}\\ &\longrightarrow \text{form prediction sets.}\end{aligned}$$

Only the first step involves model fitting. The calibration step is therefore cheap and can be applied after the neural network has already been trained [64].

For scalar regression, a simple score is the absolute residual

$$s_i = |y_i - \hat{y}_i|. \tag{4.7}$$

If the calibration set contains n examples, the conformal correction is the finite-sample quantile

$$k = \lceil (n+1)(1-\varepsilon) \rceil, \qquad Q = s_{(k)}, \tag{4.8}$$

where $s_{(k)}$ is the k-th ordered calibration score. If $k > n$, the exact finite-sample construction returns an unbounded interval; in practice this means that the calibration set is too small for the requested miscoverage level. For a new point x, the conformal interval is

$$C(x) = [\hat{y}(x) - Q, \hat{y}(x) + Q]. \tag{4.9}$$

The crucial point is that Q is not learned by gradient descent. It is a post-hoc calibration constant determined by the empirical error distribution on held-out data [61, 64].

More generally, conformal prediction starts from a score function

$$S : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}, \tag{4.10}$$

where $S(x, y)$ measures how nonconforming the pair (x, y) is. In standard regression,

$$S(x, y) = |y - \hat{\mu}(x)|. \tag{4.11}$$

If the noise is heteroscedastic, one can use a normalized score:

$$S(x, y) = \frac{|y - \hat{\mu}(x)|}{\hat{\sigma}(x)}, \tag{4.12}$$

where $\hat{\sigma}(x)$ estimates local uncertainty. The interval becomes

$$C(x) = [\hat{\mu}(x) - Q\hat{\sigma}(x), \hat{\mu}(x) + Q\hat{\sigma}(x)]. \tag{4.13}$$

This variant can produce narrower intervals where the problem is easy and wider intervals where the problem is difficult [61, 64].

4.4 Coverage Guarantee

Under exchangeability between calibration and test examples, split conformal prediction guarantees

$$\mathbb{P}\left(Y_{\text{test}} \in C(X_{\text{test}})\right) \geq 1 - \varepsilon. \tag{4.14}$$

This guarantee is marginal: it means that the average coverage over new examples is at least $1 - \varepsilon$. It does not automatically imply conditional coverage for every subgroup, every value of D, every noise level or every change point position.

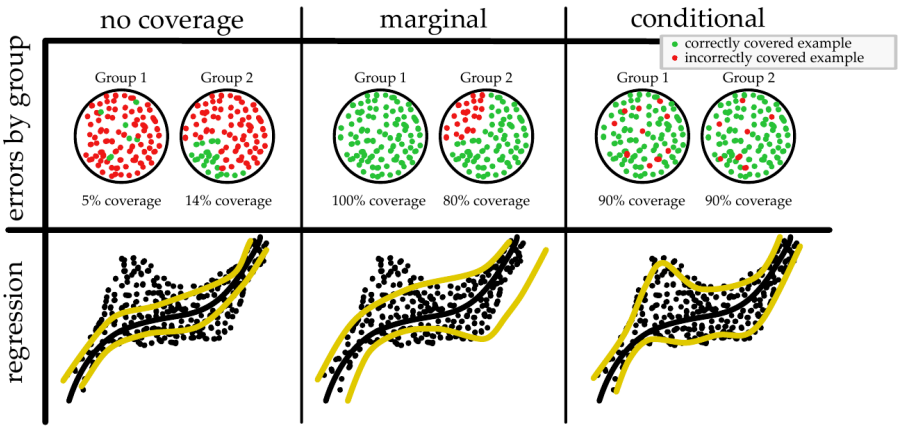


Figure 4.1: Different notions of coverage. Marginal coverage controls the average error rate, whereas conditional coverage would require the same error rate within each relevant subgroup or region of the input space. Figure adapted from Angelopoulos and Bates [64].

Figure 4.1 illustrates why marginal coverage is both useful and limited. In the middle panel the total coverage can be close to the nominal level even though the errors are concentrated in one group. Conditional coverage would avoid this behavior, but it is much stronger: it asks for valid coverage after conditioning on the input, or at least on meaningful subgroups of the input space [63, 64].

In this thesis the conformal guarantee is used in its marginal form. This is the appropriate baseline because the calibration trajectories are treated as exchangeable samples from the same simulation procedure as the test trajectories, and because the available calibration set is not large enough to calibrate separate guarantees for every combination of D, α, trajectory length, noise level and changepoint position. A conditional or group-balanced conformal method would require defining such groups and keeping enough calibration examples in each of them. Here, instead, marginal coverage provides a clear global uncertainty guarantee, while subgroup behavior is left to empirical diagnostics.

Exchangeability is weaker than assuming a known probability distribution. It means that the joint distribution of the calibration examples and the future test example is invariant under permutation. Independent and identically distributed data are exchangeable, but the conformal guarantee itself does not require knowing the distribution that generated them [59, 64].

The reason for the guarantee is a rank argument. A short proof is the following. Let

$$S_i = S(X_i, Y_i), \qquad i = 1, \ldots, n, \tag{4.15}$$

be the calibration scores, and let

$$S_{n+1} = S(X_{n+1}, Y_{n+1}) \tag{4.16}$$

be the score that would be obtained on a future test example if its true label were known. In split conformal regression with score $S(x, y) = |y - \hat{\mu}(x)|$, the interval

$$C(X_{n+1}) = [\hat{\mu}(X_{n+1}) - Q, \hat{\mu}(X_{n+1}) + Q] \tag{4.17}$$

covers the true label exactly when

$$Y_{n+1} \in C(X_{n+1}) \quad \Longleftrightarrow \quad |Y_{n+1} - \hat{\mu}(X_{n+1})| \leq Q \quad \Longleftrightarrow \quad S_{n+1} \leq Q. \tag{4.18}$$

Therefore the coverage probability is

$$\mathbb{P}\{Y_{n+1} \in C(X_{n+1})\} = \mathbb{P}\{S_{n+1} \leq Q\}. \tag{4.19}$$

Now order the calibration scores:

$$S_{(1)} \leq S_{(2)} \leq \cdots \leq S_{(n)}. \tag{4.20}$$

Split conformal prediction sets

$$Q = S_{(k)}, \qquad k = \lceil (n+1)(1-\varepsilon) \rceil. \tag{4.21}$$

Assume first, for simplicity, that there are no ties among the $n+1$ scores. Because the calibration examples and the test example are exchangeable, the scores

$$S_1, \ldots, S_n, S_{n+1} \tag{4.22}$$

are exchangeable as well. Hence the rank

$$R_{n+1} = |\{i \in \{1, \ldots, n+1\} : S_i \leq S_{n+1}\}| \tag{4.23}$$

is uniformly distributed on $\{1, \ldots, n+1\}$. The event $S_{n+1} \leq Q$ means that the test score is not larger than the k-th smallest calibration score. A conservative way to see this is to compare ranks among all $n+1$ scores:

$$\mathbb{P}\{S_{n+1} \leq Q\} \geq \mathbb{P}\{R_{n+1} \leq k\} = \frac{k}{n+1}. \tag{4.24}$$

By the definition of k,

$$\frac{k}{n+1} \geq 1 - \varepsilon. \tag{4.25}$$

Combining the previous equations gives

$$\mathbb{P}\{Y_{n+1} \in C(X_{n+1})\} \geq 1 - \varepsilon. \tag{4.26}$$

If ties occur, the same conclusion holds with conservative tie handling, or exactly with randomized tie-breaking. This is the strength of conformal prediction: the guarantee does not require the neural network to be correct and does not require Gaussian errors [61, 64].

The distinction between marginal and conditional coverage is important in applications. Marginal coverage may be correct on average while still being uneven across subgroups. For instance, a method can cover well for trajectories with large D and poorly for trajectories with small D, while still achieving the nominal average coverage. Therefore, conformal prediction gives a formal baseline guarantee, but empirical checks across relevant regimes remain necessary [63, 64].

4.5 Conformal Prediction on Trajectories

The most relevant use of conformal prediction in this work is not an advanced variant of the method, but the direct calibration of neural-network predictions on trajectories. The trained model receives a trajectory x_i and returns a prediction $\hat{y}_i$. Depending on the task, y_i may be a single scalar, such as α or $\log_{10} D$, or a sequence of frame-wise labels along the trajectory.

Conformal prediction is then applied after training: the neural network is kept fixed, and the calibration set is used only to measure how large its errors typically are [61, 64].

Although the construction can in principle be applied to either α or $\log_{10} D$, the final analysis applies and visualizes the conformal bands only for $\log_{10}(D_t)$. This is consistent with the physical focus of the results: the local diffusion coefficient is the main signal used for the changepoint analysis because it measures changes in mobility scale, while α is treated as a complementary descriptor and is evaluated through prediction and JSC plots rather than through a conformal-band figure.

For a scalar output, the calibration score can simply be

$$s_i = |y_i - \hat{y}_i|. \tag{4.27}$$

After sorting the calibration scores, the conformal quantile Q is chosen at level $1 - \varepsilon$. A new prediction is then reported as

$$C(x) = [\hat{y}(x) - Q, \hat{y}(x) + Q]. \tag{4.28}$$

Thus the final result is not only a point estimate, but an interval whose width is determined by previous calibration errors. This is the idea used when a model predicts one diffusion parameter for the whole trajectory.

For trajectory outputs, the target and the prediction are sequences:

$$y_i = (y_{i,1}, \ldots, y_{i,T}), \qquad \hat{y}_i = (\hat{y}_{i,1}, \ldots, \hat{y}_{i,T}). \tag{4.29}$$

If one calibrated each frame independently, the resulting intervals would only give pointwise statements. In change point analysis, however, the entire temporal profile matters. A small error at many frames may be acceptable, while a large local error near a transition can move the estimated changepoint. Therefore a natural trajectory-level score is the maximum absolute error along the curve:

$$s_i = \max_{t=1,\ldots,T} |y_{i,t} - \hat{y}_{i,t}|. \tag{4.30}$$

The calibrated band for a new trajectory is then

$$C_t(x) = [\hat{y}_t(x) - Q, \hat{y}_t(x) + Q], \qquad t = 1, \ldots, T. \tag{4.31}$$

The maximum in the score is the essential point: Q is selected so that, on average over future trajectories drawn from the same distribution, the whole true curve is covered by the band with probability at least $1-\varepsilon$. This gives a simultaneous trajectory-level uncertainty statement rather than a collection of unrelated frame-wise intervals.

In the final analysis, this is useful because the model output is interpreted as a temporal signal. If the predicted signal changes sharply, it suggests a possible transition between diffusive states. The conformal band indicates how stable that interpretation is: a narrow band around the transition suggests a more reliable change, while a wide band means that the network prediction is too uncertain to localize the transition precisely. Conformal prediction therefore does not replace the neural network. It wraps the trained network with a calibrated measure of uncertainty, making the final trajectory analysis more transparent and easier to report.

Score used in this work

For the frame-wise diffusion profile, the nonconformity score used in the final calibration is

$$s_i = \max_t |z_{i,t} - \hat{z}_{i,t}|, \tag{4.32}$$

where $z_{i,t}$ is the true $\log_{10}(D_t)$ profile and $\hat{z}_{i,t}$ is the neural prediction. The maximum over time is deliberately conservative: it calibrates a simultaneous band for the whole trajectory rather than independent pointwise intervals for each frame. This is appropriate for changepoint analysis, because the relevant object is the full temporal profile.

Chapter 5

Results

5.1 Preprocessing and Datasets

I use `csv` files containing two-dimensional trajectories, both clean and noisy, as well as simulated datasets generated with `andi_datasets`. The first operational step is to convert coordinates into increments. For each temporally ordered trajectory:

$$dx_t = x_t - x_{t-1}, \qquad dy_t = y_t - y_{t-1}. \tag{5.1}$$

The final dataset contains `traj_idx`, `frame_start`, `frame_end`, `dx`, `dy`, α and D. For a trajectory of length T, the model receives a sequence of length $T - 1$.

Setting	Value
Trajectory length	$T = 16$
Spatial dimension	$d = 2$
Simulated trajectories	1500 (AnDi dataset)
Train/validation/test split	75%/15%/10%
Calibration set	validation split, after model selection
Diffusion-coefficient range	$\log_{10}(D) \in [-3, 1]$
Anomalous-exponent range	$\alpha \in [0, 2]$
Final evaluation length	fixed at $T = 16$ frames

Table 5.1: Simulation and dataset settings used for the final results.

This transformation incorporates a useful invariance. If the whole trajectory is translated by a constant vector $\mathbf{a}$,

$$\mathbf{x}'_t = \mathbf{x}_t + \mathbf{a}, \tag{5.2}$$

then

$$\Delta\mathbf{x}'_t = \mathbf{x}'_{t+1} - \mathbf{x}'_t = \Delta\mathbf{x}_t. \tag{5.3}$$

The model does not need to learn absolute position: the inputs directly contain the dynamics. This is important when trajectories are generated or observed in different spatial regions.

5.2 Diffusion-Parameter Estimation with CNNs

The CNN is used to estimate α, D, or both, from increments. The input has shape

$$X \in \mathbb{R}^{2\times(T-1)}. \tag{5.4}$$

I consider scenarios where D varies and α is fixed, where α varies and D is fixed, and where both vary. Predictions are grouped by the true parameter value, and the mean and standard

deviation of the predictions are computed. This makes it possible to inspect not only the average error, but also systematic bias in specific regions of the parameter range.

For scalar regression, the model implements

$$\hat{y} = f_\theta(X), \qquad y \in \{D, \alpha\}. \tag{5.5}$$

During training, the average loss over a batch of size B is

$$\mathcal{L}_{\text{batch}}(\theta) = \frac{1}{B} \sum_{i=1}^{B} \ell\left(y_i, f_\theta(X_i)\right). \tag{5.6}$$

The difficulty of the problem depends strongly on T. For small T, the variance of estimators is high and many trajectories with different parameters can produce similar increment sequences. For larger T, the model observes more samples of the dynamics and can better distinguish variance, correlations and local changes.

5.3 Conformal Prediction for Homogeneous Trajectories

Before considering trajectories with changepoints, I first study a simpler setting in which each trajectory is generated with constant parameters. In this case there is no local state transition: the whole trajectory is described by a single pair (D, α). This experiment is useful as a baseline because it separates the difficulty of parameter estimation from the additional difficulty of localizing a transition in time.

For these homogeneous trajectories, the CNN produces scalar predictions

$$\hat{D} = f_{\theta,D}(X), \qquad \hat{\alpha} = f_{\theta,\alpha}(X), \tag{5.7}$$

or, when the diffusion coefficient is treated logarithmically,

$$\widehat{\log_{10} D} = f_{\theta,D}(X). \tag{5.8}$$

Split conformal prediction is then applied after training. Given a calibration set, the nonconformity scores are

$$s_i^{(D)} = |D_i - \hat{D}_i|, \qquad s_i^{(\alpha)} = |\alpha_i - \hat{\alpha}_i|, \tag{5.9}$$

or the analogous score in $\log_{10} D$ if the target is represented on a logarithmic scale. The conformal quantiles Q_D and Q_α define the prediction intervals

$$C_D(X) = [\hat{D} - Q_D, \hat{D} + Q_D], \qquad C_\alpha(X) = [\hat{\alpha} - Q_\alpha, \hat{\alpha} + Q_\alpha]. \tag{5.10}$$

These intervals quantify how uncertain the CNN is when estimating global diffusion parameters from short trajectories without changepoints. Figures 5.1 and 5.2 show the corresponding conformal-calibration examples for α and D.

This baseline has two roles. First, it checks whether the CNN can estimate global diffusion parameters with calibrated uncertainty when the trajectory has a single dynamical state. Second, it motivates the next step of the analysis. When changepoints are present, a single scalar prediction is no longer enough: the model must return a temporal profile such as $(\log_{10} D_t)_{t=1}^{T}$ or $(\alpha_t)_{t=1}^{T}$. For this reason, I then introduce a Transformer-based sequence model. The convolutional part extracts local displacement features, while the attention mechanism lets each frame use information from the whole trajectory before producing a frame-wise prediction.

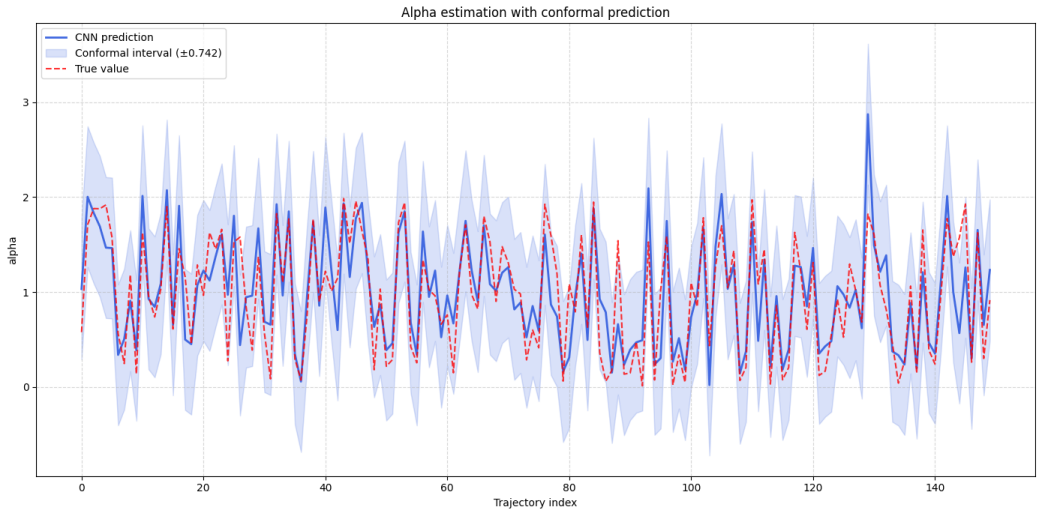


Figure 5.1: Split conformal prediction for α on homogeneous trajectories without changepoints. The horizontal axis indexes test trajectories and the vertical axis reports the anomalous exponent; the true values are compared with CNN predictions and conformal intervals. The plot should be read as a calibration check for global α estimation rather than as a changepoint result.

5.4 CNN–Transformer Architecture for Change Point Detection

For change point detection, I use a CNN–Transformer model. The important point is that this is a sequence-to-sequence architecture: it receives a whole trajectory and returns a prediction at each time point. In the present analysis, the target is the local value of $\log_{10}(D_t)$. Therefore, the network does not directly output a changepoint label. It outputs a temporal diffusion profile, and a changepoint is then inferred from a sharp variation of that profile.

The analysis focuses more strongly on D than on α for both biological and statistical reasons. Biologically, the diffusion coefficient is the most direct observable of molecular mobility. It fixes the spatial scale of the trajectory over the acquisition time: a larger D means that the protein explores more space per unit time, whereas a smaller D means that the same protein is effectively slowed down. This change in mobility scale can be linked, with the usual experimental cautions, to effective viscosity, molecular size, protein-complex formation, transient binding, confinement, chromatin association, membrane association or residence in a denser cellular region [3, 4, 22, 41, 37, 38, 40]. For instance, a protein that enters a bound, chromatin-associated or confined state is expected to show a reduced apparent diffusion coefficient over the observation window. Model-based SPT tools such as Spot-On explicitly use diffusion coefficients and state fractions to distinguish mobile and bound populations [22]. In this sense, D is not merely a fitting parameter: it is the quantity that most directly encodes the biological question of whether a molecule is mobile, slowed, confined or effectively immobile.

The anomalous exponent α is also biologically informative, because it describes whether the motion is Brownian-like, subdiffusive or persistent. However, its interpretation is less specific. Several microscopic mechanisms can produce the same value of α, including crowding, viscoelasticity, confinement, transient interactions and heterogeneous environments [33, 35, 37, 38, 40, 44]. Therefore, α is best used here as a descriptor of the transport regime, not as the main variable for defining a biological state. The central biophysical question of the results is whether the molecule changes mobility state along the trajectory, and that question is more directly expressed as a change in $\log_{10}(D_t)$.

There is also a practical reason for this choice. On trajectories of only $T = 16$ frames, changes in D immediately affect the amplitude of the observed displacements. Changes in α, instead, are

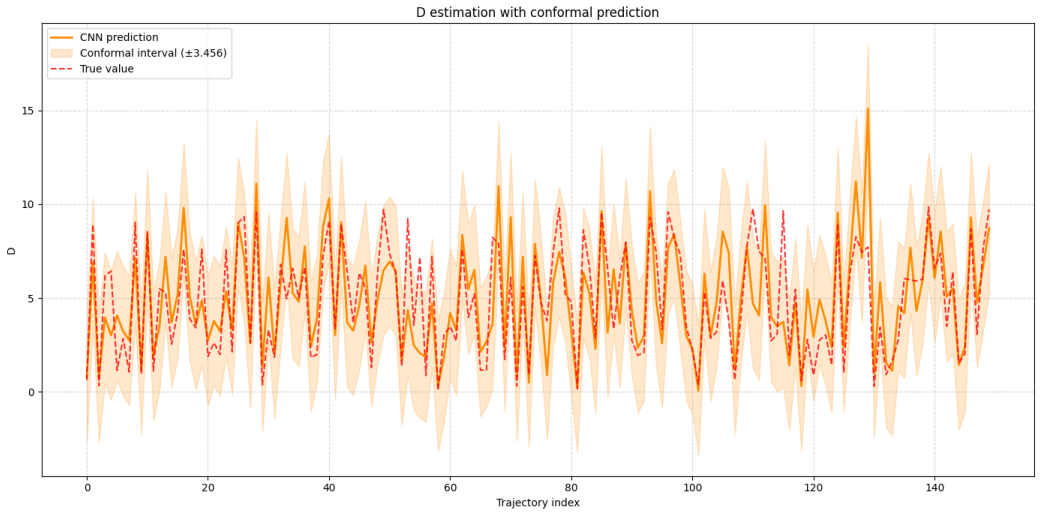


Figure 5.2: Split conformal prediction for D on homogeneous trajectories without changepoints. The horizontal axis indexes test trajectories and the vertical axis reports the diffusion coefficient; the true values are compared with CNN predictions and conformal intervals. Figure 5.2 shows how uncertainty calibration behaves for global mobility estimates before introducing changepoints.

expressed through temporal correlations and scaling over lag time. Estimating such correlations requires more temporal information, so α-based changepoint detection is intrinsically harder in the short-trajectory regime. A more detailed analysis of $\log_{10}(D_t)$ is therefore the most stable way to evaluate local state changes in the present dataset.

This hierarchy between D and α is also consistent with the range of α values reported in cellular SPT studies. Freely diffusing proteins and small protein complexes in intracellular aqueous environments are often modeled as Brownian, with $\alpha \simeq 1$, whereas anomalous diffusion is more commonly associated with interactions with cellular structures, active motion, confinement or viscoelastic/crowded environments [41, 40]. Reported subdiffusive exponents in bacterial cells are typically below one but still spread over a broad interval: RNA–protein complexes have been reported around $\alpha \simeq 0.5$–0.7, chromosomal loci around $\alpha \simeq 0.3$–0.4, and cytoplasmic proteins in recent experimental analyses around $\alpha \simeq 0.6$–0.8 [42]. Thus α is biologically meaningful because it indicates whether motion is Brownian-like or anomalous, but small differences in α are not always easy to map to a unique molecular mechanism. Different models can produce the same MSD scaling exponent, and localization or motion-blur artifacts can bias the inferred value [44, 42]. By contrast, changes in $\log_{10} D$ directly report changes in mobility scale and are therefore more useful here for detecting transitions between mobile, slow and bound-like states. The role of α is consequently not discarded, but subordinated to this physical interpretation: it helps qualify the type of motion once the main mobility changes have been identified from D.

The input can be represented as a batch of two-dimensional trajectories,

$$X \in \mathbb{R}^{B\times T\times d}, \qquad d = 2, \tag{5.11}$$

where B is the batch size, T is the trajectory length and the last dimension contains the spatial coordinates or increments. The model maps this input to

$$\hat{\mathbf{z}} = f_\theta(X), \qquad \hat{\mathbf{z}} \in \mathbb{R}^{B\times T\times 1}, \tag{5.12}$$

where $\hat{z}_{i,t}$ is the predicted value of $\log_{10}(D_{i,t})$ for trajectory i at frame t. This pointwise output is what makes the method appropriate for heterogeneous trajectories: the same model can describe constant diffusion, abrupt switching between states and smoother temporal variations.

The architecture used in this thesis follows the same three-stage design summarized in the implementation notes. The input is a trajectory of length T, written as

$$x_1, \dots, x_T \in \mathbb{R}^d, \tag{5.13}$$

with $d = 2$ for planar trajectories, and the output is a sequence of predicted diffusion parameters, one value per time step. In practice, the network is fed with displacement increments rather than absolute coordinates, so that the input focuses on local motion and is invariant to translations of the trajectory.

The first module is convolutional. One-dimensional convolutions are applied along the temporal axis, with batch normalization after each convolution and ReLU activations. A stem convolution first maps the input sequence to 64 filters. It is followed by residual blocks with 128, 256 and 512 filters, so that the last convolutional representation has embedding size 512. If $H^{(0)} = X$, a typical residual block has the form

$$H^{(\ell+1)} = \mathrm{ReLU}\Big(\mathrm{BN}\big(F_\ell(H^{(\ell)})\big) + G_\ell(H^{(\ell)})\Big), \tag{5.14}$$

where F_ℓ denotes the convolutional branch, G_ℓ the skip connection, and BN batch normalization. This module captures local displacement patterns and short-range temporal correlations, while progressively building a latent representation

$$H_{\mathrm{cnn}} \in \mathbb{R}^{B\times T\times 512}. \tag{5.15}$$

The second module is a self-attention block that captures long-range dependencies. The convolutional embeddings are passed to a Transformer encoder made of 4 encoder blocks, each with 8 attention heads. For one head, the attention map is computed as

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V, \tag{5.16}$$

where Q, K and V are learned projections of the convolutional embeddings. Through this operation, each frame can integrate information from all other frames of the same trajectory. This is important in SPT because the mobility at one time step should not be inferred from one displacement alone, but from the broader temporal context.

The third module is a feedforward prediction head. Pointwise fully connected layers are applied to the Transformer output at each time step to predict the target diffusion quantity. Denoting by H_{tr} the output of the Transformer encoder, the prediction is

$$\hat{z}_{i,t} = h_\phi(H_{\mathrm{tr},i,t}), \qquad t = 1, \dots, T_i. \tag{5.17}$$

For the $\log_{10}(D_t)$ task, a scaled sigmoid is used in the last layer to constrain the output to the physical range

$$\log_{10}(D_t) \in [-3.1,\, 1.1]. \tag{5.18}$$

A scaled sigmoid is also used for the α_t task, but with the output range adapted to the physical interval

$$\alpha_t \in [0.05,\, 2.05]. \tag{5.19}$$

Thus, the same CNN–Transformer backbone is used for both targets, while the final activation is scaled according to the parameter being predicted.

Overall, the pipeline can be read as follows: the convolutional module extracts local features, the self-attention module integrates global temporal information, and the feedforward module returns a frame-wise profile of $\log_{10}(D_t)$ or α_t. This hybrid design is well suited to short SPT trajectories, where local displacement statistics and long-range temporal context are both needed to detect true mobility changes.

Training is supervised. Each simulated trajectory x_i is paired with a frame-wise target profile $y_{i,t}$, either $\alpha_{i,t}$ or $\log_{10}(D_{i,t})$. The parameters are optimized by minimizing the trajectory-averaged mean absolute error

$$\mathcal{L}_{\mathrm{MAE}}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{T_i}\sum_{t=1}^{T_i}\left|y_{i,t} - f_\theta(x_i)_t\right|, \tag{5.20}$$

where n is the number of trajectories in the training batch and T_i is the length of trajectory i. In the experiments reported here, the final trajectory length is fixed to $T_i = 16$, so this loss reduces to the usual mean absolute error over all frames and trajectories. The model is trained with the ADAM optimizer, using a learning rate of 10^{-4}. This loss is appropriate for the present regression task because it penalizes frame-wise errors directly while being less sensitive to isolated large deviations than a squared-error objective.

For $T = 16$, I use 1500 simulated two-dimensional trajectories with one changepoint. Diffusion coefficients are sampled over a logarithmic range and localization noise is used as data augmentation. The dataset is split into training, validation and test subsets with proportions 75%/15%/10%. After model selection, the validation subset is also used as the calibration set for split conformal prediction.

Component	Choice used in the final pipeline
Input representation	displacement increments (dx_t, dy_t)
Neural target	frame-wise α_t or $\log_{10}(D_t)$
Training loss	mean absolute error / L_1 loss
Optimizer	ADAM with learning rate 10^{-4}
Changepoint estimator	KernelCPD applied to the predicted temporal profile
Conformal score	$\max_t \lvert z_t - \hat{z}_t\rvert$ for the $\log_{10}(D_t)$ profile
Nominal conformal level	95% trajectory-level coverage
Main diagnostic	JSC curves and binned frame-wise confusion matrices

Table 5.2: Summary of the final experimental protocol.

A very simple way to convert the predicted sequence into a change point estimate would be to search for the largest local jump,

$$\hat{t}_c = \arg\max_t |\hat{z}_{t+1} - \hat{z}_t|. \tag{5.21}$$

This rule is easy to interpret, but it is too local: a single noisy frame can produce a large finite difference even when the surrounding trajectory does not support a real change of state. For this reason, I estimate changepoints with kernel change point detection using the `ruptures` implementation, applied to the predicted temporal profile. This turns changepoint localization into a global segmentation problem rather than a single-frame thresholding problem [57, 58].

Let

$$y_t = \hat{z}_t, \qquad t = 1, \ldots, T, \tag{5.22}$$

be the one-dimensional signal obtained from the neural network, where y_t represents the predicted local value of $\log_{10}(D_t)$. More generally, $y_t \in \mathbb{R}^p$ could contain several predicted quantities. A segmentation with K changepoints is a set of ordered indices

$$\mathcal{T} = \{\tau_1, \ldots, \tau_K\}, \qquad 0 = \tau_0 < \tau_1 < \cdots < \tau_K < \tau_{K+1} = T. \tag{5.23}$$

The signal is then split into homogeneous segments

$$y_{\tau_j..\tau_{j+1}} = \{y_t : \tau_j < t \leq \tau_{j+1}\}. \tag{5.24}$$

Offline changepoint detection chooses the segmentation that minimizes an additive contrast:

$$V(\mathcal{T}) = \sum_{j=0}^{K} c\big(y_{\tau_j..\tau_{j+1}}\big)\,, \tag{5.25}$$

where $c(\cdot)$ is a segment cost. If the number of changepoints is known, the optimization problem is

$$\widehat{\mathcal{T}} = \arg\min_{|\mathcal{T}|=K} V(\mathcal{T}). \tag{5.26}$$

If K is unknown, one instead minimizes a penalized criterion,

$$\widehat{\mathcal{T}} = \arg\min_{\mathcal{T}} \left\{V(\mathcal{T}) + \mathrm{pen}(\mathcal{T})\right\}, \tag{5.27}$$

where the penalty prevents over-segmentation by making additional changepoints costly [58].

The kernel version is useful because it does not only look for a change in the Euclidean mean of the observed signal. It maps each sample y_t into a reproducing kernel Hilbert space $\mathcal{H}$ through a feature map ϕ, associated with a positive definite kernel

$$k(y_s, y_t) = \langle \phi(y_s), \phi(y_t) \rangle_{\mathcal{H}}. \tag{5.28}$$

A segment is considered homogeneous when the mapped samples $\phi(y_t)$ are well represented by a single empirical mean in $\mathcal{H}$:

$$\bar{\mu}_{a..b} = \frac{1}{b-a} \sum_{t=a+1}^{b} \phi(y_t). \tag{5.29}$$

The corresponding kernel segment cost is

$$c_{\mathrm{kernel}}(y_{a..b}) = \sum_{t=a+1}^{b} \left\|\phi(y_t) - \bar{\mu}_{a..b}\right\|_{\mathcal{H}}^2. \tag{5.30}$$

Using the kernel trick, this cost can be computed without explicitly evaluating ϕ:

$$c_{\mathrm{kernel}}(y_{a..b}) = \sum_{t=a+1}^{b} k(y_t, y_t) - \frac{1}{b-a} \sum_{s,t=a+1}^{b} k(y_s, y_t). \tag{5.31}$$

For the radial basis function kernel,

$$k(y_s, y_t) = \exp\big(-\gamma \|y_s - y_t\|^2\big)\,, \qquad \gamma > 0, \tag{5.32}$$

the cost becomes

$$c_{\mathrm{rbf}}(y_{a..b}) = (b-a) - \frac{1}{b-a} \sum_{s,t=a+1}^{b} \exp\big(-\gamma \|y_s - y_t\|^2\big)\,. \tag{5.33}$$

Minimizing the sum of these costs favors segments whose samples are close in the kernel feature space. A changepoint is introduced when splitting the signal into two parts substantially decreases the total within-segment kernel dispersion. Equivalently, kernel change point detection searches for changes in the mean embedding of the local distribution of the predicted signal. With a characteristic kernel such as the Gaussian/RBF kernel, a change in distribution corresponds to a change in this mean embedding, which explains why the method can detect more general changes than a simple mean-shift detector [57, 58].

In the present setting, the input to `KernelCPD` is the predicted $\log_{10}(D_t)$ profile. If the molecule switches from a low-diffusivity state to a high-diffusivity state, the distribution of

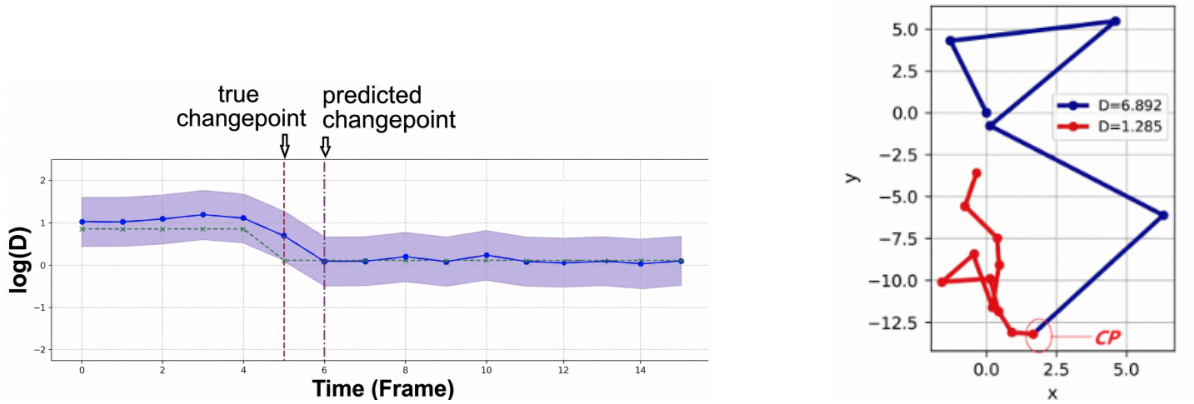


Figure 5.3: Example of changepoint detection on a short trajectory. Left: true and predicted $\log_{10}(D_t)$ profiles as functions of time, together with the conformal uncertainty band and the true and predicted changepoint positions. In the axis label of the plot, $\log(D)$ denotes the logarithmic diffusion profile, corresponding to $\log_{10}(D_t)$ in the notation used in the text. Right: the corresponding two-dimensional trajectory, with the two mobility states shown in different colors and the changepoint (CP) marked on the path. The combined figure highlights how a mobility transition in physical space is reflected in the predicted diffusion profile.

the predicted values before and after the transition changes. The kernel cost detects this by comparing whole candidate segments, not only adjacent frames. This makes the procedure more robust than the finite-difference rule above, especially when the neural prediction is noisy but still contains a persistent change in level.

Figure 5.3 connects the temporal $\log_{10}(D_t)$ profile to the corresponding two-dimensional trajectory, making explicit how a change in mobility appears both in physical space and in the predicted diffusion signal.

5.5 Conformal Prediction on the Results

In the results shown here, conformal prediction is applied to the frame-wise $\log_{10}(D_t)$ profile. No conformal-band figure is reported for α_t. This distinction is important: the conformal prediction chapter describes the method in general, but the numerical and visual uncertainty analysis here is focused on the diffusion-coefficient profile, which is the main signal used for changepoint detection.

For $T = 16$, split conformal prediction at the trajectory level gives

$$Q = 1.2500 \tag{5.34}$$

for a 95% target coverage, with empirical coverage 97.60%. This indicates slightly conservative intervals.

The fact that the empirical coverage is above the target indicates conservative calibration: the intervals cover more often than necessary, probably because the conformal quantile must protect difficult trajectories. In the short-track regime, this is expected. With only 16 frames, a single uncertain segment or a noisy displacement can dominate the trajectory-level score, increasing the width of the calibrated band.

5.6 Evaluation Plots

The final evaluation uses two complementary visual summaries. First, the Jaccard similarity coefficient (JSC) is used to evaluate the quality of changepoint detection. Given the set of true

changepoints $\mathcal{C}$ and the set of predicted changepoints $\widehat{\mathcal{C}}$, the ideal set-based definition is

$$\mathrm{JSC} = \frac{|\mathcal{C} \cap \widehat{\mathcal{C}}|}{|\mathcal{C} \cup \widehat{\mathcal{C}}|}. \tag{5.35}$$

In practice, changepoints are temporal indices and exact equality is too strict. A predicted changepoint is therefore counted as correct when it lies within a tolerance window around a true changepoint. This produces a JSC that is more appropriate for short and noisy trajectories, where an error of one or two frames may not be biologically meaningful.

The calculations reported in this part of the work are performed on short trajectories of length $T = 16$ frames. This is a deliberately challenging regime: only a small number of displacements is available to infer both local diffusion parameters and changepoint locations. It is also the regime in which classical or sequential methods can become fragile, because local estimates of α and D have high variance.

The JSC is computed separately for changepoints inferred from the predicted α_t profile and from the predicted $\log_{10}(D_t)$ profile. This separation is important because the two quantities encode different dynamical information. A change in $\log_{10}(D)$ mainly changes the scale of the displacements, whereas a change in α changes the temporal correlation structure of the trajectory. Consequently, the two tasks can have different levels of difficulty.

For this reason, the $\log_{10}(D_t)$-based analysis is the main one in the results. It is closer to the biological question of whether a molecule changes its mobility state, for example by entering a slower bound or confined state. The α_t-based analysis is still reported, but mainly as a complementary test of whether the model also detects changes in the anomalous character of the motion. This distinction avoids over-interpreting noisy α estimates from very short tracks while still preserving the information carried by anomalous diffusion [3, 4, 22, 38].

For the same trajectory length, the CNN–Transformer-based pipeline provides a clear changepoint signal in the setting considered in this work. The comparison with BI-ADD should be interpreted at fixed length $T = 16$, because trajectory length has a strong effect on changepoint detectability. It should also be read as a reference comparison rather than as a strict head-to-head benchmark, since the two methods are not evaluated under exactly identical experimental conditions. BI-ADD remains an important methodological reference for heterogeneous anomalous diffusion, while the present pipeline should be interpreted only within the short-trajectory protocol evaluated here [65].

Figures 5.4 and 5.5 report the Jaccard similarity coefficient curves used to evaluate changepoint localization from α_t and $\log_{10}(D_t)$, respectively.

Second, binned confusion matrices are used to summarize the frame-wise regression performance. Since both α and $\log_{10}(D)$ are continuous variables, the confusion matrices are not classification matrices in the strict sense. Instead, the true and predicted values are discretized into bins. The entry in row i and column j counts how often a frame whose true value falls in bin i is predicted in bin j. Row-normalized versions are especially useful because each row can be interpreted as the conditional distribution of the prediction given the true bin.

For an ideal regressor, the mass of the binned confusion matrix would be concentrated along the main diagonal. Mass above or below the diagonal indicates systematic overestimation or underestimation. For $\log_{10}(D)$, this visualization is also useful to check whether the model compresses large diffusion coefficients into a smaller predicted range. For α, it reveals whether the model can distinguish subdiffusive, Brownian and superdiffusive regimes.

Figures 5.6 and 5.7 show the binned confusion matrices for the two frame-wise regression targets.

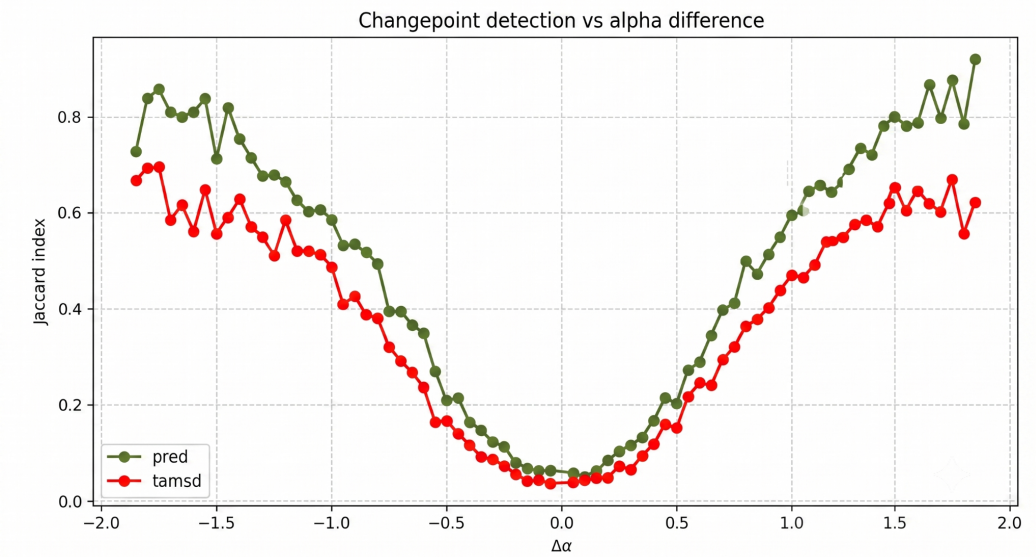


Figure 5.4: Jaccard similarity coefficient for changepoint detection based on changes in α. The horizontal axis is the jump in anomalous exponent $\Delta\alpha$, and the vertical axis is the JSC; the curves compare changepoints inferred from the neural α_t profile with a TAMSD-based reference. The low values near $\Delta\alpha = 0$ show that small changes in anomalous exponent are difficult to localize in short trajectories.

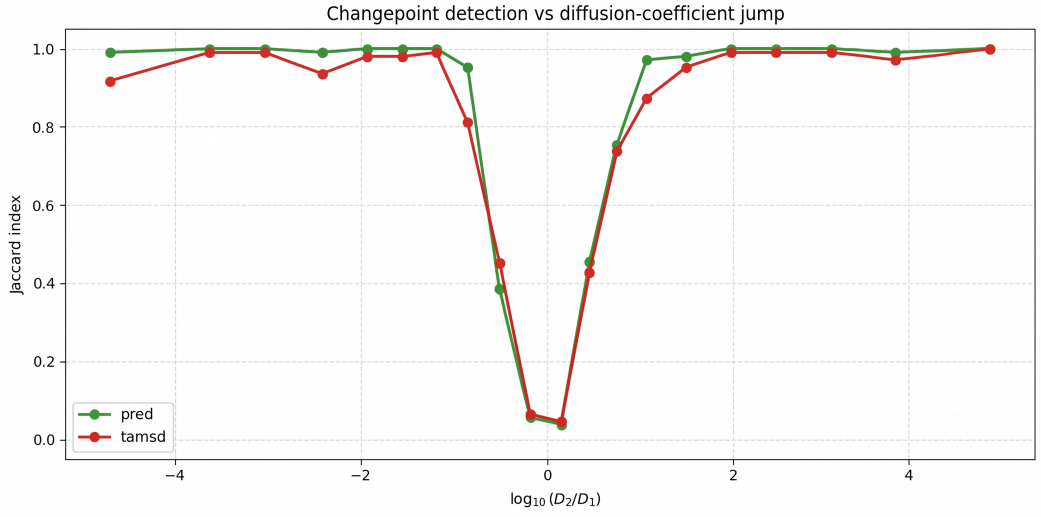


Figure 5.5: Jaccard similarity coefficient for changepoint detection based on changes in diffusion coefficient. The horizontal axis is $\log_{10}(D_2/D_1)$, and the vertical axis is the JSC; the curves compare changepoints inferred from the neural $\log_{10}(D_t)$ profile with a TAMSD-based reference. The high JSC away from zero supports the use of $\log_{10}(D_t)$ as the main mobility signal for changepoint detection.

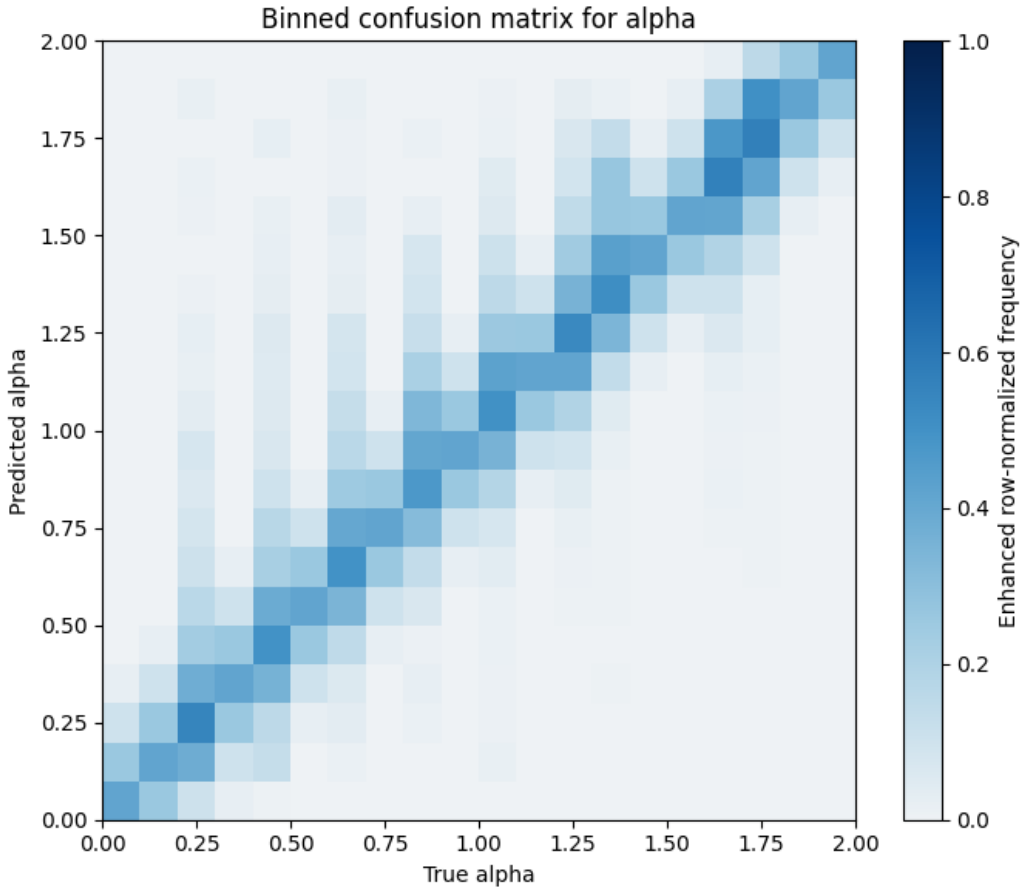


Figure 5.6: Binned confusion matrix for frame-wise α predictions. The horizontal axis is the true anomalous exponent and the vertical axis is the predicted anomalous exponent; color intensity gives the normalized frequency in each bin. Concentration near the diagonal indicates that the model preserves the scale of α, while off-diagonal spread shows residual regression uncertainty.

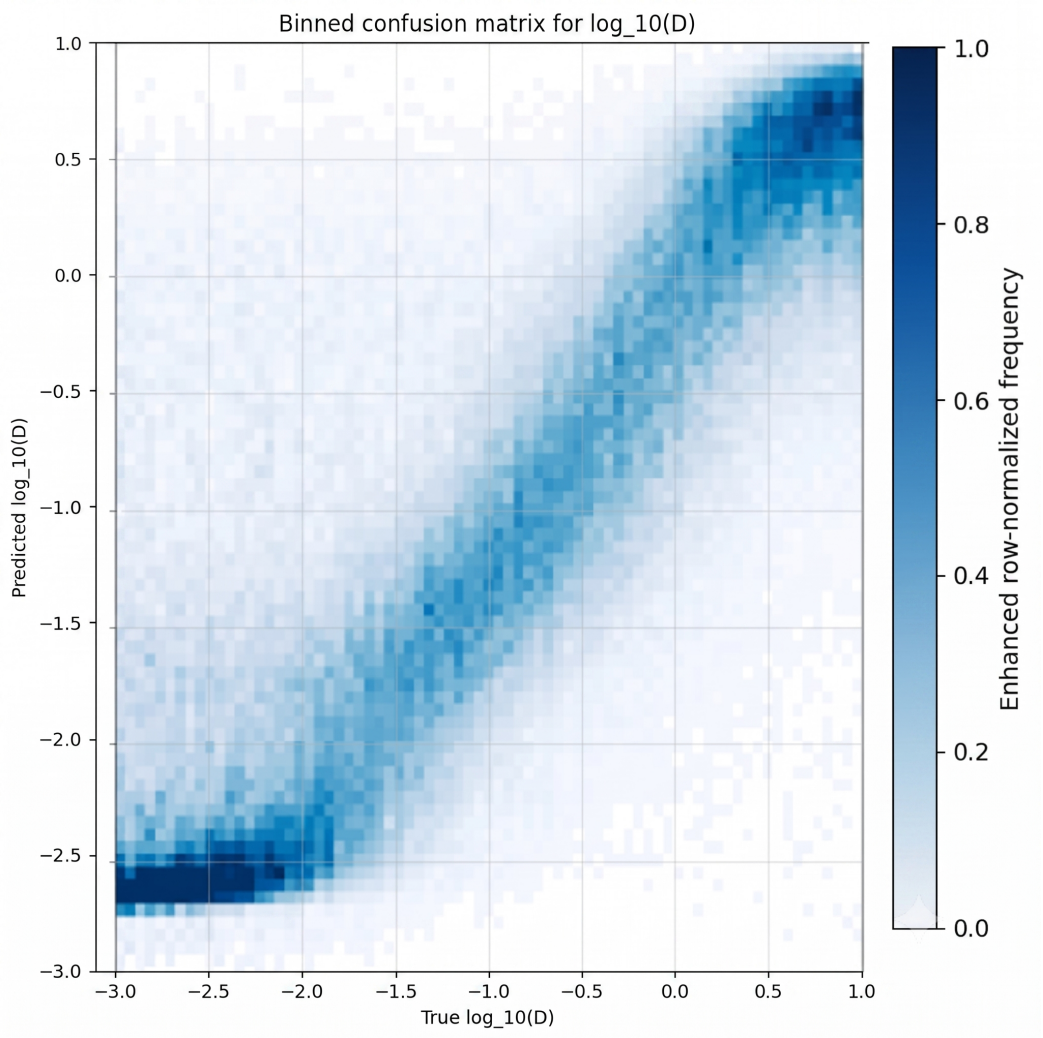


Figure 5.7: Binned confusion matrix for frame-wise $\log_{10}(D)$ predictions, with real $\log_{10}(D)$ on the horizontal axis and predicted $\log_{10}(D)$ on the vertical axis. The color scale reports enhanced row-normalized frequency. The diagonal trend indicates that the model captures the ordering of diffusion coefficients, while the spread and saturation show where the predicted dynamic range is compressed.

5.7 Quantitative Summary

The quantitative results reported in this section refer to trajectories of length $T = 16$ frames, unless explicitly stated otherwise. This choice matches the final evaluation performed here and should be kept fixed when comparing different changepoint-detection strategies. In particular, the comparison with BI-ADD is discussed at the same trajectory length, because increasing the number of frames would make the estimation of both α and $\log_{10}(D)$ easier and would change the difficulty of the evaluation.

Quantity	Value	Interpretation
MAE α	0.16 ± 0.015	Mean absolute error of the model's frame-wise anomalous-exponent predictions; uncertainty is reported as standard deviation.
MAE $\log_{10}(D)$	0.12 ± 0.018	Mean absolute error of the model's frame-wise log-diffusion predictions; uncertainty is reported as standard deviation.
JSC α	max $\simeq$ 0.85–0.95; low near $\Delta\alpha = 0$; mean $\simeq$ 0.45–0.50	Changepoint detection based on α becomes reliable only for sufficiently large jumps, while small $\Delta\alpha$ values are difficult to detect.
JSC $\log_{10}(D)$	$\simeq$ 0.95–1.00 away from zero; mean $\simeq$ 0.9; low near zero	$\log_{10}(D_t)$ is the strongest changepoint signal because changes in D directly affect displacement amplitudes.
Conformal coverage	97.60%	Empirical split-conformal coverage for the frame-wise $\log_{10}(D_t)$ analysis at nominal 95% coverage; the intervals are therefore slightly conservative.
Conformal quantile Q	1.2500	Calibration quantile used to construct the conformal band around the predicted $\log_{10}(D_t)$ profile.
Comparison with BI-ADD	Reference comparison only	BI-ADD reports maximum JSC values around 0.4 for α-driven changes and around 0.8 for K-driven or diffusion-scale changes. The present $\log_{10}(D_t)$-based curve reaches values close to one away from indistinguishable states, but this should be read as qualitative context rather than as evidence of a definitive ranking.

Table 5.3: Quantitative summary of the main results for trajectories of length $T = 16$ frames. JSC ranges are approximate and inferred from the plotted curves.

The JSC values reported in Table 5.3 should be read as summaries of the plotted curves rather than as a full matched benchmark against BI-ADD. A strict comparative ranking with BI-ADD would require running both pipelines on the same simulated trajectories, with the same changepoint tolerance and the same hyperparameter selection protocol. The safer conclusion is that the present CNN–Transformer–KernelCPD pipeline gives a promising changepoint signal in the short-trajectory setting considered here.

The JSC values are better suited to assess the comparison considered here because they directly quantify changepoint localization. On this basis, the present method gives promising results in this short-trajectory setting, without implying a strict ranking over BI-ADD across all parameter regimes.

The main conclusion is that the neural sequence-to-sequence approach extracts a useful changepoint signal even from very short trajectories. For $T = 16$, the predicted α_t and $\log_{10}(D_t)$ profiles provide enough temporal structure to support changepoint localization in the evaluated

setting. This behavior is naturally explained by the architecture: convolutional layers extract local displacement features and attention layers compare different parts of the trajectory, while the final `KernelCPD` step performs a global segmentation of the predicted profile. BI-ADD relies on segment-wise estimation and iterative merging, which may become challenging when only sixteen frames are available; therefore, the comparison is best interpreted as methodologically informative rather than as a definitive ranking across all conditions [65].

The confusion matrices complement the JSC analysis. The JSC evaluates whether the changepoint position is recovered, while the binned confusion matrices show whether the frame-wise values of α and $\log_{10}(D)$ are predicted with the correct scale. Therefore, the two types of plots should be read together: a good changepoint score without a diagonal confusion matrix would mean that the model finds transitions but does not estimate the underlying parameters accurately, whereas a good confusion matrix with poor JSC would mean that the model estimates local parameters but fails to localize transitions.

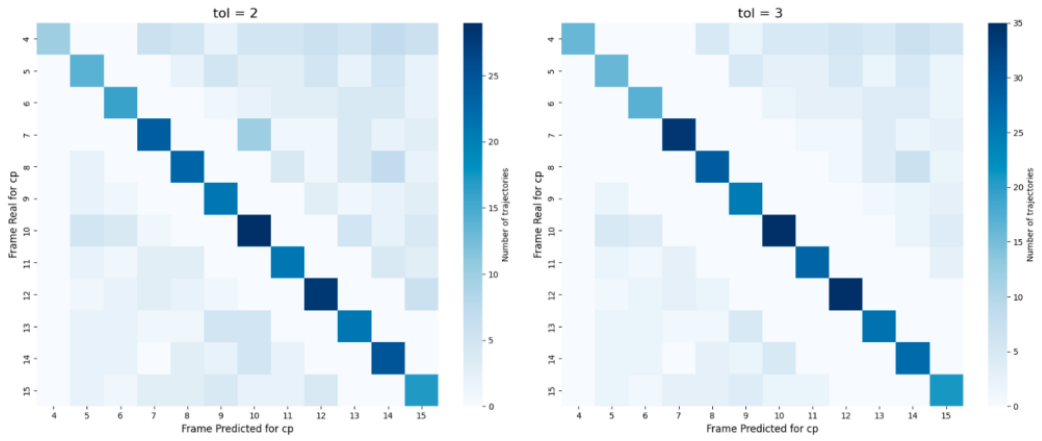


Figure 5.8: Heatmaps comparing true and predicted changepoint frames for two temporal tolerances. The horizontal axis is the predicted changepoint frame and the vertical axis is the true changepoint frame; color intensity counts how often each pair occurs. Strong diagonal structure indicates accurate temporal localization, while off-diagonal mass corresponds to systematic early or late predictions.

Figure 5.8 complements the JSC curves by showing where predicted changepoints fall relative to the true changepoint frame.

5.8 Personal Contribution

The practical contribution of this work was the implementation and evaluation of a complete analysis pipeline for short SPT trajectories. First, trajectories were preprocessed by converting absolute coordinates into displacement increments, so that the learning models operated directly on dynamical information rather than on arbitrary positions. CNN models were then trained for diffusion-parameter estimation on homogeneous trajectories, and a CNN–Transformer sequence-to-sequence model was used to obtain frame-wise profiles of α_t and $\log_{10}(D_t)$.

The predicted profiles were evaluated in several complementary ways. The frame-wise estimates of α and $\log_{10}(D)$ were summarized with binned confusion matrices, while changepoint detection was performed by applying `KernelCPD` to the predicted temporal profiles. Jaccard similarity coefficient curves were computed to compare changepoint recovery as a function of the parameter jump. Finally, split conformal prediction was applied to quantify predictive uncertainty, with particular attention to coverage and interval width.

5.9 Discussion of the Results

The work suggests three main messages. First, increments are a natural representation for neural models on SPT trajectories, because they remove absolute position and emphasize dynamics. Second, the most physically and biologically interpretable target in these short trajectories is $\log_{10}(D_t)$: it directly measures local mobility scale and therefore connects the prediction problem to mobile, slow, confined and bound-like states. Third, CNNs and hybrid CNN–Transformer architectures can learn useful information from very short trajectories, but change point detection remains sensitive to noise, trajectory length and the position of the regime change. Conformal prediction is useful to avoid presenting model outputs as certainties, but it should be evaluated together with interval width and the validity of the exchangeability assumption.

There are, however, several limitations. The first is trajectory length. The final experiments use $T = 16$ frames, which is deliberately challenging but also statistically restrictive. A change in D can be detected from changes in displacement amplitude, but a change in α requires information about temporal correlations and scaling over lag time. With only a few displacements, these correlations are weakly sampled. This limitation could be addressed by training and evaluating the same architecture across several fixed lengths, or by using architectures that can aggregate information from multiple short tracks belonging to the same biological condition.

The second limitation is the dependence on simulated data. Simulations are essential because they provide ground truth for D, α and changepoint positions, but real SPT experiments contain additional effects: localization error, motion blur, blinking, defocalization, missed detections and linking errors. If the simulation model does not reproduce these effects well, the performance measured on synthetic trajectories may overestimate the performance on experimental data. A natural improvement would be to perform a systematic robustness analysis with increasing localization noise σ_{loc}, different blinking probabilities and controlled tracking errors. A second improvement would be to validate the method on experimental controls where the expected mobility state is known, for example freely diffusing versus immobilized or chromatin-bound proteins.

The third limitation concerns the interpretation of α. The anomalous exponent is biologically useful, but it is not mechanism-specific: crowding, viscoelasticity, confinement and transient binding can all produce subdiffusive behavior. For this reason, the thesis treats D as the main signal for local mobility changes and α as a complementary descriptor. This choice is not only computational. It reflects the physical interpretation of SPT data: the displacement scale, and therefore D, is the quantity that most directly changes when a protein becomes part of a larger complex, binds to a structure, or enters a more obstructed region. A possible extension would be to train a multi-task model that predicts $\log_{10}(D_t)$, α_t and a diffusion-model class jointly. This could help separate changes in amplitude from changes in correlation structure, and would reduce the risk of interpreting a noisy α estimate as a specific biophysical mechanism.

The fourth limitation is the comparison with BI-ADD. The comparison is made at the same trajectory length, which is fair, but it does not exhaust all possible parameter choices of BI-ADD or all possible post-processing choices for the CNN–Transformer model. A stronger comparison would tune both methods on a validation set and report JSC as a function of changepoint tolerance, jump size and changepoint position. This would show in which regimes each method is more effective, and in which regimes each method fails.

Finally, conformal prediction provides marginal coverage. This is useful because it gives a distribution-free uncertainty statement, but it does not guarantee equal coverage for every subgroup of trajectories. For example, coverage may be excellent on trajectories with large jumps in D and poor on trajectories with small jumps or changepoints near the boundary. This motivates subgroup diagnostics such as

$$\mathrm{coverage}(A) = \frac{1}{|A|} \sum_{i \in A} \mathbf{1}\{y_i \in C(X_i)\}, \tag{5.36}$$

where A may be the group of trajectories with low D, with change points near the boundary, or with $\alpha < 1$. This would reveal whether average coverage hides regions where the model is less reliable. If such failures are observed, one could use stratified calibration, normalized conformal scores or separate calibration sets for different trajectory regimes.

Conclusion

Single Particle Tracking provides a direct way to study molecular motion at the level of individual trajectories. This is scientifically useful because cellular motion is heterogeneous: proteins can diffuse freely, interact transiently with partners, become confined, or switch between mobility states. Reliable inference is therefore essential. Without careful estimation of diffusion parameters, changepoints and uncertainty, short SPT trajectories can easily lead to over-interpretation of noise, localization artifacts or finite-sample fluctuations.

The methodological contribution of this work is a machine-learning pipeline for short-trajectory analysis. Trajectories are represented through displacement increments and analyzed with CNN and CNN–Transformer models. The resulting frame-wise profiles of α_t and $\log_{10}(D_t)$ are used to study local transport properties, with particular emphasis on $\log_{10}(D_t)$ as a mobility-scale signal. Changepoints are inferred by applying `KernelCPD` to these predicted profiles, while split conformal prediction provides a calibration layer for uncertainty quantification.

Several limitations remain. The analysis is based mainly on simulated data, so the gap between simulations and real SPT experiments must be addressed before strong biological conclusions can be drawn. The trajectories considered here are also very short, which makes the estimation of temporal correlations and changepoints intrinsically difficult. Future work should therefore test robustness to localization noise, motion blur, blinking and tracking errors, and validate the pipeline on experimental controls with known mobility states, such as freely diffusing, confined and immobilized proteins. Overall, the results indicate that, in the short-trajectory regime considered here, frame-wise prediction of $\log_{10}(D_t)$ combined with kernel changepoint detection is a robust strategy for detecting mobility changes, whereas α_t is better interpreted as a complementary descriptor of the transport regime.

Bibliography

[1] H. Qian, M. P. Sheetz and E. L. Elson. Single particle tracking. Analysis of diffusion and flow in two-dimensional systems. *Biophysical Journal*, 60(4), 910–921, 1991. https://doi.org/10.1016/S0006-3495(91)82125-7

[2] M. J. Saxton. Single-particle tracking: the distribution of diffusion coefficients. *Biophysical Journal*, 72(4), 1744–1753, 1997. https://doi.org/10.1016/S0006-3495(97)78820-9

[3] C. Manzo and M. F. Garcia-Parajo. A review of progress in single particle tracking: from methods to biophysical insights. *Reports on Progress in Physics*, 78, 124601, 2015. https://doi.org/10.1088/0034-4885/78/12/124601

[4] H. Shen et al. Single particle tracking: from theory to biophysical applications. *Chemical Reviews*, 117, 7331–7376, 2017. https://doi.org/10.1021/acs.chemrev.6b00815

[5] A. Kobayashi, J. Park, J. Mine-Hattab and F. Garcia Fernandez. A guide for Single Particle Tracking: from sample preparation and image acquisition to the analysis of individual trajectories. bioRxiv, 2025. https://doi.org/10.1101/2025.04.03.647105

[6] L. Schermelleh et al. Super-resolution microscopy demystified. *Nature Cell Biology*, 21, 72–84, 2019. https://doi.org/10.1038/s41556-018-0251-8

[7] E. Betzig et al. Imaging intracellular fluorescent proteins at nanometer resolution. *Science*, 313, 1642–1645, 2006. https://doi.org/10.1126/science.1127344

[8] M. J. Rust, M. Bates and X. Zhuang. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM). *Nature Methods*, 3, 793–796, 2006. https://doi.org/10.1038/nmeth929

[9] S. Manley et al. High-density mapping of single-molecule trajectories with photoactivated localization microscopy. *Nature Methods*, 5, 155–157, 2008. https://doi.org/10.1038/nmeth.1176

[10] H. C. Ishikawa-Ankerhold, R. Ankerhold and G. P. C. Drummen. Advanced fluorescence microscopy techniques: FRAP, FLIP, FLAP, FRET and FLIM. *Molecules*, 17, 4047–4132, 2012. https://doi.org/10.3390/molecules17044047

[11] H. A. Shaban, R. Barth, L. Recoules and K. Bystricky. Hi-D: nanoscale mapping of nuclear dynamics in single living cells. *Genome Biology*, 21, 95, 2020. https://doi.org/10.1186/s13059-020-02002-6

[12] A. Cook, F. Walterspiel and C. Deo. HaloTag-based reporters for fluorescence imaging and biosensing. *ChemBioChem*, 24, e202300022, 2023. https://doi.org/10.1002/cbic.202300022

[13] M. Tokunaga, N. Imamoto and K. Sakata-Sogawa. Highly inclined thin illumination enables clear single-molecule imaging in cells. *Nature Methods*, 5, 159–161, 2008. https://doi.org/10.1038/nmeth1171

[14] R. E. Thompson, D. R. Larson and W. W. Webb. Precise nanometer localization analysis for individual fluorescent probes. *Biophysical Journal*, 82(5), 2775–2783, 2002. https://doi.org/10.1016/S0006-3495(02)75618-X

[15] K. I. Mortensen, L. S. Churchman, J. A. Spudich and H. Flyvbjerg. Optimized localization analysis for single-molecule tracking and super-resolution microscopy. *Nature Methods*, 7, 377–381, 2010. https://doi.org/10.1038/nmeth.1447

[16] A. J. Berglund. Statistics of camera-based single-particle tracking. *Physical Review E*, 82, 011917, 2010. https://doi.org/10.1103/PhysRevE.82.011917

[17] T. Savin and P. S. Doyle. Static and dynamic errors in particle tracking microrheology. *Biophysical Journal*, 88(1), 623–638, 2005. https://doi.org/10.1529/biophysj.104.042457

[18] J. C. Crocker and D. G. Grier. Methods of digital video microscopy for colloidal studies. *Journal of Colloid and Interface Science*, 179(1), 298–310, 1996. https://doi.org/10.1006/jcis.1996.0217

[19] K. Jaqaman et al. Robust single-particle tracking in live-cell time-lapse sequences. *Nature Methods*, 5, 695–702, 2008. https://doi.org/10.1038/nmeth.1237

[20] N. Chenouard et al. Objective comparison of particle tracking methods. *Nature Methods*, 11, 281–289, 2014. https://doi.org/10.1038/nmeth.2808

[21] J.-Y. Tinevez et al. TrackMate: An open and extensible platform for single-particle tracking. *Methods*, 115, 80–90, 2017. https://doi.org/10.1016/j.ymeth.2016.09.016

[22] A. S. Hansen et al. Robust model-based analysis of single-particle tracking experiments with Spot-On. *eLife*, 7, e33125, 2018. https://doi.org/10.7554/eLife.33125

[23] X. Michalet. Mean square displacement analysis of single-particle trajectories with localization error: Brownian motion in an isotropic medium. *Physical Review E*, 82, 041914, 2010. https://doi.org/10.1103/PhysRevE.82.041914

[24] G. Munoz-Gil et al. AnDi: The Anomalous Diffusion Challenge. arXiv:2003.12036, 2020. https://arxiv.org/abs/2003.12036

[25] G. Munoz-Gil et al. Objective comparison of methods to decode anomalous diffusion. *Nature Communications*, 12, 6253, 2021. https://doi.org/10.1038/s41467-021-26320-w

[26] N. Granik et al. Single-particle diffusion characterization by deep learning. *Biophysical Journal*, 117(2), 185–192, 2019. https://doi.org/10.1016/j.bpj.2019.06.015

[27] H. Seckler and R. Metzler. Bayesian deep learning for error estimation in the analysis of anomalous diffusion. *Nature Communications*, 13, 6717, 2022. https://doi.org/10.1038/s41467-022-34305-6

[28] J. Kaestel-Hansen et al. Deep learning-assisted analysis of single-particle tracking for automated correlation between diffusion and function. *Nature Methods*, 2025. https://doi.org/10.1038/s41592-025-02665-8

[29] A. Einstein. On the movement of small particles suspended in stationary liquids required by the molecular-kinetic theory of heat. *Annalen der Physik*, 17, 549–560, 1905. https://doi.org/10.1002/andp.19053220806

[30] M. von Smoluchowski. On the kinetic theory of Brownian molecular motion and suspensions. *Annalen der Physik*, 21, 756–780, 1906. https://doi.org/10.1002/andp.19063261405

[31] C. W. Gardiner. *Stochastic Methods: A Handbook for the Natural and Social Sciences*. Springer, 4th edition, 2009. https://link.springer.com/book/10.1007/978-3-540-70713-4

[32] C. Texier. *Stochastic Processes*. Master 2 Physics of Complex Systems lecture notes, updated October 15, 2025.

[33] R. Metzler and J. Klafter. The random walk's guide to anomalous diffusion: a fractional dynamics approach. *Physics Reports*, 339, 1–77, 2000. https://doi.org/10.1016/S0370-1573(00)00070-3

[34] J. Klafter and I. M. Sokolov. Anomalous diffusion spreads its wings. *Physics World*, 18(8), 29–32, 2005. https://doi.org/10.1088/2058-7058/18/8/33

[35] E. Barkai, Y. Garini and R. Metzler. Strange kinetics of single molecules in living cells. *Physics Today*, 65(8), 29–35, 2012. https://doi.org/10.1063/PT.3.1677

[36] G. Guigas, C. Kalla and M. Weiss. Probing the nanoscale viscoelasticity of intracellular fluids in living cells. *Biophysical Journal*, 93(1), 316–323, 2007. https://doi.org/10.1529/biophysj.106.099267

[37] D. S. Banks and C. Fradin. Anomalous diffusion of proteins due to molecular crowding. *Biophysical Journal*, 89(5), 2960–2971, 2005. https://doi.org/10.1529/biophysj.104.051078

[38] J. A. Dix and A. S. Verkman. Crowding effects on diffusion in solutions and cells. *Annual Review of Biophysics*, 37, 247–263, 2008. https://doi.org/10.1146/annurev.biophys.37.032807.125824

[39] D. Ernst, M. Hellmann, J. Kohler and M. Weiss. Fractional Brownian motion in crowded fluids. *Soft Matter*, 8, 4886–4889, 2012. https://doi.org/10.1039/C2SM25220A

[40] M. Woringer, I. Izeddin, C. Favard and H. Berry. Anomalous subdiffusion in living cells: bridging the gap between experiments and realistic models through collaborative challenges. *Frontiers in Physics*, 8, 134, 2020. https://doi.org/10.3389/fphy.2020.00134

[41] J. R. Prindle, O. I. C. de Cuba and A. Gahlmann. Single-molecule tracking to determine the abundances and stoichiometries of freely-diffusing protein complexes in living cells: past applications and future prospects. *Journal of Chemical Physics*, 159, 071002, 2023. https://doi.org/10.1063/5.0155638

[42] A. Ghosal, Y.-H. Wang, N. Nguyen, L. Troyer and S. Kim. Practical considerations for accurate estimation of diffusion parameters from single-particle tracking in living cells. *Journal of Chemical Physics*, 163, 134202, 2025. https://doi.org/10.1063/5.0284172

[43] I. M. Sokolov. Models of anomalous diffusion in crowded environments. *Soft Matter*, 8, 9043–9052, 2012. https://doi.org/10.1039/C2SM25701G

[44] R. Metzler, J.-H. Jeon, A. G. Cherstvy and E. Barkai. Anomalous diffusion models and their properties: non-stationarity, non-ergodicity, and ageing at the centenary of single particle tracking. *Physical Chemistry Chemical Physics*, 16, 24128–24164, 2014. https://doi.org/10.1039/C4CP03465A

[45] B. B. Mandelbrot and J. W. Van Ness. Fractional Brownian motions, fractional noises and applications. *SIAM Review*, 10(4), 422–437, 1968. https://doi.org/10.1137/1010093

[46] J.-H. Jeon and R. Metzler. Fractional Brownian motion and motion governed by the fractional Langevin equation in confined geometries. *Physical Review E*, 81, 021103, 2010. https://doi.org/10.1103/PhysRevE.81.021103

[47] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324, 1998. http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

[48] I. Goodfellow, Y. Bengio and A. Courville. *Deep Learning*. MIT Press, 2016. https://www.deeplearningbook.org/

[49] P. Mehta, M. Bukov, C.-H. Wang, A. G. R. Day, C. Richardson, C. K. Fisher and D. J. Schwab. A high-bias, low-variance introduction to machine learning for physicists. *Physics Reports*, 810, 1–124, 2019. https://doi.org/10.1016/j.physrep.2019.03.001

[50] T. Mikolov, K. Chen, G. Corrado and J. Dean. Efficient estimation of word representations in vector space. arXiv:1301.3781, 2013. https://arxiv.org/abs/1301.3781

[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 1929–1958, 2014. https://jmlr.org/papers/v15/srivastava14a.html

[52] K. He, X. Zhang, S. Ren and J. Sun. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778, 2016. https://doi.org/10.1109/CVPR.2016.90

[53] J. L. Ba, J. R. Kiros and G. E. Hinton. Layer normalization. arXiv:1607.06450, 2016. https://arxiv.org/abs/1607.06450

[54] A. Vaswani et al. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. https://arxiv.org/abs/1706.03762

[55] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8), 1735–1780, 1997. https://doi.org/10.1162/neco.1997.9.8.1735

[56] X. Shi et al. Convolutional LSTM network: a machine learning approach for precipitation nowcasting. *Advances in Neural Information Processing Systems*, 2015. https://arxiv.org/abs/1506.04214

[57] C. Truong, L. Oudre and N. Vayatis. ruptures: change point detection in Python. arXiv:1801.00826, 2018. https://arxiv.org/abs/1801.00826

[58] C. Truong, L. Oudre and N. Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167, 107299, 2020. https://doi.org/10.1016/j.sigpro.2019.107299

[59] V. Vovk, A. Gammerman and G. Shafer. *Algorithmic Learning in a Random World*. Springer, 2005. https://link.springer.com/book/10.1007/b106715

[60] G. Shafer and V. Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9, 371–421, 2008. https://jmlr.org/papers/v9/shafer08a.html

[61] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani and L. Wasserman. Distribution-free predictive inference for regression. arXiv:1604.04173, 2016. https://arxiv.org/abs/1604.04173

[62] Y. Romano, E. Patterson and E. Candes. Conformalized quantile regression. *Advances in Neural Information Processing Systems*, 2019. https://arxiv.org/abs/1905.03222

[63] R. F. Barber, E. J. Candes, A. Ramdas and R. J. Tibshirani. The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*, 10(2), 455–482, 2021. https://doi.org/10.1093/imaiai/iaaa017

[64] A. N. Angelopoulos and S. Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. arXiv:2107.07511, 2021. https://arxiv.org/abs/2107.07511

[65] J. Park, N. Sokolovska, C. Cabriel, I. Izeddin and J. Mine-Hattab. Bottom-up Iterative Anomalous Diffusion Detector (BI-ADD). *Journal of Physics: Photonics*, 7(4), 045027, 2025. https://doi.org/10.1088/2515-7647/adfc19