Journal of Physics: Photonics

OPEN ACCESS

RECEIVED
17 January 2025

REVISED
2 June 2025

ACCEPTED FOR PUBLICATION
8 July 2025

PUBLISHED
17 July 2025

PAPER

Pointwise prediction of protein diffusive properties using machine learning

Rasched Haidari[1,2,*] and **Achillefs N Kapanidis**[1,2]

[1] Gene Machines Group, Clarendon Laboratory, Department of Physics, University of Oxford, Oxford, United Kingdom
[2] Kavli Institute of Nanoscience Discovery, Dorothy Crowfoot Hodgkin Building, University of Oxford, Oxford, United Kingdom
[*] Author to whom any correspondence should be addressed.

E-mail: rasched.haidari@linacre.ox.ac.uk

Keywords: anomalous diffusion, pointwise inference, diffusion, AnDi$_2$ challenge, machine learning, LSTM, changepoint analysis

Abstract

The understanding of cellular mechanisms benefits substantially from accurate determination of protein diffusive properties. Prior work in this field primarily focuses on traditional methods, such as mean square displacements, for calculation of protein diffusion coefficients and biological states. This proves difficult and error-prone for proteins undergoing heterogeneous behaviour, particularly in complex environments, limiting the exploration of new biological behaviours. The importance of determining protein diffusion coefficients, anomalous exponents, and biological behaviours led to the Anomalous Diffusion Challenge 2024, exploring machine learning methods to infer these variables in heterogeneous trajectories with time-dependent changepoints. In response to the challenge, we present M3, a machine learning method for pointwise inference of diffusive coefficients, anomalous exponents, and states along noisy heterogenous protein trajectories. M3 makes use of long short-term memory cells to achieve small mean absolute errors for the diffusion coefficient and anomalous exponent alongside high state accuracies (>90%). Subsequently, we implement changepoint detection to determine timepoints at which protein behaviour changes. M3 removes the need for expert fine-tuning required in most conventional statistical methods while being computationally inexpensive to train. The model finished in the Top 5 of the Anomalous Diffusive Challenge 2024, with small improvements made since challenge closure.

1. Introduction

Protein diffusion is vital to understanding cellular processes and the mechanisms which govern cellular functionality [1–4]. Advancements in fields such as single-molecule imaging and single-particle tracking have allowed for direct experimental observation of protein movement [4–15]. These studies have made extensive use of spatial-temporal properties of protein trajectories such as mean square displacements (MSD) and diffusion coefficients. Subsequent hard thresholding of diffusion coefficients allows for categorisation of proteins into different states, with each state representing underlying biological behaviour (e.g. a RNA polymerase molecule with a low diffusion coefficient may be interacting with the DNA) [14, 16–18].

Although successful for a small number of states with relatively simple behaviours, MSD approaches are difficult to extend to complex behaviours such as changes in environment (e.g. viscosity), confinement or directed motion [19–21]. This is further complicated with proteins transitioning between different states, leading to inaccurate diffusion coefficients and overlapping states causing misclassifications [22].

To address these issues, Muñoz–Gil *et al* introduced the Anomalous Diffusion (AnDi) Challenge in 2020 [21]. This challenge focused on advanced statistical and machine learning approaches to predict anomalous exponents and diffusion type for protein trajectories undergoing at most a single changepoint between states. The anomalous exponent is essential in understanding protein movement and is related to the diffusion coefficient by the equation [23, 24]:

$$\mathrm{MSD}\,(t) = 4Kt^{\alpha}$$

where K is the diffusion coefficient and α is the anomalous exponent. There are three distinct cases of α for which changes in diffusive behaviour are observed:

- $a < 1$ —sub-diffusion, protein trajectories display 'confined' steps; MSD $\propto t^{\alpha}$ ($\alpha = 0$ representing immobile motion)
- $a = 1$ corresponds to Brownian motion; MSD $\propto t$
- $1 < a < 2$ —super-diffusion, protein trajectories display 'directed' steps; MSD $\propto t^{\alpha}$ ($\alpha = 2$ representing ballistic motion)

The above MSD formula can be used to describe fractional Brownian motion (fBM), a generalisation of Brownian motion, in which the increments in position are no longer independent [23–28]. The covariance function for FBM is given by [24]:

$$\mathrm{Cov}\left[B_H(t)\,B_H(s)\right] = K\left(|t|^{2H} + |s|^{2H} - |t-s|^{2H}\right)$$

where B_H is the stochastic process, t and s are time points, and H is the Hurst exponent given by $\alpha = 2H$ [24]. When $\alpha \neq 1$, different timesteps of the protein trajectories are negatively or positively correlated, leading to 'confined' or 'directed' behaviour respectively.

The outcome of AnDi 2020 led to more accurate machine learning techniques as compared to traditional MSD methods [21]. Machine learning models, developed both during and outside of the challenge, have implemented various architectures such as recurrent neural networks, convolutional neural networks, graph neural networks, feed-forward networks with feature engineering, and more recently transformers [29–45].

The success of the first challenge led to AnDi 2024 [46], in which the problem focused more specifically on fBM of proteins undergoing different biological behaviours and an arbitrary number of changepoints between these states. Given noisy protein coordinates, $x(t)$ and $y(t)$, the task is to infer K, α, and the protein state (s) over the timeseries, where each variable can have an unknown number of arbitrary changepoints.

1.1. AnDi challenge 2024

AnDi 2024 considers protein trajectories simulated from five different biological behaviours as shown in figure 1(a). A single state model (SSM) does not have any changepoints, providing a control for the sensitivity of the model in the detection of changes. The rest of the models can exhibit an arbitrary number of changepoints either between two or more states as given by a transition matrix. At these changepoints, at least one of K, α or s will change. The multi-state model (MSM) considers time-dependent changes; the quenched-trap model (QTM) considers proteins in trapped (immobile) regions; the transient-confinement model considers proteins entering and exiting compartments; and lastly the dimerization model (DIM) considers co-diffusion between interacting proteins. The model can be extended to higher-order oligomeric states, which would involve scanning localisation coordinates for particles within Gaussian noise of each other. This would not require any further changes to the model (rather just data filtering) and such particles can be assigned into groups.

For QTM, $K = \alpha = s = 0$ when the protein is trapped. For the rest of the diffusive states, we have $\alpha \in [0,2]$, $K \in \left[10^{-12}, 10^{6}\right]$, and $s \in \{0,1,2,3\}$. Note there are five underlying biological behaviours but four states. These represent immobile (imm.) motion ($s = 0$ with $\alpha = 0$), confinement (conf.) ($s = 1$ with $0 < \alpha < 1.9$), free diffusion ($s = 2$ with $0 < \alpha < 1.9$), and directed (dir.) motion ($s = 3$ with $\alpha > 1.9$) [46]. The value of α or K does not necessarily have to change with s.

In response to the AnDi 2024 challenge, we present M3, a long short-term memory (LSTM) based method for the pointwise inference of K, α, s, and subsequent changepoint detection using the Python package ruptures [47, 48]. M3, as depicted in figure 1(b), extracts multivariate time series features from the original coordinates of the protein trajectory and passes these into bidirectional LSTM (biLSTMs) layers (with skip connections and dropout), followed by a feedforward layer. The use of LSTMs with skip connections allows the model to pick up on long-range dependencies while keeping the architecture simple.

2. Methods

2.1. Model architecture

As we are dealing with timeseries data, we implement a custom sequence-to-sequence model based on LSTM cells. Notably, we have three stacked biLSTMs with Rectified Linear activation functions, skip connections, and layer normalisations (see figure 1(b)). The first two biLSTM cells have two hidden layers and the last

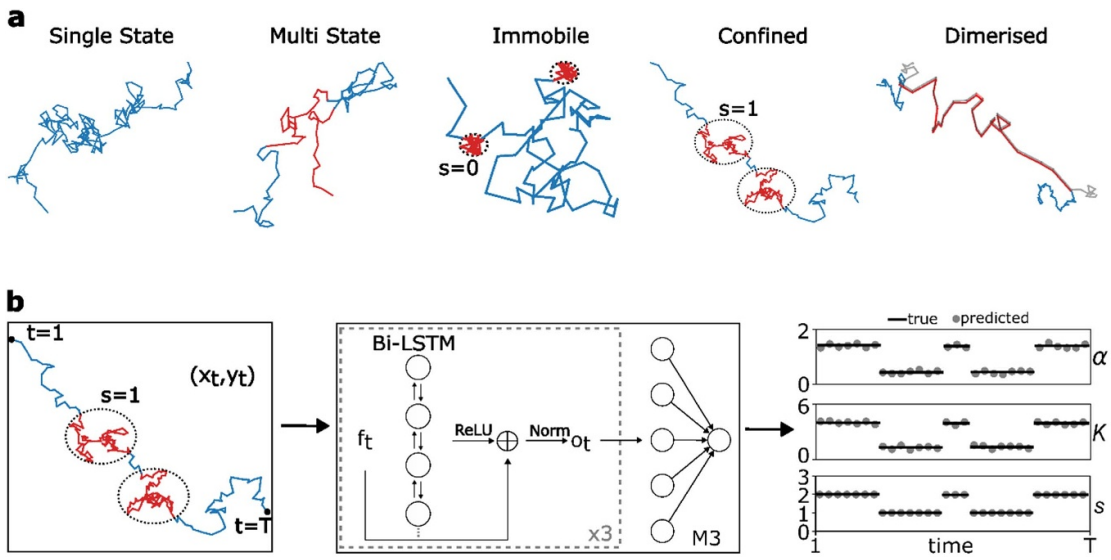

Figure 1. (a) Biological behaviours simulated using the AnDi package [49]. Red paths denote a change in behaviour. Immobile regions are assigned a state, s, equal to zero, while confined regions are assigned a state value of one. Unless specified, the state is either two or three. Figure Reproduced from [46]. CC BY 4.0. (b) M3 model architecture. The model takes in a protein trajectory and extract features f_t. This is passed through a bidirectional LSTM and we concatenate with the input using skip connections and subsequently layer normalise. This is performed three times, followed by a feed-forward layer providing pointwise predictions. The model is retrained for every variable.

biLSTM has a single layer, with all three cells returning a vector of length 128. The output of the last biLSTM is passed onto a fully connected layer, mapping the result into the desired shape. The input to the model is ten features (see *2.3 Feature Selection*), in the form of timeseries, extracted from the initial coordinates $x(t)$ and $y(t)$. The model returns a sequence of $K(t)$, $\alpha(t)$ or $s(t)$ for every timestep. For each of the three output variables, we duplicate the model and retrain, with the exception of adding a LogSoftmax layer before the output of the state model (classification task). M3 has a total of roughly 513 000 trainable parameters per model.

The model architecture resulted from starting with a single biLSTM cell and progressively incrementing the number of cells and layers until no improvement in validation loss was seen. The same procedure was carried out for other parameters in the model such as the dropout rate, number of fully connected layers, learning rate and regularisation parameter. To streamline this process, we use Optuna [50], an open source Python library implementing Bayesian hyperparameter search, and some trial and error (see *2.5 Training* for hyperparameter values and *Data Availability* for tuning process).

2.2. Training data

Training data was simulated using the AnDi Python package [21, 46, 49]. Table 1 shows the general parameters used for model training. Briefly, we simulate 400 000 trajectories each for the single state, multi-state, immobile, confined, and dimerised models. This leads to a total of two million trajectories. We repeat this process for the case where α is constrained to [1.9, 2]. The constrained range for α simulated more trajectories displaying directed motion ($s = 3$), a minority class. We further simulated trajectories (using the same sampling ranges as in table 1) in which changepoints occurred in only one of α or K while the other remained fixed across the changepoint. This is only possible for the multi-state, dimerised, and confined models, allowing isolation of α or K without interference from the other. In total we obtained 5.2 million trajectories. The trajectories are split randomly into train/test/validation datasets in a ratio of 60:20:20, such that there are 3.12 million trajectories in the training data. A few figures have been generated using different datasets, but this will be described in the text. Units for K are pixels^2/ frame$^\alpha$.

2.3. Feature selection

We carry out an extensive literature search, including past AnDi challenge literature, to obtain features specific to the problem [21, 29–46, 51–59]. We add generic timeseries features to this (e.g. running means and standard deviations) resulting in a list of roughly 80 features. Using Pearson's correlation coefficient (PCC) we remove highly collinear features ($|\text{PCC}| > 0.95$). Then, we perform greedy forward feature selection starting with the z-normalised (zero mean and unit variance) $x(t)$ and $y(t)$. By symmetry, a feature

Table 1. Parameters for simulation using the AnDi Python package [21, 46, 49].

Model	Parameters	Sampling range
Single state (SSM)	α	[0,2]
	K	$[10^{-12}, 10^{6}]$
Multi state (MSM)	Number of states (N_s)	{2, 3}
	Transition matrix	$[0, 1]^{Ns \times Ns}$
	α_i for $i = 1, \ldots, N_s$	[0, 2]
	K_i for $i = 1, \ldots, N_s$	$[10^{-12}, 10^{6}]$
Quenched trap (QTM)	Unbinding probability	[0, 0.1]
	Binding probability	1
	Trap radius	[0.5, 2]
	Number of traps	[100, 300]
	Free α, K	[0, 2], $[10^{-12}, 10^{6}]$
	Trapped α, K	0, 0
Dimerization (DIM)	Unbinding probability	[0, 0.1]
	Binding probability	1
	Interaction radius	[0.5, 5]
	Free α, K	[0, 2], $[10^{-12}, 10^{6}]$
	Dimerised α, K	[0, 2], $[10^{-12}, 10^{6}]$
Transient confinement (TCM)	Transition probability	[0, 0.3]
	Number of compartments	[30, 50]
	Compartment radius	[5, 10]
	Free α, K	[0, 2], $[10^{-12}, 10^{6}]$
	Confined α, K	[0, 2], $[10^{-12}, 10^{6}]$

not improving the model in x was removed alongside it is y counterpart (e.g. x and y running standard deviation).

After this process, we were left with ten features, each being a timeseries itself, derived from the original coordinates. These are:

- z -normalised x and y coordinates $\hat{x}(t) = \frac{x(t)-\bar{x}}{\sigma_x}$ and $\hat{y}(t) = \frac{y(t)-\bar{y}}{\sigma_y}$
- displacement from origin $d(t) = \sqrt{(x(t)-x(0))^2 + (y(t)-y(0))^2}$
- z-normalised step size $\widehat{\text{step}}(t) = \frac{\text{step}(t)-\overline{\text{step}}}{\sigma_{\text{step}}}$ with $\text{step}(t) = \sqrt{\Delta x(t)^2 + \Delta y(t)^2}$ and Δ is the difference between consecutive elements in the timeseries.
- Angle between segments of the trajectory $\vartheta(t)$
- $\text{str}(t) = \frac{\sqrt{(\hat{x}(t)-\hat{x}(0))^2+(\hat{y}(t)-\hat{y}(0))^2}}{\sum_{j=1}^{t}\sqrt{\Delta\hat{x}(j)^2+\Delta\hat{y}(j)^2}}$
- $\text{str}_2(t) = \frac{(\hat{x}(t)-\hat{x}(0))^2+(\hat{y}(t)-\hat{y}(0))^2}{\sum_{j=1}^{t}\Delta\hat{x}(j)^2+\Delta\hat{y}(j)^2}$
- $F(\hat{x})$, $F(\hat{y})$, and $F\left(\widehat{\text{step}}\right)$ where $F(k) = \log|\Delta k(t)|$.

The features $\text{str}(t)$ and $\text{str}_2(t)$ are measures of trajectory straightness, where a value of 1 represents perfectly directed motion (straight path) and a value of 0 represents a path which returned to its starting position. The final input to the model is $[\hat{x}, \hat{y}, d, \widehat{\text{step}}, \vartheta, \text{str}, \text{str}_2, F_{\hat{x}}, F_{\hat{y}}, F_{\widehat{\text{step}}}]$. Any features shorter than the total number of timesteps is post-padded with zeros. The input to the model is of shape $(B, 10, T)$ where T is the maximum trajectory length in the batch and B is the batch size (see *2.5 Training*).

2.4. Augmentations

We add the following augmentations to the coordinates $x(t)$ and $y(t)$:

- Gaussian noise with $\mu = 0$ and $\sigma = 0.1$ pixels, $y'(t) = y(t) + \varepsilon(t), x'(t) = x(t) + \varepsilon(t), \varepsilon(t) \sim \mathcal{N}(0, 0.1)$.
- Random rotations, $[x'(t), y'(t)] = M_\vartheta [x(t), y(t)]$, where M is the standard rotation matrix with angle $\vartheta \in [0, 2\pi]$).
- Vertical and horizontal flips in the $x-$ and $y-$ axes , $x'(t) = -x(t)$, $y'(t) = -y(t)$.

- Truncation of coordinates; we select two random times t_1 and t_2 such that $0 \leqslant t_1 \leqslant T - t_{\min}$ and $t_1 + t_{\min} \leqslant t_2 \leqslant T$ resulting in truncated coordinates $x(t') = x(t_1 : t_2)$ and $y(t') = y(t_1 : t_2)$ ($t_{\min} = 20$, T is trajectory length).

The augmentations are applied independently with a 30% probability for each training instance. The remaining features are calculated based on the augmented coordinates.

2.5. Training

We train the models using a batch size of 32 for a maximum of 30 epochs. During training, 10% dropout is applied to the first two biLSTM cells. Training converges typically before this and the model with the smallest validation loss is saved. A scheduler is implemented, decreasing the learning rate by a factor of ten every five epochs, if there is no improvement in validation loss. We use the Adam optimiser with an initial learning rate of 1×10^{-3} and a weight decay (λ) of 2×10^{-6} to prevent overfitting [60].

The loss functions for a single output timeseries are given by:

$$\mathcal{L}_\alpha = \frac{1}{T}\sum_{i=1}^{T} \left|\alpha_{\mathrm{p},i} - \alpha_{\mathrm{g},i}\right|$$

$$\mathcal{L}_K = \frac{1}{T}\sum_{i=1}^{T} \left|\log\left(K_{\mathrm{p},i}+1\right) - \log\left(K_{\mathrm{g},i}+1\right)\right|$$

$$\mathcal{L}_s = \frac{1}{T}\sum_{i=1}^{T} -w_{y_i}\log p\left(y_i\right)$$

where α_{p} and K_{p} are the predicted timeseries, α_{g} and K_{g} are the ground truth timeseries, and T is the total length of the protein trajectory ($T_{\max} = 200$). The state labels are imbalanced so we include class weights, w_{y_i} (inverse frequency from training dataset) in the state loss function ($\mathcal{L}_s$). The losses are normalised by trajectory length to account for varying lengths. The backpropagated loss is given by the average over the batch size (B) plus a regularisation term $\overline{\mathcal{L}} = \frac{1}{B}\sum_{i=1}^{B}\mathcal{L}_i + \lambda\sum_{j=1}^{n}\theta_j^2$. As the range of K is between 10^{-12} and 10^{6}, we work in the $\log_{10}(K+1)$ space, in which the value of the variable ranges between 0 and 6. This provides a normalised range for the model. All postprocessing steps on the predicted timeseries for K are performed in its $\log_{10}(K+1)$ space.

Training was performed on a single NVIDIA RTX A5000 GPU taking a maximum of 10 h for a single output variable. This can be done in parallel with multiple GPU instances. We had access to two GPUs, such that training took a total of 20 h.

2.6. Post-processing output

The output of the model is a timeseries of the same length as the input sequence. We smooth $K(t)$ and $\alpha(t)$ by first replacing sections in which the output series varies by 0.01 with their mean. This removes any noisy variations in the output sequence and considers jumps in values that are above this threshold. After initial smoothing, we apply a median filter with a window size of 3. Median filters preserve edges well, providing better changepoint detection. As part of the challenge constraints, the minimum time spent in a state must be at least three frames; hence, the state series is smoothed such that any state segment lasting fewer than three consecutive frames is deemed invalid and replaced with the value of its preceding state.

2.7. Changepoint detection

Changepoint detection was done using the ruptures library in Python [47]. Ruptures implements offline detection methods using a variety of cost functions. As we do not know the number of changes beforehand, an additional penalty parameter is applied. To obtain an ideal penalty value, we performed a quick grid-search over a sensible range of starting values to maximise the Jaccard index (described shortly) across the training set. Specifically, we use the Pruned Exact Linear Time (PELT) algorithm with a Least Squares Deviation (L2) cost function and a penalty of 0.3 [61]. This identifies changepoints in a timeseries by segmenting data to minimize the sum of squared deviations within segments, plus a penalty term dependent on the number of segments. We set the minimum distance of changepoints to be at least three-time steps and we min-max normalise the predicted timeseries for α and K before changepoint detection. For the state series, changepoint detection is easier as the label value changes.

Given changepoints from all three outputs, we combined them by adding changepoints from α and s to the K changepoints and removing duplicates within a window size of 5. This centres changepoints around the K changepoints, where the model generally performed better on. We cannot only take K changepoints, as K could be constant across a changepoint while α may not be, leading to missing changepoints.

2.8. Evaluation metrics

To evaluate the model on unseen testing data we calculate the mean absolute error per timestep for continuous variables, given by:

$$\frac{1}{N}\sum_{j=1}^{N}\frac{1}{T_j}\sum_{i=1}^{T_j}\left|p_{i,j}-g_{i,j}\right|$$

where N is the number of trajectories, T_j the number of timesteps for the jth trajectory, $p_{i,j}$ is the ith predicted value of α or $\log(K+1)$ for the $j$trajectory and $g_{i,j}$ is the ground truth for the corresponding predicted point. For α, this is simply the mean absolute loss (MAE), whereas for K, this is the mean absolute log error (MALE), as we work in the $\log_{10}(K+1)$ space.

For the state variable, we evaluate the confusion matrix (row-wise normalised), as the class distributions are imbalanced, and the state loss ($\mathcal{L}_s$) averaged over all trajectories. Evaluation of changepoint detection is done using two indicators discussed below: the Jaccard index and the root mean square error (RMSE).

The Jaccard index measures the level of overlap between the predicted changepoints and the ground truth changepoints with an allowable error of 5 timesteps, in-line with the challenge constraints. For two sets A and B, the Jaccard index is defined by:

$$J=\frac{A\cap B}{A\cup B}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}+\mathrm{FP}}$$

where TP, FN, and FP are the number true positives, false negatives, and false positives respectively.

The RMSE for a single trajectory is given by [46]:

$$\mathrm{RMSE}=\sqrt{\frac{1}{N_{\mathrm{CP}}}\sum_{\text{paired CPs}}\left(t_{g,i}-t_{p,j}\right)^2}$$

where N_{CP} is the number of changepoints in the trajectory, $t_{\mathrm{g},i}$ is the ground truth changepoint, and $t_{\mathrm{p},j}$ is the predicted changepoint for some pair of changepoints (i,j). The RMSE is only calculated for paired changepoints (<5 timesteps away). Matching predicted and ground truth changepoints is performed using the Hungarian algorithm, performing cost minimisation, which is handled by the AnDi package [21, 46, 49].

3. Results

Work in single molecule imaging has primarily relied on classical statistics to calculate a single diffusion coefficient for an entire trajectory [13, 24–26]. These methods are difficult to implement, rely on expert input, and encounter large difficulties in detecting complex biological phenomena. With the rise of machine learning in the single molecule field, new models allow for the inference of diffusion properties at per timestep resolutions [21, 46]. Machine learning, particularly deep learning, removes the need for expert input and finetuned manual thresholding, while being able to detect complex states.

M3, an LSTM approach to the AnDi 2024 challenge [46], returns the diffusion coefficient, anomalous exponent, and protein state at every timestep of a protein trajectory, with an unknown number of changepoints. M3 is trained on simulations from biologically relevant scenarios including free and directed diffusion, confinement, and trapped motion.

Given a trajectory, the model extracts ten features from the coordinates, resulting in a multivariate timeseries. These features were selected after an extensive literature search [21, 29–46, 51–59], performing Pearson's rank to remove correlated features, and greedy forward feature selection on the remaining features. The model is duplicated three times and trained independently for each output variable, with the exception of a LogSoftmax layer for the state variable as it is a multi-class variable.

During training, we implement augmentations; Gaussian noise, coordinate rotations, flips in the x/y axes, and truncations of the time series (see *Methods*).

Once each model is trained, we perform inference on unseen trajectories from the test data set. Inference is performed once for each model, returning $\alpha(t)$, $K(t)$, and $s(t)$ for every timestep of the trajectory. Details of the dataset and simulation are described in table 1. The output timeseries have sudden changes reflecting changepoints in the protein trajectory.

M3 finished in the Top 5 of the AnDi 2024 challenge at the time of challenge closure [62]. Since then, we have fixed a bug in the training phase and slightly altered the model architecture to include another normalisation layer.

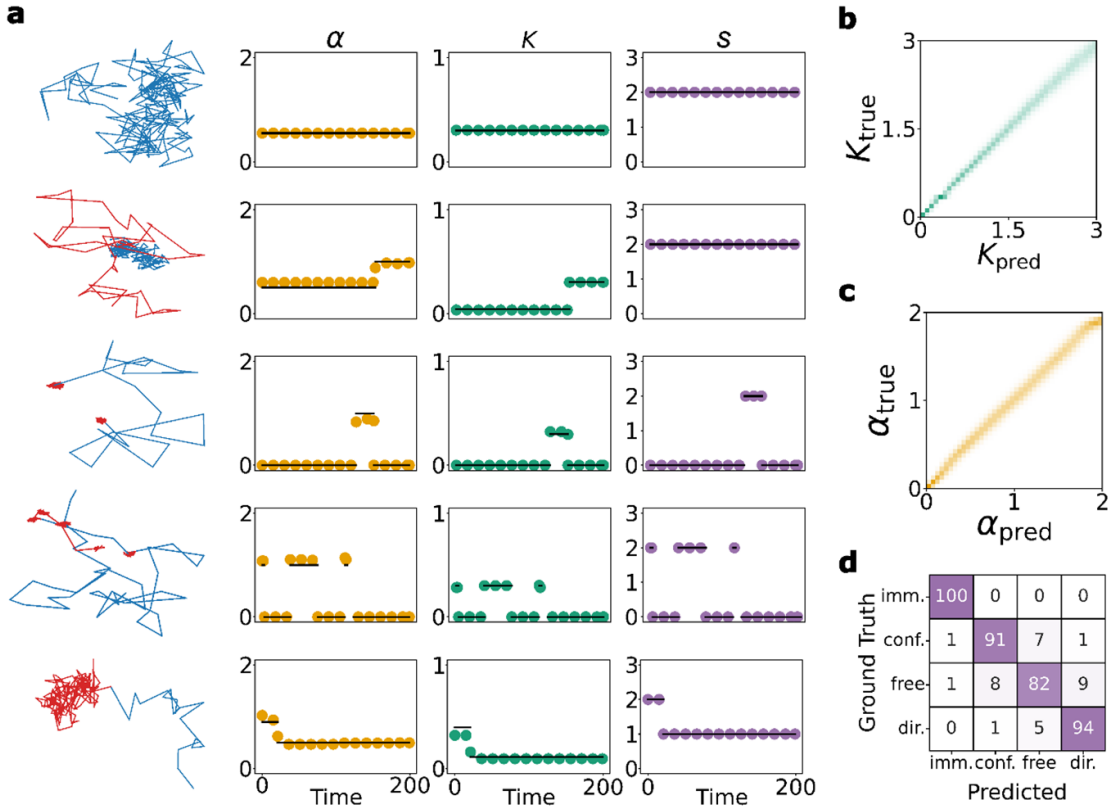

Figure 2. (a) Example model predictions on trajectories from the test set. We have (from top to bottom row), a single state, multi-state, two immobile trajectories with differing changepoint numbers, and lastly a confined trajectory. For clarity, not every timepoint prediction is shown. (b) K predictions (in log space) compared to the ground truth. Trajectories for this plot were generated using the single state model and α was fixed at one. For every K ground truth bin, 1000 protein predictions were averaged. (c) The same plot as (b) except varying α, while K was fixed at one. (d) Confusion matrix, row wise normalised, for the state variable over the entire test set.

3.1. Pointwise estimation of α, K, and s

We evaluate M3 on simulated trajectories from five underlying biological behaviours (see figure 1(a)) representing fBM with piecewise constant values of $\alpha \in [0,2]$, $K \in [10^{-12}, 10^{6}]$ $\left(\text{pixels}^2/\text{frame}^{\alpha}\right)$, and state $s \in \{0,1,2,3\}$. The protein states represent immobile (imm.) motion ($s=0$), confinement (conf., $s=1$), free diffusion ($s=2$), and directed (dir.) motion ($s=3$, $\alpha >= 1.9$). Details of the simulation can be found in *Methods* 2.2.

Once training is completed, we first visualise model performance by inspecting predicted outputs compared to ground truth variables for different trajectories from the test set. Hereof, we show all plots for K in the $\log(K+1)$ space ($K \rightarrow \log(K+1)$) such that all results are between 0 and 6. This is also evident from the scale on the plots.

Figure 2(a) shows example trajectories from different biological behaviours and the resulting model output. Generally, it was observed that predicted K values are closer to the ground truth than α, and immobile states ($s=0$) were easily detected by the model, likely due to their distinct difference from the other states. The difference in prediction of α and K can also arise from the inherent difficulty of the problem. Varying K directly affects the step size of the protein whereas α can be seen as a measure of 'directionality', making it a more difficult task for the model to learn.

To verify our observations, we first calculate the mean absolute errors as measures of accuracy for α and K averaged over the test set. This gives a MAE of 0.16 for α and a MALE of 0.14 for K. We also evaluate model predictions on trajectories simulated solely from the SSM for a wide range of α and K values. Heatmaps of predictions against ground truth are shown on figures 2(b) and (c), and allowed for model evaluation while holding the other variable fixed. At an ensemble level, the predictions from both continuous variables lie close to the ground truth with the exception that K ranges from 0 to 3 in log space. The restricted range of K reflects the fact that the protein leaves the field of view during the simulation for diffusion coefficients larger than this. However, the range includes a wide range of diffusion coefficients ($0-100\mu\text{m}^2\,\text{s}^{-1}$) beyond most experimentally observed values [63].

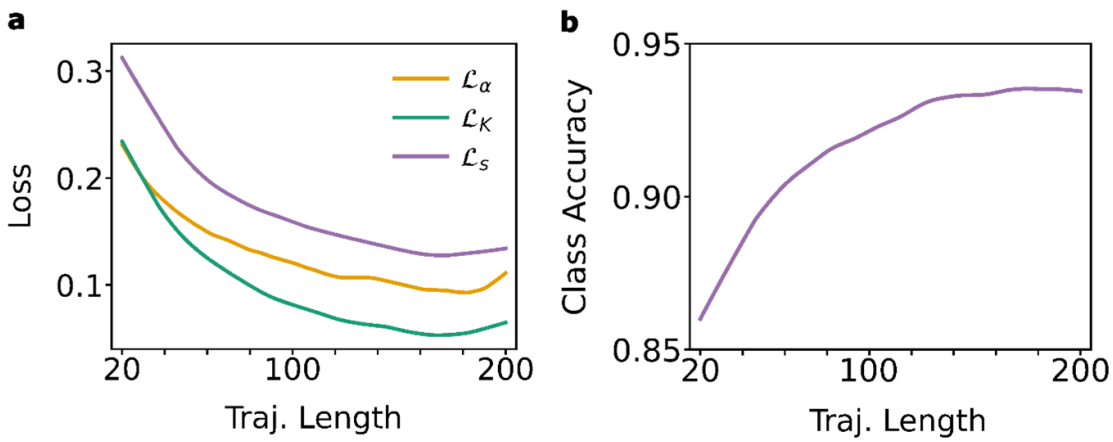

Figure 3. (a) Losses per timestep for each target variable with increasing trajectory length, averaged over the test set. (b) Average class accuracy with trajectory length over the test set.

Due to the nature of the challenge and simulations, the test set is imbalanced, with the free state ($s = 2$) observed more frequently. We implement class weights (inverse frequency) during training and evaluate the confusion matrix. This gives an average class accuracy of 92%, as shown on figure 2(d). The confusion matrix verifies the ability of the model to detect immobile states ($s = 0$) very well compared to the other states. It is interesting to note the interplay between the states $s = 1$, 2, and 3. For certain trajectories, these three states can be confused for another, not necessarily due to an inherent fault in the model setup, but due to the difficulty of the classification. As an example, a slightly larger compartment ($s = 1$) can be detected as free diffusion ($s = 2$) and vice versa. As shown later, these effects can be amplified for shorter trajectories, where the model error increases, due to less information being available to the model. Another factor affecting this may stem from the use of class weights which favours minority classes, decreasing the accuracy of the model on the majority class.

3.2. Dependence on trajectory length

We further evaluate model predictions over trajectory lengths in the range of 20 to 200. For this, we plot the model loss and average accuracy per timepoint for varying lengths across the test set. Consistent with previous work in the field, we a see a decrease in error as the trajectory length increases (see figures 3(a) and (b)). As more information is available to the model, better predictions are obtained. Shorter trajectories can originate from a range of possible parameters, leading to higher errors in α, K, and s. Figures 3(a) and (b) show plateaus in both loss and class accuracy at lengths of roughly 150 time steps and greater.

3.3. Changepoint detection

M3 does not directly return changepoints. Detecting changepoints in the output timeseries is performed using ruptures, an open-source changepoint detection library in Python, through a cost function and penalty parameter [47] (selected from a grid search across the training set, see *Methods 2.7*). For the state timeseries, finding changepoints is easier as the value of the label changes. Figure 4(a) shows an example model prediction for K and the detected changepoints. The model is able to pick up all changes in the protein trajectory for sufficiently long enough timeseries. The penalty parameter in ruptures is tuneable. To avoid constant tuning, all timeseries are min-max normalised before changepoint detection, resulting in the same penalty for both K and α.

As shown on figure 4(b), with an increasing number of changepoints (N_{CP}) in the protein trajectory, the MAE for α increases while the MALE for K remains roughly constant. This aligns with the asymptotic behaviour of α, which leads to greater prediction estimates for a higher number of changepoints, in which the average segment length is shorter (inversely related) [21]. K can be estimated with lower error from shorter segments (with less information per segment) and thus we see a constant loss as the number of changepoints increases.

Changepoint detection can be performed on the α, K, and s predictions. We first perform this individually, then combine sets of changepoints into a single set with repeats removed. This merges the changepoints from all timeseries and ensures the model is able to pick up on changepoints from any of the output variables. In figure 4(c), we evaluate the RMSE for changepoint detection using combinations of α, K, and s. As expected, the α timeseries alone returns the highest RMSE across the range of changepoints, with changepoints from K predictions providing the lowest RMSE values. The state timeseries alone is not enough

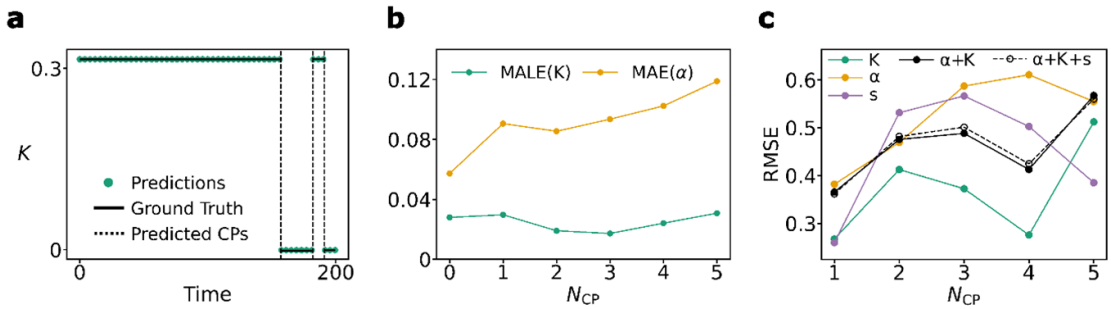

Figure 4. (a) Example application of changepoint detection, using the PELT algorithm [47], on the model prediction for K (b) The MAE and MALE over the test set for varying number of changepoints (N_{CP}) (c) RMSE for changepoint detection performed using different combinations of α, K and s over the test set. Every N_{CP} represents RMSE values for 100 randomly sampled protein trajectories of length 200 from the test set.

to detect changepoints, since in multi-state trajectories, the state label remains the same although a changepoint can occur. However, using only K timeseries for changepoints may mean we lose out on changepoints in which K remains constant across the changepoint whereas α does not. Thus, we settle for using $\alpha + K$ changepoints (see figure 4(c)), while removing duplicates, for producing a set of changepoints for any given trajectory (providing slightly lower RMSE values than $\alpha + K + s$). This may mean we miss on changepoints in which both of these variables do not change, while s does, but these trajectories are less likely ($\sim$0.04% of test dataset). Typically, a change in biological behaviour is accompanied by a change in diffusion coefficient or anomalous exponent.

We also evaluate the average Jaccard index on figure 5(a) for varying changepoints using the same sets of variables. Again, we see slightly better performance across the test set using $\alpha + K$ compared to $\alpha + K + s$. Figure 5(b) shows changepoint detection for α, K, and s for trajectories with a single changepoint across the test set. The model's bidirectional nature allows for consistent changepoint detection across various changepoint positions in trajectories.

Changepoint detection is clearly dependent on the difference of α and K at a changepoint. Little to no difference in α or K at a transition in a protein trajectory leads to no changepoint being detected using the PELT algorithm. For small differences in diffusive properties at a changepoint, the model may also return a single value at all timesteps for the entire trajectory, as it is unable to detect a change.

To evaluate performance as a function of the difference at a changepoint, we simulated two new datasets in which there exists a single changepoint exactly halfway through the trajectory (MSm), with all trajectories of length 200. The first dataset fixes α at one for all trajectories, allowing K to change between the two segments, while the second dataset fixes K at one for all trajectories and allows α to vary between the two segments. These datasets allow for robust evaluation of changepoint detection for varying $\Delta\alpha$ and ΔK between the segments. We show the Jaccard index for $\Delta\alpha$ between segments on figure 5(c). The model performs well for large differences in $\Delta\alpha$ between the segments, with the Jaccard index drastically reducing for smaller differences. As expected, we obtain a Jaccard value of zero when the two segments have identical α values, as the model did not pick up any changepoints. Figure 5(d) repeats the same analysis for ΔK. This shows high values of Jaccard index for a much larger range of values, again reducing for ΔK close to zero. As mentioned before, we believe the increased performance in ΔK is due to a change in K directly affecting the trajectory step sizes. Trajectories with fixed K and increasingly closer values of α are much harder to distinguish as the model needs to spot a measure of 'directionality' over stochastic consecutive time steps.

3.4. Dependence on biological behaviours

Lastly, we evaluate the Jaccard index using the combined changepoints from $\alpha + K$ timeseries against simulated biological states. As figure 6(a) highlights, the Jaccard index decreases as the number of changepoints increases for most biological states. The immobile and free states provide high Jaccard values over a range of changepoints. We see decreased detection of changepoints using the $\alpha + K$ set for the confined and directed models (any trajectories containing $s = 1$ and $s = 3$ respectively), particularly for a greater number of changepoints. This aligns with figure 6(b), which shows significantly higher values of MAE and MALE for the confined and directed models. The larger error in α and K for these models directly affects subsequent changepoint detection, so we expect to see decreases Jaccard indices. The higher error in the confined model may arise from large values of K we simulated. As the protein reflects at a compartment

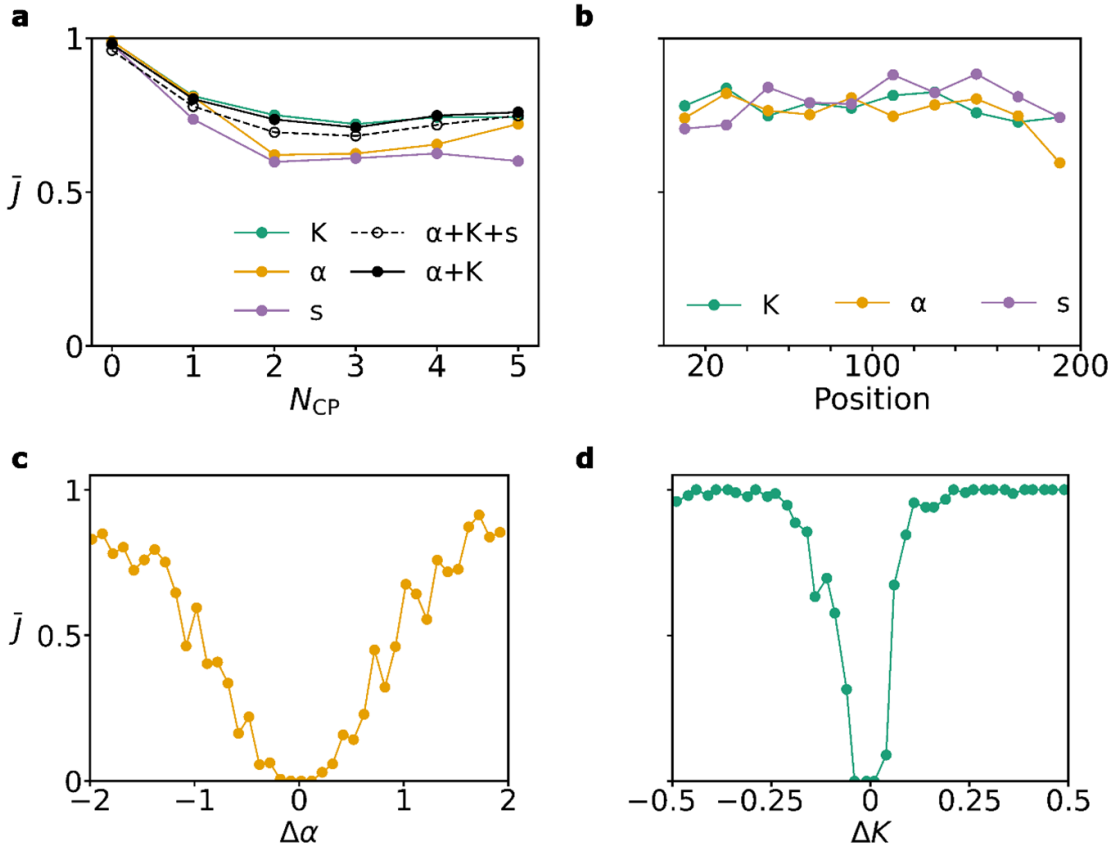

Figure 5. (a) Average Jaccard index, $\bar{J}$, using combinations of variables for changepoint detection. Each point represents an average over 100 randomly selected protein trajectories of length 200 from the test set. (b) Average Jaccard index for trajectories of length 200 from the test set with a single changepoint. Each point represents the average over 100 randomly selected trajectories and bins a range of 20 lengths. (c) Average Jaccard index for trajectories with a single changepoint (thus having two segments, multi-state model), with $\Delta\alpha$ being the difference in α between the two segments. K is fixed at one for both segments. Each point is the average of roughly 50 trajectories of length 200. (d) The same plot as (c), except for ΔK, with α fixed at one between segments.

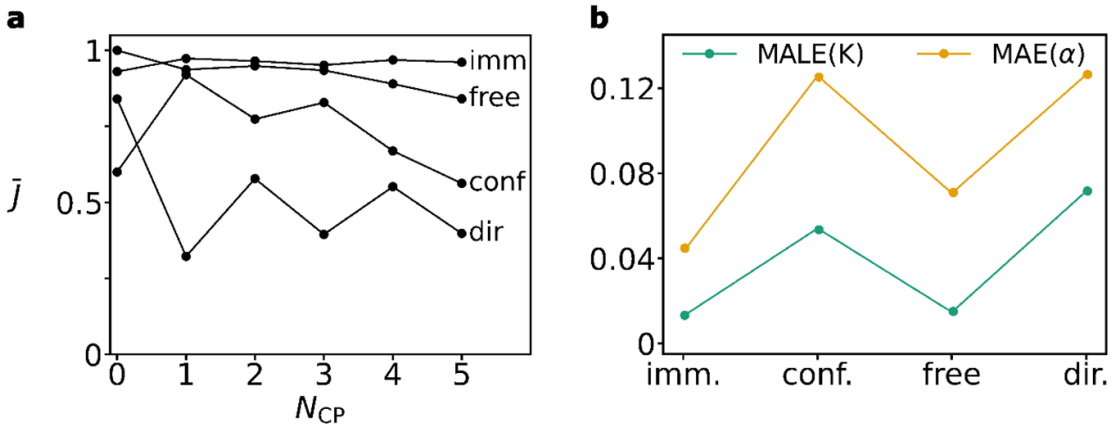

Figure 6. (a) Average Jaccard index for changepoint detection for various biological states and number of changepoints N_{CP}. Each point represents the average over 100 randomly selected protein trajectories of length 200. (b) MAE(α) and MALE(K) given the ground truth biological state over the entire test set.

boundary, consecutive timesteps can appear close for both high and small values of K. This may impact the model negatively, as it sees higher and lower values of K during training for similarly looking coordinates. A similar issue may be present for directed trajectories, where high values of K can lead to large displacements, potentially creating trajectories similar to lower K values and higher values of α.

4. Discussion

In this work we have presented M3, a machine learning model with three stacked biLSTMs layers for inferring diffusion coefficients, anomalous exponents, and states, for every frame of a protein trajectory. Specifically, we show M3 performs well on noisy stochastic trajectories simulated from varying biologically relevant scenarios.

M3 provides a new and simple method for detecting complex motion of proteins in cells. Additionally, the model does not require prior expert knowledge or manual finetuning of thresholds, as done with classical statistics or methods relying on MSD. In the AnDi Challenge, at the time of challenge closure, the model placed in the Top 5 on a set of combined metrics including absolute errors, Jaccard indices, and RMSE of predicted changepoints [62]. Since then, improvements have been made to the model, predominantly in the model architecture (addition of another layer normalisation) and the correction of a bug in the training data generation. The model consists of roughly half a million trainable parameters per variable, making it possible to train on a single NVIDIA GPU in 10 h for a single output variable.

It is possible to combine the models into a single model with three output heads, but this is difficult as it combines classification and regression. This can serve as future work, as a more complex loss function with additional parameters would be required.

Applying M3 to experimental single-molecule data requires consideration of several caveats commonly encountered in single molecule imaging. Fluorophore blinking may lead to intermittent signal loss. This can be mitigated through longer exposure times which increase photon collection and reduce blinking effects although at the cost of temporal resolution. The use of more photostable fluorophores will reduce both blinking and photobleaching enabling longer trajectory lengths. Excitation laser intensities in single molecule experiments can also be tuned to access longer timescales, which would be beneficial when wanting to observe various diffusive behaviours and allows for best model performance. Lower laser intensities also reduce potential photo-induced damage to living cells.

In cases where missing frames arise due to blinking, we can interpolate coordinates from surrounding frames. As part of this process accurate localisation is critical. Out-of-focus molecules, reduced signal-to-noise, and motion blur can introduce localisation errors leading to errors in downstream diffusion parameters. A potential approach to tackling localisation errors would be to resample trajectory coordinates by adding Gaussian noise based on localisation uncertainty. Subsequently, inference on each trajectory can be used to plot a probability distribution over possible output values. The output distribution can be used to infer both a final value and error range (similar to MC dropout techniques). Such an approach will be useful in cases where localisation cannot be performed accurately and provides pointwise errors.

Lastly, specific cellular boundaries will constrain diffusion. As such, it may be better to finetune the model using simulated tracks that account for cell geometry. Finetuning ML models on experimental data will be harder and subjective as compared to simulated data due to small dataset sizes and the lack of a ground truth.

As a starting point for experimental validation, future work can involve the use of M3, without any further finetuning, to analyse binding kinetics. M3 performs incredibly well (see figure 2(d)) in detecting immobile states which can be used to calculate dwell times, crucial for understanding cellular mechanisms and gene expression *in vivo* [16–18].

Improvements can be also made to the model architecture, such as the addition of convolutional layers, which would require further training and hyperparameter optimisation. Convolutional layers may be less flexible with varying length timeseries but can be incorporated in a unified (multi-headed) architecture. The model can also be entirely replaced by transformers, which have shown to outperform LSTMs. Transformers alongside hybrid LSTM-CNN models were tested. All architectures were kept small (minimising free parameters) for compatibility with single-GPU training within a day, but none outperformed the evaluation metrics as given by the current M3 model. However, experimentation was brief and should be further explored.

Future changes to the model architecture should potentially focus on short trajectories (<150 timesteps) as this is where the model performs worse, with a plateau in accuracy for longer trajectories (>150 timesteps). For current applications to experimental single molecule data, longer trajectories are preferable, which may require running experiments using low laser intensities and photostable fluorophores.

The ability of machine learning methods to achieve better performance, as shown by the AnDi challenges, while also providing pointwise diffusive properties, highlights the need for a shift in analysis of experimental data [21, 46]. New techniques, such as M3, provide fine grained insights into diffusive behaviours, allowing for better understanding of cellular mechanisms. Combined with advancements in experimental data and techniques, this can lead to new discoveries in protein behaviour and mechanisms at cellular levels.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://github.com/raschedh/AnomalousDiffusion.

Acknowledgments

The authors would like to thank Dr Piers Turner for insightful discussions throughout the challenge timeline and valuable comments on the manuscript. R H would like to thank the Medical Sciences Doctoral Training Centre University of Oxford for funding. A N K was supported by Wellcome Trust Grant 226662/Z/22/Z and UK Biotechnology and Biological Sciences Research Council Grant BB/X015637/1.

Conflict of interest

A N K is a cofounder and shareholder of Oxford Nanoimaging (ONI), a company that builds, sells, and supports miniaturized fluorescence microscopes for single-molecule imaging, including for single-particle tracking.

References

[1] Elf J, Li G-W and Xie X S 2007 Probing transcription factor dynamics at the single-molecule level in a living cell *Science* **316** 1191–4
[2] Gebhardt J C M, Suter D M, Roy R, Zhao Z W, Chapman A R, Basu S, Maniatis T and Xie X S 2013 Single-molecule imaging of transcription factor binding to DNA in live mammalian cells *Nat. Methods* **10** 421–6
[3] Liu Z, Lavis L D and Betzig E 2015 Imaging live-cell dynamics and structure at the single-molecule level *Mol. Cell* **58** 644–59
[4] Betzig E, Patterson G H, Sougrat R, Lindwasser O W, Olenych S, Bonifacino J S, Davidson M W, Lippincott-Schwartz J and Hess H F 2006 Imaging intracellular fluorescent proteins at nanometer resolution *Science* **313** 1642–5
[5] Balzarotti F, Eilers Y, Gwosch K C, Gynnå H A, Westphal V, Stefani F D, Elf J and Hell S W 2017 Nanometer resolution imaging and tracking of fluorescent molecules with minimal photon fluxes *Science* **355** 606–12
[6] Lelek M, Gyparaki M T, Beliu G, Schueder F, Griffié J, Manley S, Jungmann R, Sauer M, Lakadamyali M and Zimmer C 2021 Single-molecule localization microscopy *Nat. Rev. Meth. Primers* **1** 1–27
[7] Heilemann M, van de Linde S, Schüttpelz M, Kasper R, Seefeldt B, Mukherjee A, Tinnefeld P and Sauer M 2008 Subdiffraction-resolution fluorescence imaging with conventional fluorescent probes *Angew. Chem., Int. Ed.* **47** 6172–6
[8] Sharonov A and Hochstrasser R M 2006 Wide-field subdiffraction imaging by accumulated binding of diffusing probes *Proc. Natl Acad. Sci.* **103** 18911–6
[9] Hess S T, Girirajan T P K and Mason M D 2006 Ultra-high resolution imaging by fluorescence photoactivation localization microscopy *Biophys. J.* **91** 4258–72
[10] Rust M J, Bates M and Zhuang X 2006 Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM) *Nat. Methods* **3** 793–6
[11] Manley S, Gillette J M, Patterson G H, Shroff H, Hess H F, Betzig E and Lippincott-Schwartz J 2008 High-density mapping of single-molecule trajectories with photoactivated localization microscopy *Nat. Methods* **5** 155–7
[12] Stracy M, Jaciuk M, Uphoff S, Kapanidis A N, Nowotny M, Sherratt D J and Zawadzki P 2016 Single-molecule imaging of UvrA and UvrB recruitment to DNA lesions in living *Escherichia coli Nat. Commun.* **7** 12568
[13] Endesfelder U, Finan K, Holden S J, Cook P R, Kapanidis A N and Heilemann M 2013 Multiscale spatial organization of RNA polymerase in *Escherichia coli Biophys. J.* **105** 172–81
[14] Kapanidis A N, Uphoff S and Stracy M 2018 Understanding protein mobility in bacteria by tracking single molecules *J. Mol. Biol.* **430** 4443–55
[15] Fan J, El Sayyed H, Pambos O J, Stracy M, Kyropoulos J and Kapanidis A N 2023 RNA polymerase redistribution supports growth in E. coli strains with a minimal number of rRNA operons *Nucleic Acids Res.* **51** 8085–101
[16] Stracy M, Uphoff S, Garza de Leon F and Kapanidis A N 2014 *In vivo* single-molecule imaging of bacterial DNA replication, transcription, and repair *FEBS Lett.* **588** 3585–94
[17] Sayyed H E, Pambos O J, Stracy M, Gottesman M E and Kapanidis A N 2024 Single-molecule tracking reveals the functional allocation, *in vivo* interactions, and spatial organization of universal transcription factor NusG *Mol. Cell* **84** 926–937.e4
[18] Stracy M, Lesterlin C, Garza de Leon F, Uphoff S, Zawadzki P and Kapanidis A N 2015 Live-cell superresolution microscopy reveals the organization of RNA polymerase in the bacterial nucleoid. *Proc. Natl Acad. Sci.* **112** E4390–9
[19] Konopka M C, Shkel I A, Cayley S, Record M T and Weisshaar J C 2006 Crowding and confinement effects on protein diffusion *in vivo J. Bacteriol.* **188** 6115–23
[20] Chenouard N *et al* 2014 Objective comparison of particle tracking methods *Nat. Methods* **11** 281–9
[21] Muñoz-Gil G *et al* 2021 Objective comparison of methods to decode anomalous diffusion *Nat. Commun.* **12** 6253
[22] Endesfelder U and Heilemann M 2014 Art and artifacts in single-molecule localization microscopy: beyond attractive images *Nat. Methods* **11** 235–8
[23] Metzler R, Jeon J-H, Cherstvy A G and Barkai E 2014 Anomalous diffusion models and their properties: non-stationarity, non-ergodicity, and ageing at the centenary of single particle tracking *Phys. Chem. Chem. Phys.* **16** 24128–64
[24] Mandelbrot B B and Van Ness J W 1968 Fractional Brownian motions, fractional noises and applications *SIAM Rev.* **10** 422–37
[25] Metzler R and Klafter J 2000 The random walk's guide to anomalous diffusion: a fractional dynamics approach *Phys. Rep.* **339** 1–77
[26] Jeon J-H, Tejedor V, Burov S, Barkai E, Selhuber-Unkel C, Berg-Sørensen K, Oddershede L and Metzler R 2011 *In vivo* anomalous diffusion and weak ergodicity breaking of lipid granules *Phys. Rev. Lett.* **106** 048103
[27] Barkai E, Garini Y and Metzler R 2012 Strange kinetics of single molecules in living cells *Phys. Today* **65** 29–35
[28] Höfling F and Franosch T 2013 Anomalous transport in the crowded world of biological cells *Rep. Prog. Phys.* **76** 046602

[29] Gentili A and Volpe G 2021 Characterization of anomalous diffusion classical statistics powered by deep learning (CONDOR) *J. Phys. A* **54** 314003
[30] Argun A, Volpe G and Bo S 2021 Classification, inference and segmentation of anomalous diffusion with recurrent neural networks *J. Phys. A* **54** 294003
[31] Garibo-i-Orts O, Baeza-Bosca A, Garcia-March M A and Conejero J A 2021 Efficient recurrent neural network methods for anomalously diffusing single particle short and noisy trajectories *J. Phys. A: Math. Theor* **54** 504002
[32] Arts M, Smal I, Paul M W, Wyman C and Meijering E 2019 Particle mobility analysis using deep learning and the moment scaling spectrum *Sci. Rep.* **9** 17160
[33] Manzo C 2021 Extreme learning machine for the characterization of anomalous diffusion from single trajectories (AnDi-ELM) *J. Phys. A* **54** 334002
[34] Li D, Yao Q and Huang Z 2021 WaveNet-based deep neural networks for the characterization of anomalous diffusion (WADNet) *J. Phys. A* **54** 404003
[35] Firbas N, Garibo-i-orts Ò, Garcia-March M Á and Conejero J A 2023 Characterization of anomalous diffusion through convolutional transformers *J. Phys. A* **56** 014001
[36] Verdier H, Duval M, Laurent F, Cassé A, Vestergaard C L and Masson J-B 2021 Learning physical properties of anomalous random walks using graph neural networks *J. Phys. A* **54** 234001
[37] Muñoz-Gil G, Garcia-March M A, Manzo C, Martín-Guerrero J D and Lewenstein M 2020 Single trajectory characterization via machine learning *New J. Phys.* **22** 013010
[38] Granik N, Weiss L E, Nehme E, Levin M, Chein M, Perlson E, Roichman Y and Shechtman Y 2019 Single-particle diffusion characterization by deep learning *Biophys. J.* **117** 185–92
[39] Kowalek P, Loch-Olszewska H and Szwabiński J 2019 Classification of diffusion modes in single-particle tracking data: feature-based versus deep-learning approach *Phys. Rev. E* **100** 032410
[40] Loch-Olszewska H and Szwabiński J 2020 Impact of feature choice on machine learning classification of fractional anomalous diffusion *Entropy* **22** 1436
[41] Bo S, Schmidt F, Eichhorn R and Volpe G 2019 Measurement of anomalous diffusion using recurrent neural networks *Phys. Rev. E* **100** 010102
[42] Requena B, Masó-Orriols S, Bertran J, Lewenstein M, Manzo C and Muñoz-Gil G 2023 Inferring pointwise diffusion properties of single trajectories with deep learning *Biophys. J.* **122** 4360–9
[43] Chen Z, Geffroy L and Biteen J S 2021 NOBIAS: analyzing anomalous diffusion in single-molecule tracks with nonparametric bayesian inference *Front. Bioinform.* **1** 742073
[44] Qu X, Hu Y, Cai W, Xu Y, Ke H, Zhu G and Huang Z 2024 Semantic segmentation of anomalous diffusion using deep convolutional networks *Phys. Rev. Res.* **6** 013054
[45] Kabbech H and Smal I 2022 Identification of diffusive states in tracking applications using unsupervised deep learning methods *2022 IEEE 19th Int. Symp. on Biomedical Imaging (ISBI)* pp 1–4
[46] Muñoz-Gil G, Bachimanchi H, Pineda J, Midtvedt B, Lewenstein M, Metzler R, Krapf D, Volpe G and Manzo C 2024 Quantitative evaluation of methods to analyze motion changes in single-particle experiments *Nature Communication Registered Report Stage 1 Protocol*
[47] Truong C, Oudre L and Vayatis N 2020 Selective review of offline change point detection methods *Signal Process.* **167** 107299
[48] Hochreiter S and Schmidhuber J 1997 Long short-term memory *Neural Comput.* **9** 1735–80
[49] Muñoz-Gil G, Requena B, Volpe G, Garcia-March M A and Manzo C 2021 AnDiChallenge/ANDI_datasets: Challenge 2020 release (https://doi.org/10.5281/zenodo.4775311)
[50] Akiba T, Sano S, Yanase T, Ohta T and Koyama M 2019 Optuna: a next-generation hyperparameter optimization framework *Proc. 25th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining KDD'19* (*New York, NY, USA*) (Association for Computing Machinery) pp 2623–31
[51] Wagner T, Kroll A, Haramagatti C R, Lipinski H-G and Wiemann M 2017 Classification and segmentation of nanoparticle diffusion trajectories in cellular micro environments *PLoS One* **12** e0170165
[52] Matsuoka S, Shibata T and Ueda M 2009 Statistical analysis of lateral diffusion and multistate kinetics in single-molecule imaging *Biophys. J.* **97** 1115–24
[53] Renner M, Wang L, Levi S, Hennekinne L and Triller A 2017 A simple and powerful analysis of lateral subdiffusion using single particle tracking *Biophys. J.* **113** 2452–63
[54] Metzner C, Mark C, Steinwachs J, Lautscham L, Stadler F and Fabry B 2015 Superstatistical analysis and modelling of heterogeneous random walks *Nat. Commun.* **6** 7516
[55] Lanoiselée Y and Grebenkov D S 2017 Unraveling intermittent features in single-particle trajectories by a local convex hull method *Phys. Rev. E* **96** 022144
[56] Vega A R, Freeman S A, Grinstein S and Jaqaman K 2018 Multistep track segmentation and motion classification for transient mobility analysis *Biophys. J.* **114** 1018–25
[57] Dosset P, Rassam P, Fernandez L, Espenel C, Rubinstein E, Margeat E and Milhiet P-E 2016 Automatic detection of diffusion modes within biological membranes using back-propagation neural network *BMC Bioinform.* **17** 197
[58] Yin S, Song N and Yang H 2018 Detection of velocity and diffusion coefficient change points in single-particle trajectories *Biophys. J.* **115** 217–29
[59] Helmuth J A, Burckhardt C J, Koumoutsakos P, Greber U F and Sbalzarini I F 2007 A novel supervised trajectory segmentation algorithm identifies distinct types of human adenovirus motion in host cells *J. Struct. Biol.* **159** 347–58
[60] Kingma D P and Ba J 2017 Adam: a method for stochastic optimization *3rd Int. Conf. for Learning Representations* (https://doi.org/10.48550/arXiv.1412.6980)
[61] Killick R, Fearnhead P and Eckley I A 2012 Optimal detection of changepoints with a linear computational cost *J. Am. Stat. Assoc.* **107** 1590–8
[62] AnDi Challenge 2024 (available at: http://andi-challenge.org/challenge-2024/) (Accessed May 2025)
[63] Stracy M, Schweizer J, Sherratt D J, Kapanidis A N, Uphoff S and Lesterlin C 2021 Transient non-specific DNA binding dominates the target search of bacterial DNA-binding proteins *Mol. Cell* **81** 1499–1514.e6