THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF BIOMEDICAL ENGINEERING

SINGLE-MOLECULE MICROSCOPY STEP DETECTION ALGORITHMS: KINESIN MOTOR PROTEINS AND THE CELLULOSE SYNTHESIS COMPLEX

NATHAN DEFFENBAUGH
SPRING 2015

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Bioengineering
with honors in Bioengineering

Reviewed and approved* by the following:

William O. Hancock
Professor of Biomedical Engineering
Thesis Supervisor, Honors Adviser

Peter J. Butler
Professor of Biomedical Engineering
Faculty Reader

Justin L. Brown
Assistant Professor of Biomedical Engineering
Faculty Reader

* Signatures are on file in the Schreyer Honors College.

ABSTRACT

Single-molecule microscopy is a versatile tool that can be used to investigate the stepping mechanism of motor proteins such as kinesin, and to determine the copy number of subunits within membrane bound proteins such as the cellulose synthesis complex. Step detection algorithms provide a means for uncovering key information within single-molecule microscopy data collected from these systems.

Kinesin proteins are intracellular molecular motors that utilize energy from adenosine triphosphate (ATP) in order to transmit force and transport cellular cargo along microtubule tracks. Despite the current wealth of knowledge regarding these proteins, many unresolved mechanisms of the kinesin stepping cycle remain. Step detection algorithms that recover underlying piecewise-constant signals within noise-corrupted, single-molecule time series position data provide a strategy for resolving these mechanisms. The work presented in this thesis shows that by treating a positional time series as an observation sequence from a hidden Markov model, we can apply the model-dependent, continuous Viterbi algorithm in order to determine the most likely hidden state sequence of the tracked motor protein. This approach has the critical capability of keeping "phase" of plateaus within a given time series, which allows for more accurate determination of kinetic rates and motor domain displacements associated with state transitions during stepping.

In growing plant cells, cellulose synthesis complexes (CSCs) exist in the plasma membrane as six-lobed rosettes that contain different cellulose synthase (CESA) isoforms, but the number and stoichiometry of CESAs in each CSC are unknown. To begin to address this question, we performed photobleaching of GFP-tagged AtCESA3-containing particles in living *Arabidopsis thaliana* cells followed by step detection analysis to estimate copy number. The step detection algorithms introduced in this work account for changes in signal variance due to changing numbers of fluorophores in order to avoid overfitting. These procedures can be applied to photobleaching data for any complex with large numbers of fluorescently tagged subunits, providing a new analytical tool with which to probe complex composition and stoichiometry.

TABLE OF CONTENTS

LIST OF FIGURES iv

LIST OF TABLES xi

Chapter 1 Introduction 1

1.1 Kinesin Motor Proteins 2

1.1.1 Physiological Relevance 3

1.1.2 Kinesin Stepping Cycle 3

1.1.3 Unresolved Mechanisms 6

1.1.4 Single-Molecule Motility Assays 8

1.2 Cellulose Synthesis Complex 12

1.3 Step Detection Algorithms 15

Chapter 2 Results: Kinesin Motor Proteins 17

2.1 Model-Dependent Step Detection 17

2.1.1 Generative Hidden Markov Model for Kinesin Single-Molecule Assays 17

2.1.2 Continuous Viterbi Algorithm 20

2.1.3 Modified Hidden Markov Model Construction for Viterbi Algorithm 21

2.1.4 Observation Sequence Offset Detection 23

2.1.5 Iterative Continuous Viterbi Algorithm (ICV) 25

2.2 Theoretical Kinesin Motor Model 27

2.2.1 Detecting the ATP-Waiting State 27

2.2.2 ATPγS Experiments to Probe Neck-Linker Docking 31

2.3 Discussion 33

Chapter 3 Results: Cellulose Synthesis Complex 35

3.1 Imaging CESA Complexes in *Arabidopsis* Seedlings 35

3.2 Generating Simulated Fluorescence Photobleaching Data 37

3.3 Using Step Detection Algorithms to Identify Bleaching Events 39

3.4 Determining Unitary Step Size from Step Detection Results 43
3.5 Using Unitary Step Size to Estimate Fluorophore Copy Number 47
3.6 Estimating Copy Number for Kinesin-4XGFP 48
3.7 Estimating Copy Number for GFP-AtCESA3 50
3.8 Materials and Methods 52
3.8.1 Photobleaching Experiments 52
3.8.2 Image Analysis 52
3.8.3 Tdetector1 Algorithm 53
3.8.4 Tdetector2 Algorithm 54
3.8.5 Calculation of Variance of Corrupting Noise 55
3.8.6 Difference of Means Significance Testing 58
3.8.7 Differences of Variances 60
3.8.8 Bdetector Algorithms 64
3.8.9 Photobleaching Rate Estimation 65
3.8.10 Definition of Sensitivity and Precision for Step Detection Algorithms 65
3.8.11 Density Estimation 66
3.9 Discussion 67

Chapter 4 Conclusion 71

Appendix A: Tdetector Step Detection Algorithm 72

Appendix B: Iterative Continuous Viterbi Algorithm 81

Appendix C: Differences of Variance; Pairwise Differences vs. χ^2 87

REFERENCES 89

LIST OF FIGURES

Figure 1.1: Schematic from [9] of conventional kinesin (kinesin-1) cargo transport along a microtubule with approximate scaling. 2

Figure 1.2: Conservative model for conventional kinesin stepping cycle adapted from [33]. Red and blue triangular objects represent kinesin motor domains (heads), bold black connecting lines represent the neck-linker and beginning of coiled-coil stalk domains. “D” denotes the ADP state, “DP” denotes the ADP and inorganic phosphate (P_i) state, “T” denotes the ATP state, and Φ denotes the no-nucleotide state. Kinetic rates of the cycle are denoted by k_x. Grey and white tracks represent microtubule binding sites (tubulin dimers). 5

Figure 1.3: Progression of kinesin stepping cycle with respective on-microtubule-axis and off-axis distances of the red, trailing motor domain (left-most in state 1). Progression of states is from left to right. Note that states 1, 2, 3, and 4 are identical to the conservative model of Figure 1.2. “δ” represents the unknown on-axis displacement that results from trailing head detachment (state 1 to 2). “ε” represents the unknown off-axis displacement associated with a head being unbound from the microtubule. The “↦ M” and “↦ S” denote the beginning of the mobile and stable sequences of the initial trailing (red) motor domain. For homodimeric motors like kinesin-1 the M and S sequences will be identical processes. 6

Figure 1.4: Schematic (top) and a selected region of typical 2D image data (bottom) of a kinesin single-molecule motility assay. Viewpoint is normal to the cover slip surface. Motors and microtubules in schematic diagram are not to scale. Gold spots on tethered heads represent labeling of motor domains (N-terminal labeling). 9

Figure 1.5: Simulation of positional time series signals (of red motor domain) from single-molecule kinesin motility assays. Note that consecutive states 3_M, 4_M, 1_S, 2_S, 3_S, 4_S, and 1_M share the same on-axis position in this case (where subscript denotes either being part of the mobile or stable sequence). 10

Figure 1.6: Piecewise-constant signal without noise.....15

Figure 1.7: Piecewise-constant signal hidden in white noise ($\sigma = 0.25$).....16

Figure 2.1: Schematic of generative hidden Markov model and simulated series for observed on-axis distance. Numbered nodes represent the set of hidden states (numbered according to model in Figure 1.2), connecting arrows represent elements of the transition matrix, **A**. Normal distribution parameter elements of the emission matrix, **B**, including their appropriate corrections (multiples of the microtubule lattice spacing, 16.4 nm) are shown to the right of the graph.....19

Figure 2.2: Observation sequence produced by generative hidden Markov model for kinesin stepping (cyan line) where $\delta = 4$ nm. Updated mean values of the generated true hidden state path (dotted black line). Mean values of the most likely modified hidden state path as returned by the continuous Viterbi algorithm (red line). 23

Figure 2.3: Example result of offset detection and correction. Simulated on-axis distance time series from generative hidden Markov model for kinesin stepping plus an arbitrary offset, ω, (dotted cyan line), and offset underlying piecewise-constant signal (dotted black line). Identical time series (red line) and underlying piecewise-constant signal (solid black line) corrected by estimated offset, ω'.....25

Figure 2.4: Example of iterative continuous Viterbi (ICV) algorithm step detection results (green line) on simulated on-axis distance time series (red line) produced by the generative hidden Markov model for kinesin stepping with $\delta = 4$ nm. Underlying piecewise-constant signal from simulation shown with black line. Figure inset: mean squared errors of Viterbi results for each of the twenty δ values tested. Green spot indicates the optimal value, δ' (4.3 nm), i.e. δ with minimum mean squared error. The hidden state path given this $\delta' = 4.3$ nm is the sequence returned by the iterative continuous Viterbi algorithm.....26

Figure 2.5: On-axis position observation sequences (grey lines) produced by the theoretical kinesin generative model at [ATP] = 50 uM. Iterative continuous Viterbi (ICV) step detection results have been separated into even and odd plateau groups. The plateau group with the greater mean plateau size within a given trace is designated

as the long plateaus group (long = cyan lines; short = blue lines). Underlying piecewise-constant signal shown with black lines. Traces have been shifted after step detection to avoid overlay. 28

Figure 2.6: Step size and plateau size results from iterative continuous Viterbi (ICV) fitting of ten observation sequences (as in Figure 2.5) produced by the generative model at [ATP] = 50 uM. Plateau sizes have been grouped into long and short plateau sizes (i.e. dwell times of the compressed and ATP-waiting states, respectively). Step sizes have been grouped according to the identity of the plateau that preceded them. Mean plateau sizes are 37.8 time points (long) and 8.9 time points (short). Step size modes from kernel density estimation are 5.4 nm (following long plateaus) and 10.5 nm (following short plateaus).....29

Figure 2.7: Iterated continuous Viterbi (ICV) results across a range of ATP concentrations. The left plot shows the inverse of mean short plateau size as a function of [ATP] ($k_{short} = 1/E[\mathbf{p}_{short}]$). The middle and right plots show kernel density estimation modes of step sizes that follow long plateaus and short plateaus, respectively, as a function of [ATP]. Blue lines indicate ICV step detection results. Red lines indicate results given the true hidden state path. Green lines indicate the expected values given the generative model parameters. 31

Figure 2.8: Alternate kinesin stepping model in which ATP hydrolysis is required before neck-linker docking. 31

Figure 2.9: Inverse mean of short plateau sizes ($k_{short} = 1/E[\mathbf{p}_{short}]$) as a function of [ATP] (solid black) and [ATPγS] (dotted grey). Left plot indicates ICV results from sets of observation sequences produced by the original model in which ATP/ATPγS binding causes immediate neck-linker docking. Right plot indicates ICV results from observation sequences in which neck-linker docking follows hydrolysis of bound-head ATP/ATPγS.....33

Figure 3.1: In vivo photobleaching of GFP-AtCESA3. (A) Photobleaching trace of a single GFP-AtCESA3 particle in hypocotyl cells of Arabidopsis seedling. Video is recorded at 5 fps and total time is 100 s to allow most GFP to be photobleached. Representative Movie S1 is included in Supplementary Data of [1]. Inset: ensemble

average of 77 photobleaching traces with exponential fit to the data. (B) Quantitative model describing photobleaching. The fluorescence signal is assumed to fall over time with constant step sizes, matching the quantal fluorescence of a single GFP. The GFP fluorescence and the background signal are treated as Gaussian distributions, Normal (μ, σ^2) and Normal (0, δ^2), respectively. The time before fluorophore bleaching, T, is assumed to be exponentially distributed with mean $\tau = 1/\lambda$ where λ is the photobleaching rate constant. The signal to noise ratio (SNR) is defined as the step size divided by the standard deviation. (C) Simulated photobleaching trace from 12 fluorophores with $\mu = 500$ a.u., $\sigma = \delta = 250$ a.u. (D) Simulated stepping data such as a kinesin walking along a microtubule in and optical trap experiment, with $\mu = 1$, $\sigma = 1$ and 10% backward steps. (Figure from [1], created by Y.C. and N.C.D.) 36

Figure 3.2: Step detection algorithms. (A-C): Bdetector algorithm. (A) To fit the first step, Bdetector scans all possible change points and calculates a corresponding BIC value at each position (blue line). If the minimum BIC is lower than the BIC value for not adding a step (green line), a step is added (red line) at the position where the minimum BIC occurs. (B) Keeping the first step, Bdetector rescans all possible change points and calculates new corresponding BIC values (blue line), and adds a second step at the position of the minimum BIC (red line). This process is iteratively repeated. (C) When the minimum BIC value for adding an additional step (blue line) is not lower than the current BIC value (green line), the program terminates. (D-F): Tdetector algorithm where, in contrast to the BIC, a higher significance for the t-test indicates a better fit. (D) To add the first step, the significance at each possible change point is calculated (blue line) and is compared to the threshold (green line). Provided it is above the significance threshold, a step is added at the point of maximum significance (red line). (E) The data are split into two segments at the detected change point and the procedure is repeated for each segment (splitting the right segment into two in this case). This process is repeated for each new segment until adding a step does result in a significance value greater than the threshold. The algorithm then moves on to another segment. (F) When adding a change point fails to raise the significance above the threshold for every segment, the program terminates. (Figure from [1], created by Y.C. with assistance from N.C.D.).....39

Figure 3.3: Detecting steps in simulated stepping data. (A) Histograms of step sizes predicted by all step detection algorithms. The simulated data have uniform step sizes of 1 with 10% backward steps and SNR of 1. Real step sizes are calculated by comparing the means of plateau regions on either side of a step. The mode at +1 represents forward steps and the mode at -1 represents backward steps. The four algorithms detect unitary forward and backward steps, but also have modes centered at +2, corresponding to twice the single step size and representing missed steps. (B) Sensitivity plots for the four algorithms. The missed steps corresponding to the lower sensitivity of Bdetector2 can be seen in (A) by the population centered at +2 step size. (C) Precision plots for the four algorithms. Bdetector1 had problems with overfitting, resulting in lower precision and a number of steps between 0 and 1 in (A). (Figure from [1], created by Y.C. with assistance from N.C.D.).....41

Figure 3.4: Detecting steps in simulated photobleaching data. (A) Simulated photobleaching data (black) with step detection by the Tdetector2 (red) and Bdetector2 (blue) algorithms. (B, C) Precision and sensitivity plots for the four algorithms. The two algorithms not assuming equal variance (Bdetector2 and Tdetector2) gave better precision but missed events, whereas Bdetector1 and Tdetector1 gave better sensitivity but led to false positives. (Figure from [1], created by Y.C. with assistance from N.C.D.).....42

Figure 3.5: Comparing methods of fitting photobleaching step size distributions to extract unitary step size. Histograms represent step size distributions from Tdetector2 applied to simulated photobleaching data with copy number 12 and SNR = 2. The distribution is made up of 570 detected steps. (A) Fit of two Gaussian functions to the data using a bin size of 50. Fit parameters are $\mu_1 = 510$ a.u., $\sigma_1 = 55$, $\mu_2 = 836$ a.u., and $\sigma_2 = 335$. (B) Fit of two Gaussian functions to the data using a bin size of 150. Fit parameters are $\mu_1 = 568$ a.u., $\sigma_1 = 67$, $\mu_2 = 873$ a.u., and $\sigma_2 = 342$. In both cases fits to more than two Gaussians did not converge. (C) Identifying modes by KDE. A histogram with bin size 50 is plotted for the purpose of visual comparison but is not used for fitting. Smooth curve is the estimation of multiple Gaussians (kernels) by KDE. (Figure from [1], created by Y.C.) 44

Figure 3.6: Step size and copy number determination for simulated photobleaching data. (A) BIC values using different numbers of Gaussians in the GMM density estimation for the same distribution used in Figure 3.5. The best fit (smallest BIC value) was achieved with 5 Gaussians. (B) Corresponding fit of 5 Gaussians to the step size data (histogram is for display only and is not used by the GMM procedure). Red, green, yellow, pink, and purple traces represent the five Gaussians in the GMM fit, with corresponding means of 560, 921, 1376, 1811, 2343 a.u., and relative weights of 0.461, 0.341, 0.162, 0.028, and 0.008. The standard deviation, which is assumed to be identical for all modes, is 135.9 a.u. Blue line is the overall density. The unitary step size is calculated as $\Sigma_{(i = 1 \text{ to } k)} ((P_i * \mu_i)/i)$, where P_i and μ_i are the relative weight and the mean, respectively, of the i^{th} peak, resulting in a value of 528.3 a.u. (C) Predicted unitary step size as a function of SNR and copy number, demonstrating good performance for copy numbers of below 12 at SNR of 1 and above, and for copy number of 20 at SNR of 2 and above. Actual step size in simulated data was 500 a.u. (D) Predicted copy number from simulated photobleaching data with SNR of 2 and copy number 12. Peak position from KDE (black line) corresponds to mean copy number of 12.3. (E) Predicted copy number across different SNR ratios. Similar to the step size estimate, a break point at SNR below 2 was seen for prediction on copy number 20. (Figure from [1], created by Y.C. with assistance from N.C.D.).....46

Figure 3.7: Estimating copy number for kinesin-4xGFP. (A) Trace of kinesin-4xGFP bleaching (black) with steps fitted by Tdetector2 (red). (B) The BIC search leads to a best fit of $k = 4$ Gaussians for fitting the step size distribution. (C) Estimating the unitary step size (60.8 a.u.) from the step size distribution (455 total detected steps). The mean values of the four modes were 63.9, 109.9, 165.8, and 258.1 a.u., relative weights were 0.622, 0.289, 0.062, and 0.027, and the SD was 19.6 a.u. (D) Copy number distribution. There were two peaks, centered at 3.28 and 6.65. These peaks are consistent with the binomial nature leading to a slight shift from four toward lower copy number and with a double-aggregate population at roughly twice the copy number of the first peak. Histograms (black boxes) are also plotted in C and D for reference but not used in the GMM fitting. (Figure from [1], created by Y.C. with assistance from N.C.D.) 48

Figure 3.8: Copy number estimation for GFP-AtCESA3 particles. (A) Trace of GFP-AtCESA3 photobleaching (black) with steps fitted by Tdetector2 (blue). (B) BIC values for step detection at increasing number of Gaussians, showing the minimum at $k = 6$. (C) Estimation of unitary step size (445.4 a.u.) by GMM based on 730 total detected steps. Step size distribution was fitted by six Gaussians, shown in red, green, yellow, pink, and purple. Mean values were 453, 864, 1337, 1799, 2335, and 3082 a.u., relative weights were 0.4953, 0.3325, 0.1252, 0.0367, 0.0074, and 0.0027, and the SD was 160 a.u. Overall fit from GMM is shown in blue. Histogram (black boxes) is also plotted for reference but not used in the GMM fitting. (D) Copy number distribution for GFP-AtCESA3 particles. Two peaks are evident from the histograms, and fitting two Gaussians (red and green curves) gives means of 9.56 and 23.5 and ratio of 0.844 and 0.156, with SD of 4.03. (Figure from [1], created by Y.C. with assistance from N.C.D.) 51

Figure 3.9: Plots of theoretical X, Y, Z vectors where $\sigma^2 = 1$, $d = 5$, $L = 100$, and $i = 40$ 56

LIST OF TABLES

Table 2.1: Summary of single-molecule motility assay hidden Markov model (HMM) parameters assuming the conservative kinesin stepping model. Table describes the univariate observation sequence case; observed values represent on-microtubule-axis distance of an N-terminal-labeled motor domain during processive stepping. 19

Table 2.2: Viterbi algorithm pseudocode for construction of storing matrices $\mathbf{T_1}$ and $\mathbf{T_2}$. Categorical emission variables assumed. **A**[:, n] denotes all rows in the n^{th} column of matrix **A**. See Appendix B for complete MATLAB implementation of the continuous Viterbi algorithm.....20

Table 3.1: Pseudo/MATLAB code of iterative pairwise difference outlier removal 58

Table 3.2: Empirically calculated standard deviation multiplier lookup table for DOM significance testing. Data vector lengths, L, are rounded values of $2^{(n/2)}$ where n = 0,1,2, …, 26. Multipliers between given L values can be linearly interpolated with good reliability. The last two L values in the table are untested extrapolations of the trend. 59

Chapter 1

Introduction

This thesis focuses on two separate topics within distinct biological systems: the mechanism of the kinesin motor protein stepping cycle, and the molecular makeup of the cellulose synthesis complex. Both topics are similar in that critical information can be revealed using step detection analysis of single-molecule microscopy data from these systems. This thesis introduces novel, high-precision step detection algorithms designed for these specific single-molecule data sets as well as generic time series signals. Chapter 2 investigates the capabilities of these algorithms in uncovering information within simulated signals of kinesin motor protein stepping. Chapter 3 investigates the capabilities of these algorithms applied to experimental and simulated cellulose synthesis complex photobleaching data.

The kinesin work presented here is unpublished. All positional kinesin data presented in this thesis are from simulations based on: current understanding of the kinesin stepping cycle, as well as high temporal and high spatial resolution single-molecule kinesin-1 tracking data recently acquired by Keith J. Mickolajczyk [unpublished] using interferometric scattering microscopy (iSCAT).

The cellulose synthesis work presented here was recently published [1]. N.C.D. developed the t-test-based step detection algorithms and the photobleach rate estimation and correction process. Y.C. developed the Bayesian Information Criterion (BIC)-based algorithms, Gaussian Mixture Model fitting process, and created the figures. Y.C. and C.T.A. performed raw data collection. All authors contributed to the design of experiments, overall data analysis approach, and writing of the paper.

1.1 Kinesin Motor Proteins

The intracellular molecular motor, kinesin, uses the energy from adenosine triphosphate (ATP) hydrolysis in order to perform directed transport by taking discrete "steps" along cytoplasmic filaments called microtubules. The most comprehensively studied kinesin is kinesin-1, commonly referred to as conventional kinesin. It is a dimer of two identical polypeptide chains called kinesin heavy chains (KHCs) that bind to two separate polypeptides called kinesin light chains (KLCs). From N-terminus to C-terminus, a single KHC consists of: the globular, catalytic motor domain or "head" which binds to the microtubule and also binds and hydrolyzes ATP; the relatively short "neck-linker" which tethers the motor domain to the stalk of a dimerized kinesin; and the stalk which is a relatively long alpha helical chain that facilitates dimerization by forming a coiled-coil with another KHC stalk [2, 3, 4, 5, 6, 7, 8] (see Figure 1.1, from [9]). KLC tails bind to the C-terminus of the KHC stalks while also binding to intracellular cargo. KLCs also play a regulatory role by suppressing futile ATP hydrolysis [10].

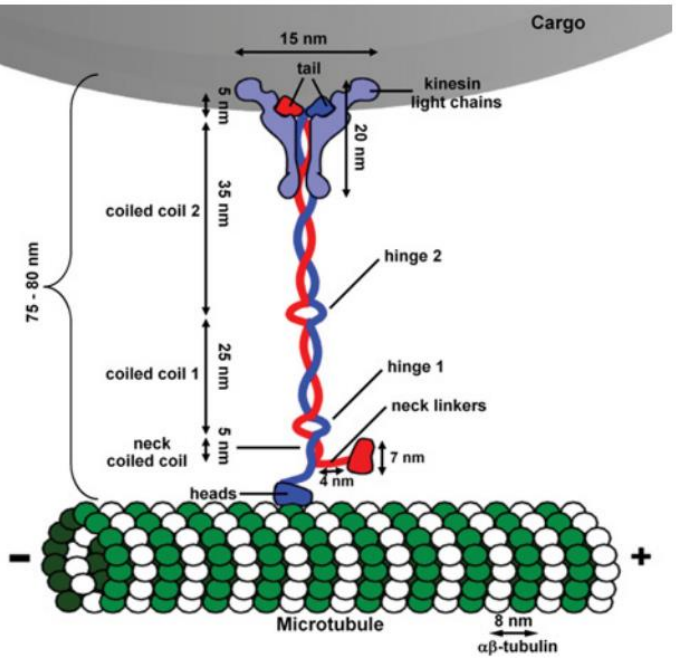

Figure 1.1: Schematic from [9] of conventional kinesin (kinesin-1) cargo transport along a microtubule with approximate scaling.

Processive kinesin stepping is accomplished by head domains alternating between being in a tight microtubule binding state and a weak microtubule binding state so that one head can step to the next binding site while one head maintains its connection to the

microtubule. This process is analogous to climbing up a ladder using only one's hands – each hand alternates between either gripping tightly to a rung of the ladder or letting go to find the next rung. The ATP hydrolysis cycle and the mechanical strain of the neck-linker domains provide the queues and communication in this coordinated hand-over-hand process.

1.1.1 Physiological Relevance

There are many different types of kinesins, and they perform a vast array of critical cellular tasks [11, 12]. For example, kinesin-1 facilitates anterograde axonal transport by binding and carrying intracellular cargoes (such as mitochondria, lysosomes, and endoplasmic reticulum) long distances towards synapses of neurons [13, 8], kinesin-2 participates in the bidirectional intraflagellar transport process [14, 15], and kinesin-5 plays a key role in mitotic spindle formation during the process of cell division [16]. More than one-hundred different kinesins have been identified since the first kinesin (kinesin-1) was discovered by Vale et al in 1985 [13, 17]. How the many different types of kinesins' structures and ATP hydrolysis cycles have been evolutionarily tuned for their diverse cellular tasks is not well understood.

Kinesin dysfunction has been linked to several neurological disorders including amyotrophic lateral sclerosis (ALS), hereditary spastic paraplegia, and Charcot-Marie-Tooth disease [18]. Due to kinesin facilitating mitotic spindle formation, it also has a highly relevant role in proliferation of cancer cells, and anti-cancer drugs that work by inhibiting function of mitotic spindle motor (kinesin-5) are being actively pursued [19, 20, 21, 22, 16]. A better understanding of kinesin's mechanism will have broad impacts on understanding these physiological problems.

1.1.2 Kinesin Stepping Cycle

Conventional kinesin advances unidirectionally towards the plus-end of a microtubule in discrete steps. Tubulin dimers, the subunit of the microtubule polymer estimated to have a spacing of 8.2 nm [23], serve as the binding sites for kinesin heads. Conventional kinesin has been shown to walk in a "hand-over-hand" fashion [24, 25] which means each step consists of the trailing head detaching from its microtubule binding site (one tubulin dimer), moving past the leading bound head, and then binding to the site adjacent

to, and in front of, the leading head. Thus, the trailing head moves the length of two tubulin dimers (16.4 nm) while the leading head remains bound to its site. A single step, which results in an 8.2 nm mean displacement of the entire kinesin motor, requires the energy of one hydrolyzed ATP molecule [26, 27]. On average, the motor takes more than 100 steps along the microtubule at a rate of approximately 100 steps per second before dissociating [27, 28].

In order to step consistently before dissociating, both heads must be highly coordinated with one another. Without coordination, both states will regularly be in a weak microtubule binding state at the same time which will result in rapid dissociation of the entire motor from the microtubule.

Whether a head is in a high or low microtubule affinity state is predominantly determined by its nucleotide state, which is defined by the form of the nucleotide, if any, that is bound to the motor domain at a given point in time. Interhead tension giving rise to gating mechanisms is also believed to control microtubule affinity of the head domains [29, 30]. Interhead tension is transmitted by the neck-linker domains that join the heads to the stalk. Recent studies have shown that neck-linker length dictates the unloaded processivity of many different kinesins – longer, compliant neck-linkers transmit strain poorly, which diminishes the coordination of their head domains, resulting in shorter run lengths, while shorter neck-linkers transmit strain efficiently, which improves the coordination of their head domains, resulting in longer run lengths [31].

A conservative model of kinesin-1 hand-over-hand stepping that is consistent with experimental kinetics and single-molecule data [32, 30, 28] provides a framework for understanding the role of the ATP hydrolysis cycle and interhead tension in providing coordination between heads (Figure 1.2).

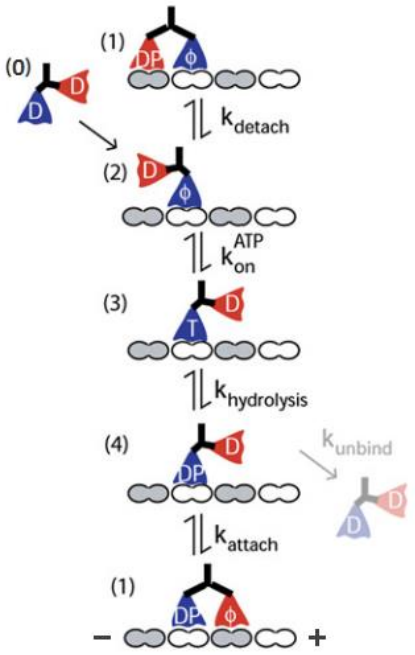

Figure 1.2: Conservative model for conventional kinesin stepping cycle adapted from [33]. Red and blue triangular objects represent kinesin motor domains (heads), bold black connecting lines represent the neck-linker and beginning of coiled-coil stalk domains. "D" denotes the ADP state, "DP" denotes the ADP and inorganic phosphate (P_i) state, "T" denotes the ATP state, and Φ denotes the no-nucleotide state. Kinetic rates of the cycle are denoted by k_x. Grey and white tracks represent microtubule binding sites (tubulin dimers).

In this model, when a motor first binds to a microtubule, the binding head promptly releases its ADP (state 0 to 2 of Figure 1.2), thus changing to the extremely high microtubule affinity state – the no-nucleotide state. Next, ATP binds to that head (state 2 to 3), inducing a conformational change termed "neck-linker docking" that biases the trailing, tethered head towards the adjacent plus-end microtubule binding site (state 3). The bound head then hydrolyzes its ATP, thus entering the ADP.P_i state (state 3 to 4). The tethered head that had translated towards the plus end via neck-linker docking undergoes its diffusional search, binds to the microtubule, promptly releases its ADP, and becomes the new leading head (state 4 to 1). During state 1, the rear-head gating mechanism suggests that interhead tension accelerates phosphate release and subsequent trailing head detachment (state 1 to 2). Upon entering state 2, the cycle is back where it began while the mean position of the entire kinesin motor has advanced 8.2 nm in the plus end direction.

Despite general agreement with most aspects of this conventional kinesin model among researchers in the field, there remain many unresolved questions regarding certain of its

mechanisms and kinetic rates of state transitions. Furthermore, the stepping cycle varies between different types of kinesin. Robust models for other kinesin families are not well-established.

1.1.3 Unresolved Mechanisms

ATP binding is not rate-limiting at normal physiological ATP concentrations (> 1 mM). This is evident from average velocity measurements of kinesin motility plateauing in the presence of ATP concentrations in the millimolar range or above (referred to as "saturating ATP") [27]. However, one controversial and unresolved question is whether different types of kinesin sit in a one- or two-head bound state as they wait for ATP to bind to their front head [34, 35, 36, 37] (state 2 or 2'of Figure 1.3).

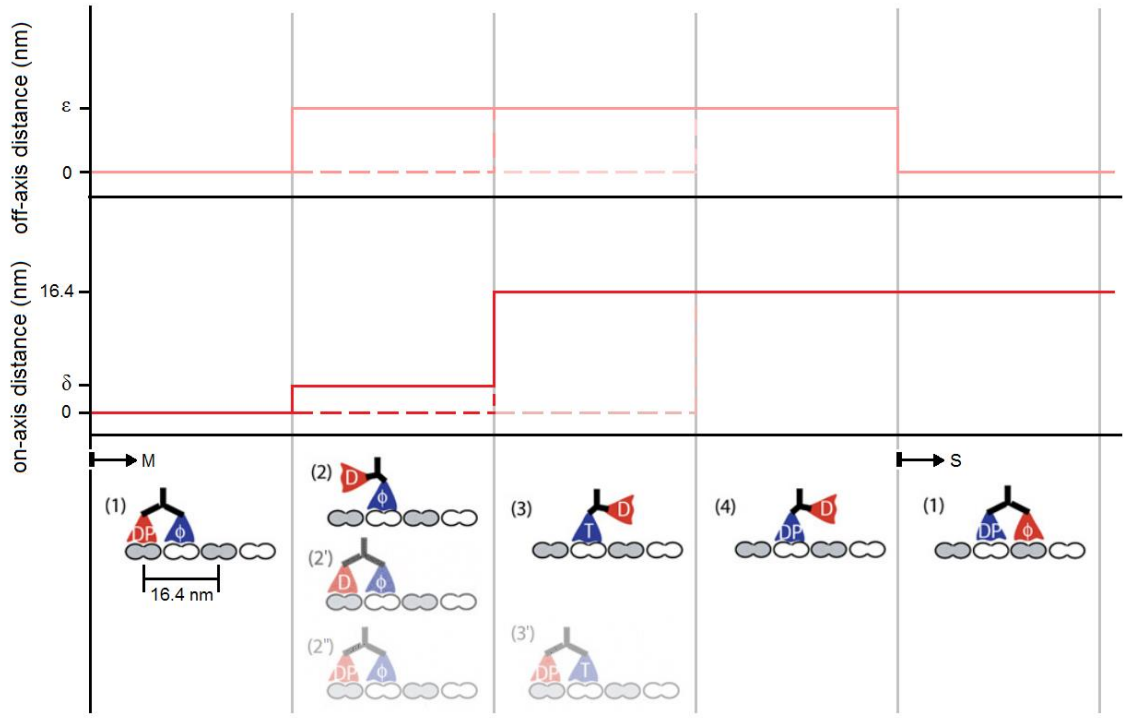

Figure 1.3: Progression of kinesin stepping cycle with respective on-microtubule-axis and off-axis distances of the red, trailing motor domain (left-most in state 1). Progression of states is from left to right. Note that states 1, 2, 3, and 4 are identical to the conservative model of Figure 1.2. "δ" represents the unknown on-axis displacement that results from trailing head detachment (state 1 to 2). "ε" represents the unknown off-axis displacement associated with a head being unbound from the microtubule. The "↦ M" and "↦ S" denote the beginning of the mobile and stable sequences of the initial trailing (red) motor domain. For homodimeric motors like kinesin-1 the M and S sequences will be identical processes.

Single-molecule kinesin-1 experiments performed by Yildiz et al. [25], in which a single head domain was fluorescently labeled and its position tracked with nanometric

precision, provided support for a two-head bound waiting state – or at least a state in which the trailing head did not move noticeably from its previous microtubule binding site upon dissociation (δ displacement in Figure 1.3 approximately 0 nm). They were able to fit discrete steps made by their labeled head domain, and they only observed step sizes deviating around one mode. As commented upon in their analysis, a bimodal distribution of step sizes (modes of δ nm and 16.4 - δ nm) would be expected if kinesin sat in a displaced one-head bound state while waiting for ATP binding (see Figure 1.2).

One would not expect δ (displacement from state 1 to 2 of Figure 1.3) to be zero given that there is expected to be tension within the neck-linker in state 1. Upon trailing head dissociation the tension should relieve by displacing the trailing head towards the microtubule plus-end now that it is free to do so. Furthermore, on this scale, since the neck-linker is not expected to be in a rigid conformation in state 2, thermodynamic fluctuations alone should displace the mean position of the free head towards the origin of its tethering (δ of ~ 8 nm). The 2’’-state is not expected to be part of the regular pathway of processive motors, as it is likely to lead to dissociation via ATP binding and hydrolysis in the leading head causing both domains to be in weak microtubule binding states [33]. The 2’-state as a regular two-head-bound ATP-waiting-state is not expected given that the ADP state is a well-established weak-binding state of conventional kinesin [38], however it is possible that this particular stage of the cycle is an exception to that rule. To clarify, the displayed state 1 to 2 transition implies that inorganic phosphate release leads to immediate detachment of the trailing head, whereas the state 1 to 2’ transition suggests that rapid dissociation of the trailing head does not occur until front-head ATP binding takes place (state 2’ to 3). It should also be noted that despite states 3 and 4 in this model being in a docked neck-linker conformation, which has been evolutionarily-tuned to allow the free head to find its next microtubule binding site, the possibility of their on-axis distance being considerably different than the 16.4 nm microtubule lattice spacing should not be strictly ruled out. Furthermore, it is not perfectly clear that neck-linker docking occurs immediately with ATP binding, it may instead require ATP hydrolysis first (not shown in Figure 1.3, see Section 2.2.2).

Another unresolved issue involves the determinants of unloaded and loaded processivity across different N-terminal kinesins. Unloaded processivity has been shown to strongly depend on neck-linker length [31] while loaded processivity depends exclusively on the properties of the catalytic motor domains [39]. However, precise mechanisms to explain these observations have not been established. Neck-linker length may dictate processivity by way of accelerating trailing head detachment from state 1 to 2 (i.e. increasing k_{detach}) and/or by increasing the rate of tethered head binding, k_{attach} in Figure 1.2, which transitions the motor out of a potential unbinding opportunity in state 4, to a stable state 1. Loaded processivities are most-likely dictated by kinetic rates of the cycle that determine the portion of time a motor spends in vulnerable one-head bound states.

The precise coupling of the ATP hydrolysis cycle and stepping cycle is still not well-established. Single-molecule motility assays – imaged with high-resolution microscopy techniques that reveal on- and off-axis displacements of labeled kinesin motors undergoing processive stepping – provide a means for uncovering the characteristics of a kinesin's stepping cycle and its mechanism.

1.1.4 Single-Molecule Motility Assays

Single-molecule kinesin motility assays emulate the fundamental processive stepping behavior of the motor along a microtubule (see Figure 1.4). In these assays, stable microtubules are fixed to the surface of a microscope coverslip while added motors step along individual microtubule tracks in an adequate ATP-concentrated and buffered solution. Total internal reflection fluorescence (TIRF) microscopy is often used to image these assays since this microscopy technique allows for exclusive collection of emitted fluorescence near the coverslip surface, therefore reducing unwanted background noise from the bulk of the sample [40]. In order to image kinesin with TIRF or any other fluorescence microscopy technique, the protein must be labeled with a fluorophore. The fluorophore can be located on the N-terminus (motor domain) or the C-terminus (stalk or tail domain). Emitted light from the fluorophore is collected by the objective lens and then recorded by the detector (camera). In general, the resulting data from these assays is in the form of a stack of 2D arrays of pixel intensities representing the viewpoint normal

to the coverslip surface at each point in time (i.e. a 3D array where third dimension is time).

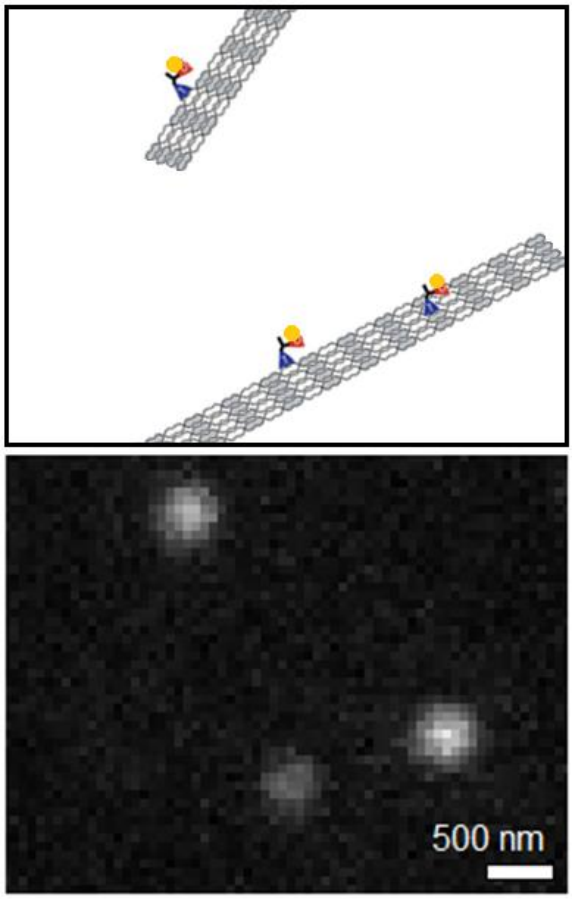

Figure 1.4: Schematic (top) and a selected region of typical 2D image data (bottom) of a kinesin single-molecule motility assay. Viewpoint is normal to the cover slip surface. Motors and microtubules in schematic diagram are not to scale. Gold spots on tethered heads represent labeling of motor domains (N-terminal labeling).

Established image processing techniques allow for 2D arrays of data to be transformed into time series traces of X- and Y-position for each individual fluorophore in the field of view [41]. If the motors have been engineered to have an N-terminus label, then a properly-rotated set of these X- and Y-position vs. time traces represent the on- and off-axis displacements shown in Figure 1.3.

As with any measurement, the resulting time series signals of position will be corrupted by some degree of noise. This noise may be due to contributions from background signal, vibrations of the microscope stage, read noise of the detector, or other sources. For

stationary and photostable probes, X- and Y-positional noise after image processing is well-characterized by a normal distribution of zero mean and some constant variance, σ^2.

$$X_{noise} \sim N(0, \sigma^2)$$

$$Y_{noise} \sim N(0, \sigma^2)$$

As a result of noise, the on- and off-axis distances associated with each state of the stepping cycle will be partially hidden (see simulated signals in Figure 1.5).

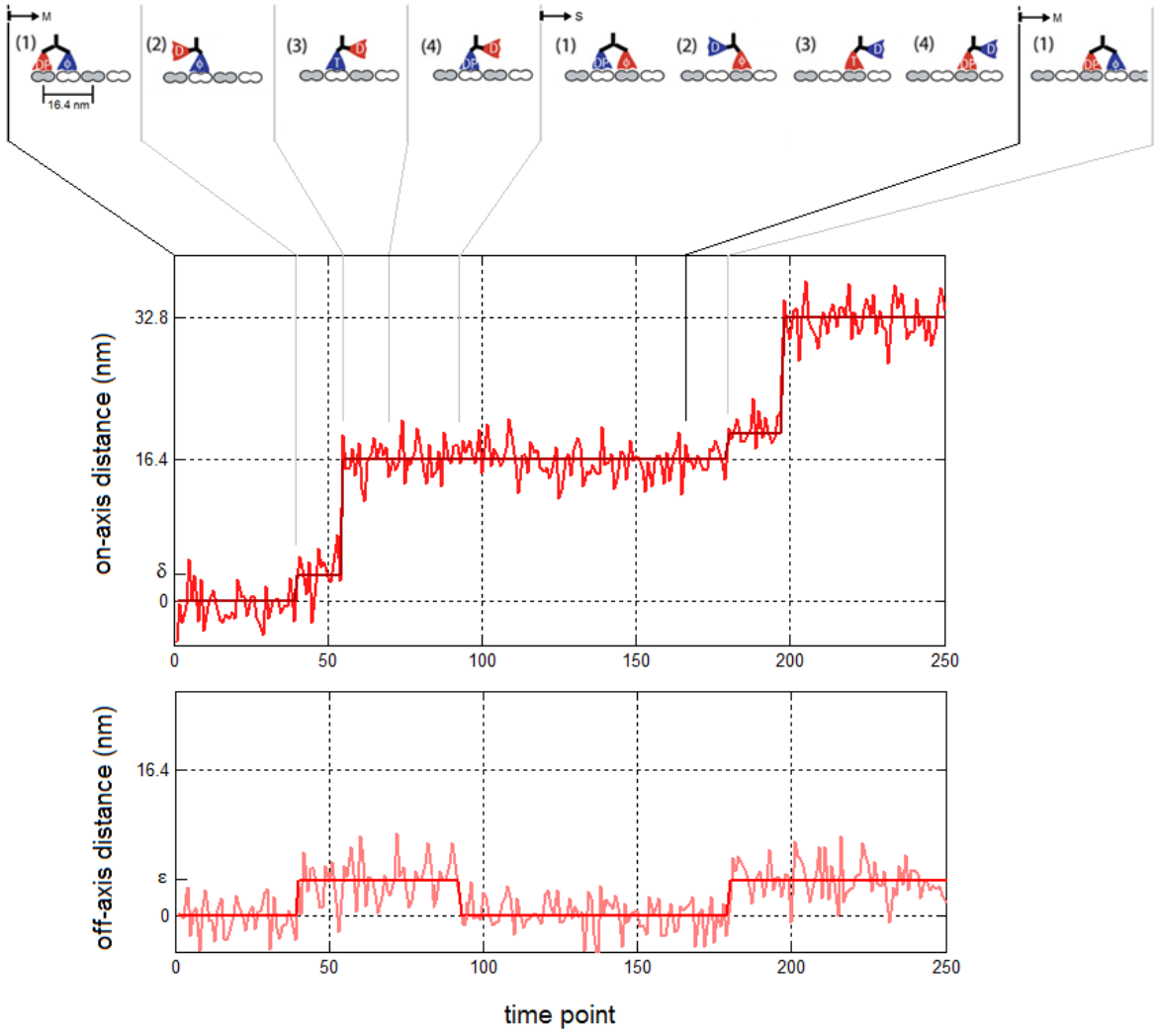

Figure 1.5: Simulation of positional time series signals (of red motor domain) from single-molecule kinesin motility assays. Note that consecutive states 3_M, 4_M, 1_S, 2_S, 3_S, 4_S, and 1_M share the same on-axis position in this case (where subscript denotes either being part of the mobile or stable sequence).

Thus, positional time series signals from single-molecule kinesin motility assays are analogous to observation values of a hidden Markov model (HMM). An HMM describes

a stochastic process in which an object transitions discretely among a set of states, each with its own probability distribution that defines the likelihood of "emitting" particular values when the object is in that state [42]. Only the sequence of these emitted values (which depend only on the current state at that point in the sequence) are observable, the sequence of states itself is not, hence the states are said to be "hidden."

Significant advancements in single-molecule imaging technologies, including interferometric scattering microscopy (iSCAT) [43] and total internal reflection dark-field microscopy (TIRDFM) [44], are now making it feasible to detect these hidden state transitions at physiological ATP concentrations. Rather than relatively weak fluorescence signals, these methods rely on photon scattering, which allows for drastically improved temporal and spatial resolution of on- and off-axis position during stepping.

If the hidden state sequence can be recovered from these traces, it will reveal a rich source of information regarding the coupling of the stepping and hydrolysis cycles. Even with advanced microscopy techniques, noise is relatively substantial compared to kinesin step displacements. Therefore it is critical to have non-biased, highly-precise algorithms for uncovering the underlying piecewise-constant signal within noise-corrupted time series data sets.

1.2 Cellulose Synthesis Complex

Cellulose is a major structural component in the plant cell wall that regulates plant cell growth and morphology and also has extensive commercial value for applications such as papermaking, textile manufacturing, and biofuel production [45]. However, the molecular processes involved in the biosynthesis of cellulose, which is composed of large numbers of β(1,4)-linked glucan chains that associate via hydrogen bonds to form cellulose microfibrils, remain incompletely understood despite intensive research over the past 15 years [46]. It is generally believed that cellulose is synthesized at the plasma membrane and extruded into the extracellular space by a cellulose synthesis complex (CSC). Each CSC contains many GT2-family glucosyltransferases called cellulose synthases (CESAs) and is assembled into a large integral membrane complex with a membrane-spanning rosette configuration of approximately 25 nm in diameter [47]. The complex is formed in the Golgi and transported to the plasma membrane, where it becomes active to synthesize the glucan chains that constitute the cellulose microfibril [46]. Genetic and biochemical data indicate that a minimum of three different CESA isoforms are present in each CSC; in the model plant *Arabidopsis thaliana*, AtCESA1, AtCESA3, and AtCESA6-type proteins are present in CSCs that synthesize cellulose in the primary walls of growing cells, whereas AtCESA4, AtCESA7, and AtCESA8 proteins are present in CSCs during secondary wall synthesis in cells that have ceased growth [48, 49, 50]. Estimations based on structural studies of cellulose microfibrils [51, 52] and molecular modeling of CESAs [53] predict that each CSC is composed of anywhere between 12 and 36 subunits [54, 46]; however, the precise stoichiometry of CESA isoforms within each CSC remains undefined. Empirically determining protein copy numbers for intact membrane-bound CSCs through nondestructive means is challenging, especially since reconstituting active, purified plant CSCs has proven to be extremely difficult [55, 56, 57].

One alternative method of estimating protein copy numbers in integral membrane complexes is to count bleaching steps for subunits tagged with intrinsically fluorescent proteins, such as green fluorescent protein (GFP), under total internal reflection fluorescent (TIRF) microscopy [58]. However, the number of proteins that can be estimated using current methods is limited: higher copy numbers lead to increases in both fluctuations in the fluorescence signal and the initial rate of photobleaching, complicating

the identification of discrete photobleaching steps. This issue can be addressed by using a median filter to reduce noise in the data, and constructing pairwise distance distributions to determine the unitary step size of photobleaching [23, 59]. However, implementing this approach to estimate subunit number typically requires empirical selection of the optimal median filter, and still does not readily resolve the precise timing and magnitude of individual bleaching steps.

Step detection algorithms, which are frequently used to analyze the spatial steps undertaken by motor proteins, are capable of automatically detecting change points in data traces [60]. Numerous methods have been developed to detect steps, but most of them depend heavily upon pre-selected parameters. Notably, the χ^2 method developed by Kerssemakers et al. requires an input of the number of steps to be detected [61], which is difficult to calculate if prior information about the data is unavailable. Methods based on information criteria are objective and do not require user-defined input parameters [62]. However, they have only been implemented in step detection algorithms by assuming that the variance associated with each step is constant [62], which is adequate for single motor protein stepping but not for photobleaching. Because intensity fluctuations of individual fluorophores around their means are uncorrelated, the presence of multiple active fluorophores in a complex will result in a higher variance in the fluorescence intensity signal than the variance associated with a single fluorophore. Hence, algorithms designed to detect steps in photobleaching data need to consider these variance changes to avoid overfitting during periods of high fluorescence intensity. Another complexity in photobleaching data is that with increasing copy number, there is an increasing probability that two or more fluorophores will bleach within a short timeframe (e.g., within a single acquisition period), which can also skew the step size distribution and complicate the estimation of a unitary photobleaching step size. Thus, there also exists a need for the development of objective analytical tools to extract unitary step sizes from step-size distribution densities that improve upon current methods of data binning and fitting a user-defined number of Gaussian functions.

In this work we developed a novel procedure that combines step detection and density estimation to determine unitary step size and copy number from experimental

photobleaching data. A mathematical model was constructed to generate simulated bleaching data, and the simulated data were used to optimize the performance of the step detection and density estimation algorithms and demonstrate their ability to accurately retrieve copy numbers from simulated data with varying degrees of experimental noise. A key goal in developing these tools was to make them as objective as possible by minimizing the number of user-defined parameters, and it is hoped that these procedures will establish best practices for analyzing photobleaching data derived from complexes with high copy numbers. We applied these analytical tools to photobleaching data collected for GFP-tagged AtCESA3 in intact cells of *Arabidopsis thaliana* seedlings and estimated the lower limit of copy number per particle to be ten.

1.3 Step Detection Algorithms

Step detection is a common problem encountered in signal processing in which the goal is to identify discrete changes in the mean of a signal. This problem is trivial if the signal contains little or no noise, but statistical approaches must be applied when the signal is hidden in relatively high noise (see Figures 1.6, 1.7). Step detection can be considered a subset of the more general class of problems referred to as change detection, or change point detection, in which the aim is to identify discrete changes in many different features of a signal including: variance, spectral density, correlation, etc. These signal processing problems are encountered in many engineering disciplines as well as in biophysics, biology, and bioinformatics [63, 60, 64, 65].

In general, a step detection algorithm is a function that accepts a noise-corrupted time series signal as input, and then returns a list of points at which there is a discrete change in the mean value as the output. Specific step detection algorithms differ in the assumptions made about the features of the input signal. Issues that are considered include, but are not limited to: the nature of the corrupting noise (e.g. normally distributed, exponentially distributed, etc.); whether or not the variance of the corrupting noise changes across the signal; whether or not the signal is autocorrelated; whether or not an accurate model for the generation of the signal is available.

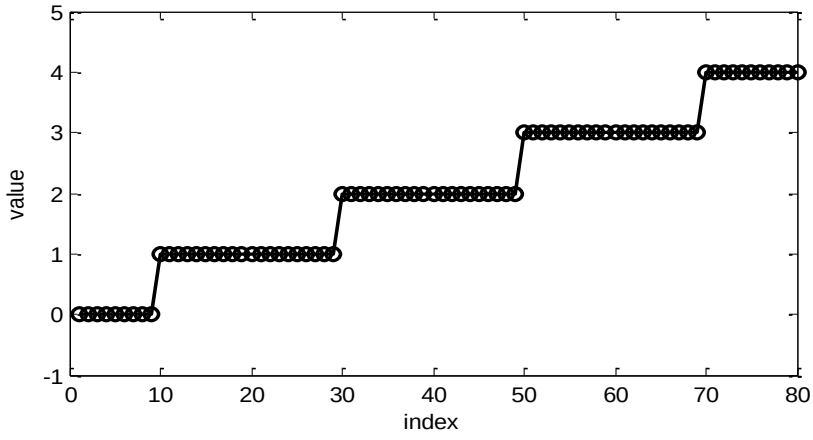

Figure 1.6: Piecewise-constant signal without noise

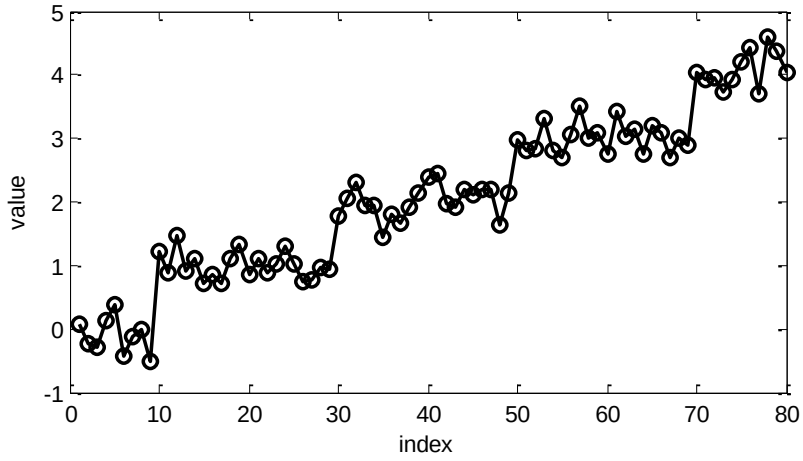

Figure 1.7: Piecewise-constant signal hidden in white noise ($\sigma = 0.25$)

The step detection algorithms described in this thesis are designed to accept piecewise-constant signals hidden in Gaussian white noise (zero or negligible autocorrelation; see Figure 1.7). The goal of the algorithms is to identify the indices (i.e. time points) at which there is a significant and discrete change in the mean value of the underlying piecewise constant signal with respect to noise. The mean value of the sections between these indices can then be calculated to recover the best estimate of the underlying piecewise-constant signal. Step detection algorithms that depend on an input hidden Markov model are presented in Chapter 2 (Section 2.1) and are applied to kinesin motor protein stepping data. Model-independent algorithms are presented in Chapter 3 (Sections 3.8.3 – 3.8.8) and are applied to cellulose synthesis complex photobleaching data.

Chapter 2

Results: Kinesin Motor Proteins

The kinesin work presented here is unpublished. All positional kinesin data presented in this thesis are from simulations based on: current understanding of the kinesin stepping cycle, as well as high temporal and high spatial resolution single-molecule kinesin-1 tracking data recently acquired by Keith J. Mickolajczyk [unpublished] using interferometric scattering microscopy (iSCAT) [43].

2.1 Model-Dependent Step Detection

2.1.1 Generative Hidden Markov Model for Kinesin Single-Molecule Assays

One approach to the step detection problem is to make prior estimations of the model that generates the observed sequence, namely in the form of a hidden Markov model (HMM). The Viterbi algorithm [66] can be used to determine the most probable hidden state sequence, called the Viterbi path, given an observation sequence and a set of model parameters. Iterating this algorithm through different potential HMM parameters followed by error calculations of the returned sequences provides an alternate strategy for uncovering model parameters that is more direct than model-independent step detection approaches.

Simple HMMs are defined by the following parameters. N: total number of hidden states in the model. T: total number of observations. **x**: sequence (T-by-1) of hidden states in which element x_t is the true hidden state at t (any integer 1 to N). **y**: sequence (T-by-1) of observation values in which element y_t is the observed value at t. **A**: transition matrix (N-by-N) in which element a_{ij} denotes the probability of the hidden state transitioning from state *i* to state *j* given that it is currently in state *i*, $a_{ij} = P(x_{t+1} = j \mid x_t = i)$. **B**: emission matrix (N-by-"1") in which element $\mathbf{b}_n$ is a set of parameters that describe $P(y_t = z \mid x_t = n)$, i.e. the probability that y_t takes on any value, z, in the observation variable space

given that the current hidden state, $x_t = n$. **U**: a prior probability matrix (N-by-1) in which element U_n denotes the probability that the initial hidden state, x_1, is state n.

As described previously (Section 1.1), observations of single-molecule motor protein motility assays can be described by an HMM. Let us return to the model of the kinesin stepping cycle and define it in the context of an HMM (see following Table 2.1 and Figure 2.1 for summary). For now we will consider only the on-microtubule-axis position data (see Figure 1.5) as the observation sequence (HMM parameter **y**). A given value in the sequence of observed on-axis position of a kinesin motor, y_t, should depend only on the current hidden state of the motor, x_t. Thus the set of hidden states is defined by the states of the kinesin stepping cycle model (N = 4; states 1, 2, 3, and 4 in Figures 1.2, 1.3, 1.5). Kinesin transitions between different states according to different kinetic rate constants. Therefore the relative magnitudes of these individual rates and the detector (camera) sampling rate will define the elements of the transition matrix, **A**. The expected values of observed on-axis displacement for an N-terminal labeled motor given the state are determined by the microtubule lattice spacing (16.4 nm) and the displacement associated with trailing-head detachment (δ in Figure 1.3), though there will be some degree of randomness due to noise in the measurement. As stated in Chapter 1, the noise of a given positon signal of a photostable probe in single-molecule microscopy is well-characterized by the normal distribution with zero mean and some constant variance, σ^2. The emission matrix, **B**, will contain these necessary parameters (b_{n1} = mean, μ, and b_{n2} = variance, σ^2) for the univariate normal distribution associated with hidden state n. Note that for motors taking multiple steps, b_{n1} values will require some form of updating. The probabilities, **U**, for the initial hidden state would depend on whether the observation sequence began with the initial binding of the motor to a microtubule, or with the arbitrary start of detector recording. In the former case, this would suggest that the initial hidden state is guaranteed to be state 2; $U_2 = 1$ (see model in Figure 1.2). In the latter case, initial state probability should be a function only of average time spent in each state. Finally, the value for T is the length of the position observation time series.

Table 2.1: Summary of single-molecule motility assay hidden Markov model (HMM) parameters assuming the conservative kinesin stepping model. Table describes the univariate observation sequence case; observed values represent on-microtubule-axis distance of an N-terminal-labeled motor domain during processive stepping.

Symbol	HMM parameter	Analogous kinesin single-molecule assay parameter
N	Number of hidden states	Number of discrete states in the kinesin stepping cycle (N = 4)
T	Total observations	Total number of frames recorded by detector
A	Transition matrix	Probabilities of transitioning among states as defined by relative kinetic rates of the kinesin and detector sampling rate
B	Emission matrix	Parameters of normal distributions, $N(\mu,\sigma^2)$, describing probability of observing a given on-axis distance value for each hidden state (note: requires updating)
U	Initial probabilities	Probabilities of starting in a given state of the stepping cycle

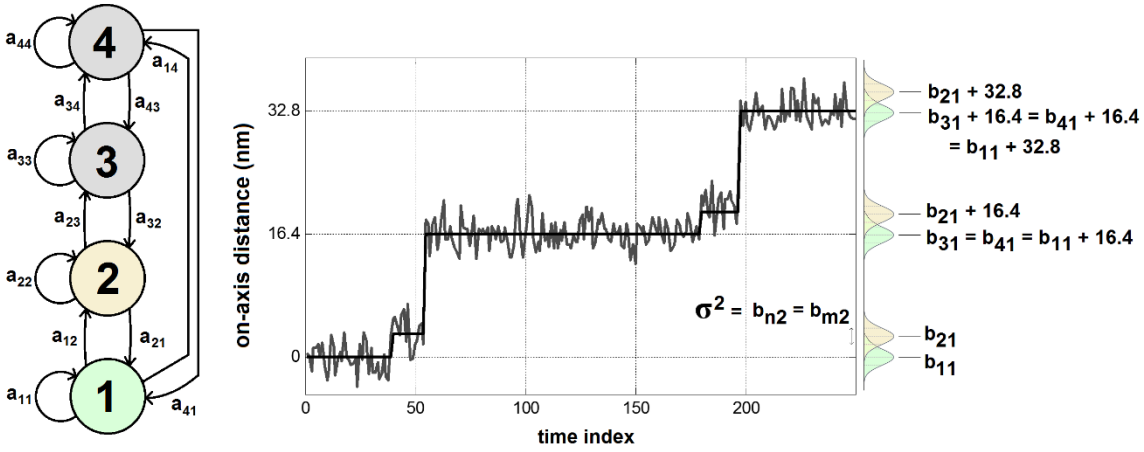

Figure 2.1: Schematic of generative hidden Markov model and simulated series for observed on-axis distance. Numbered nodes represent the set of hidden states (numbered according to model in Figure 1.2), connecting arrows represent elements of the transition matrix, **A**. Normal distribution parameter elements of the emission matrix, **B**, including their appropriate corrections (multiples of the microtubule lattice spacing, 16.4 nm) are shown to the right of the graph.

These parameters (summarized in Table 2.1 and Figure 2.1) form a generative model for an observed on-axis-position time series. Given an observation time series, **y**, and HMM parameters **A**, **B**, and **U**, the Viterbi algorithm returns the most likely hidden state sequence (parameters N and T can be inferred from others). A brief description of the Viterbi algorithm follows, along with a solution to the emission matrix updating complication.

2.1.2 Continuous Viterbi Algorithm

The Viterbi algorithm [66] is essentially a recursion of Bayes' Theorem that stores the most likely previous hidden state, x'_{t-1}, for each possible hidden state at t (stored in N-by-T matrix, $\mathbf{T_2}$). Also stored (in N-by-T matrix, $\mathbf{T_1}$) are the probabilities that each most likely previous hidden state, x'_{t-1}, transitioned to each following hidden state and then emitted the observed value at t, y_t. These storing matrices, $\mathbf{T_1}$ and $\mathbf{T_2}$, are constructed sequentially as described by the following pseudocode (Table 2.2). The first column of storing matrix, $\mathbf{T_1}$, is determined from y_1 and initial probabilities, $\mathbf{U}$.

Table 2.2: Viterbi algorithm pseudocode for construction of storing matrices $\mathbf{T_1}$ and $\mathbf{T_2}$. Categorical emission variables assumed. $\mathbf{A}[:, n]$ denotes all rows in the n^{th} column of matrix $\mathbf{A}$. See Appendix B for complete MATLAB implementation of the continuous Viterbi algorithm.

```
for t ← 2,3 … , T :
        for n ← 1,2 … , N :
                T1[n, t] ← max (T1[: , t − 1] .∗ A[: , n] ∗ B[n, y_t])
                T2[n, t] ← argmax (T1[: , t − 1] .∗ A[: , n] ∗ B[n, y_t])
        end
end
```

The emission probability terms, $\mathbf{B}[n, y_t]$, in Table 2.2 are for categorical or discrete emission variables. These terms reference the probability mass function described by the n^{th} row of matrix $\mathbf{B}$. For normally-distributed continuous emission variables, the $\mathbf{B}[n, y_t]$ terms can simply be replaced by the density function describing the probability of observing a certain y_t value given the hidden state, n:

$$P(Y = y_t | x_t = n, \mathbf{B}) = \frac{1}{\sqrt{2\pi\, b_{n2}}} e^{\left\{-\frac{(y_t - b_{n1})^2}{(2\, b_{n2})}\right\}} \qquad (2.1)$$

Once the storing matrices have been calculated, the most likely final hidden state, x'_T, is determined from $\mathbf{T_1}$:

$$x'_T = \underset{n}{\operatorname{argmax}}(\mathbf{T_1}[:, T])$$

The most likely hidden state path is then determined by tracing back through most likely previous hidden states stored in $\mathbf{T_2}$. This algorithm is guaranteed to return the global maximum likelihood hidden state sequence.

2.1.3 Modified Hidden Markov Model Construction for Viterbi Algorithm

To solve the issue of needing to shift the mean values in the emission matrix to account for motors taking multiple steps, we can instead construct modified HMM parameters with an expanded set of hidden states according to the range of the given observation series, **y**. Again, we will consider only on-axis position for now.

In this modified hidden state set, we compress consecutive states 3_M, 4_M, 1_S, 2_S, 3_S, 4_S, and 1_M since they will share mean emission values (recall Figure 1.5). Therefore, state numbers in this modified model no longer correspond to those of the kinesin stepping cycle presented in Chapter 1. The new number of states will be determined by the microtubule lattice spacing (16.4 nm) and the rounded maximum and minimum values of **y**:

$$N = 2\left(\left\lceil\frac{\max(\mathbf{y})}{16.4}\right\rceil - \left\lfloor\frac{\min(\mathbf{y})}{16.4}\right\rfloor + 1\right)$$

Each hidden state number now defines its mean value parameter (b_{n1}). The value for δ is unknown (see Figure 1.3), while the variance, σ^2, can be estimated accurately using the process described later in Section 3.8.5.

$$b_{n1} = \begin{cases} 16.4\left(\left\lfloor\frac{\min(\mathbf{y})}{16.4}\right\rfloor + \frac{n-1}{2}\right) & n \text{ is odd} \\ 16.4\left(\left\lfloor\frac{\min(\mathbf{y})}{16.4}\right\rfloor + \frac{n-2}{2}\right) + \delta & n \text{ is even} \end{cases}$$

$$b_{n2} = \sigma^2$$

A rough estimation of the transition matrix can be made from the observation sequence as follows (recall T = length(**y**), let η = expected ratio of forward steps to backward steps):

$$\lambda = \frac{N}{T}$$

$$\forall\, i \neq 1:\ a_{i,(i-1)} = (1-\eta)\lambda$$

$$\forall\, i:\ a_{i,i} = 1-\lambda$$

$$\forall\, i \neq N:\ a_{i,(i+1)} = \eta\lambda$$

The transition matrix in the case where N = 5 is shown as an example:

$$\mathbf{A} = \begin{bmatrix} 1-\lambda & \lambda & 0 & 0 & 0 \\ (1-\eta)\lambda & 1-\lambda & (\eta)\lambda & 0 & 0 \\ 0 & (1-\eta)\lambda & 1-\lambda & (\eta)\lambda & 0 \\ 0 & 0 & (1-\eta)\lambda & 1-\lambda & (\eta)\lambda \\ 0 & 0 & 0 & \lambda & 1-\lambda \end{bmatrix}$$

Finally, a uniform initial probability matrix can be assumed:

$$U_n = 1/N$$

Thus, given an observation sequence produced by the generative HMM model for on-axis position, the true δ value, and an approximate η value (~ 1), a complete modified HMM can be constructed. The continuous Viterbi algorithm can then accept these modified HMM parameters and the observation sequence, and then return the most likely hidden state path (modified hidden state path). The following Figure 2.2 shows an example observation sequence produced by the generative HMM model for kinesin stepping, and the results of the continuous Viterbi algorithm given the modified HMM parameters. We can see from Figure 2.2 that the continuous Viterbi algorithm with modified HMM parameters works as a highly precise model-dependent step detection algorithm.

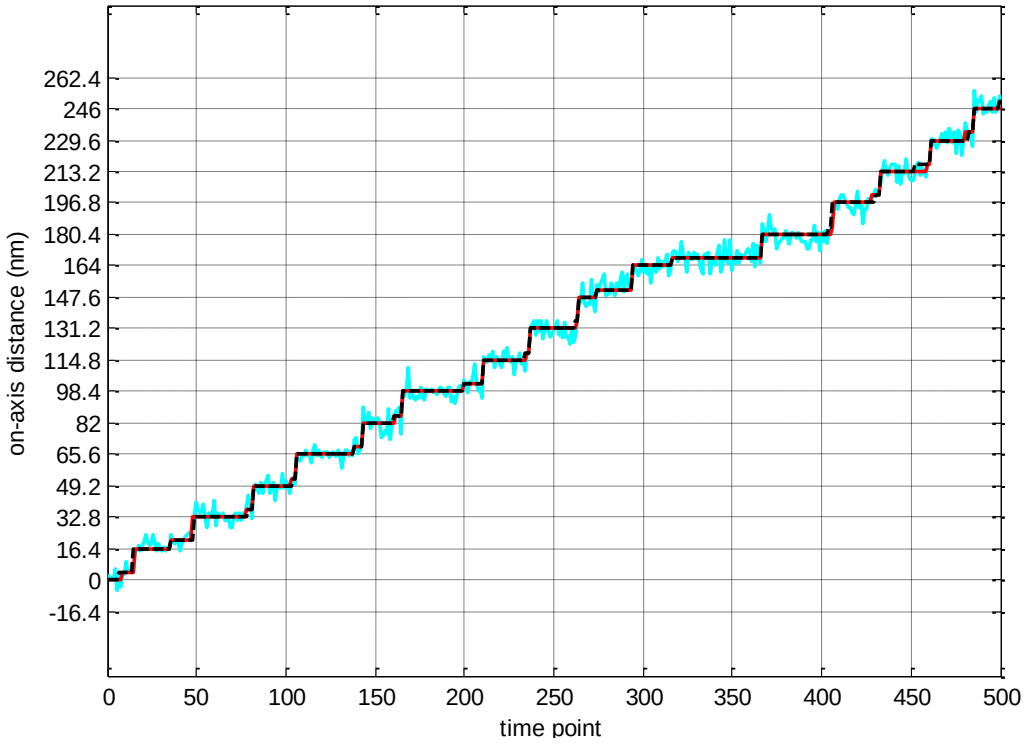

Figure 2.2: Observation sequence produced by generative hidden Markov model for kinesin stepping (cyan line) where $\delta = 4$ nm. Updated mean values of the generated true hidden state path (dotted black line). Mean values of the most likely modified hidden state path as returned by the continuous Viterbi algorithm (red line).

So far, two key assumptions about a given observation sequence have been made that must be addressed: (1) an "offset" of zero is assumed, i.e. the microtubule lattice spacing is aligned perfectly with increments of 16.4 nm, and (2) the true value for δ is assumed. Let us first address the offset problem.

2.1.4 Observation Sequence Offset Detection

Microtubule binding sites cannot be visualized in single-molecule microscopy explicitly. Only the on-axis distance time series obtained from the fluorophore attached to the motor can be used to estimate the location of microtubule binding sites. The origin of an on-axis distance time series is arbitrary; often it is defined by the position at the first time point. Even if this first time point represents the initial binding of the labeled motor to the microtubule, the error due to measurement noise makes this an inaccurate estimation of a microtubule binding site center. Thus, an observation sequence is said to have some offset, ω, relative to the true microtubule binding site spacing (16.4 nm lattice spacing)

that is not expected to be zero. Said another way, plateaus of the underlying piecewise constant signal will coincide with on-axis distances of ω, ω + 16.4, ω + 32.8 nm and so on, rather than precisely 0, 16.4, and 32.8 nm. This will render the Viterbi algorithm unable to produce an accurate or meaningful result. Therefore, it is necessary to first estimate the value of ω so that the observation sequence can be properly adjusted for input into the Viterbi algorithm. This can be accomplished using the results from a model-independent step detection algorithm, such as the Tdetector1 described previously. Note that the output of model-independent step detection algorithms like the Tdetector, do not depend on the offset of the input time series data.

The declared step indexes (indexes at which there is a significant change in the mean value) returned from the Tdetector1 algorithm can be used to create a set of plateaus, $\mathbf{p}$, where a single plateau, $\mathbf{p_i}$, contains the set of points from one declared step index to the next index. Given the set of plateaus, $\mathbf{p}$, and an assumed repeated spacing (16.4 nm for microtubule lattice), then the following steps can be taken in order to reliably estimate the offset, ω:

(1) For a given plateau, $\mathbf{p_i}$, find all plateaus, $\mathbf{p_j}$, in which the difference of their mean values is within a certain acceptable range, ξ, of an integer multiple of 16.4. That is, with $\mathbf{p_i}$ fixed, find all indices, j, in which the following inequality is true:

$$\frac{\mathrm{E}[\mathbf{p_i}] - \mathrm{E}[\mathbf{p_j}]}{16.4} - \left\| \frac{\mathrm{E}[\mathbf{p_i}] - \mathrm{E}[\mathbf{p_j}]}{16.4} \right\| \leq \xi$$

(2) Repeat this process for all plateaus in $\mathbf{p}$. Let $\mathbf{q_i}$ denote the subset of plateaus that were within range, ξ, of a 16.4-integer-multiple of plateau $\mathbf{p_i}$, including $\mathbf{p_i}$.

(3) Let $\mathbf{q_k}$ denote the subset of plateaus that has the most points within all of its contained plateaus, and that contains at least one additional plateau that is not $\mathbf{p_k}$.

(4) For each plateau, $\mathbf{p_j}$, in the optimal subset $\mathbf{q_k}$, shift all values in $\mathbf{p_j}$ by the 16.4-integer-multiple that is nearest the mean of $\mathbf{p_j}$

$$\forall\, \mathrm{j}\colon\ \mathbf{p_j} \leftarrow \left(\mathbf{p_j} - 16.4 \left\| \frac{\mathrm{E}[\mathbf{p_j}]}{16.4} \right\| \right)$$

(5) Concatenate all shifted plateaus within the optimal plateau subset, $\mathbf{q_k}$, into a single vector of points, **Q**. Now, an accurate estimate of the offset can be made from the mean of **Q**.

$$\omega' = \mathrm{E}[\mathbf{Q}]$$

The following Figure 2.3 demonstrates the results of offset detection and correction on a simulated observation sequence from the generative HMM kinesin stepping model. We can see that the offset-corrected underlying piecewise-constant signal (solid black line) lies almost perfectly on integer multiples of 16.4 nm.

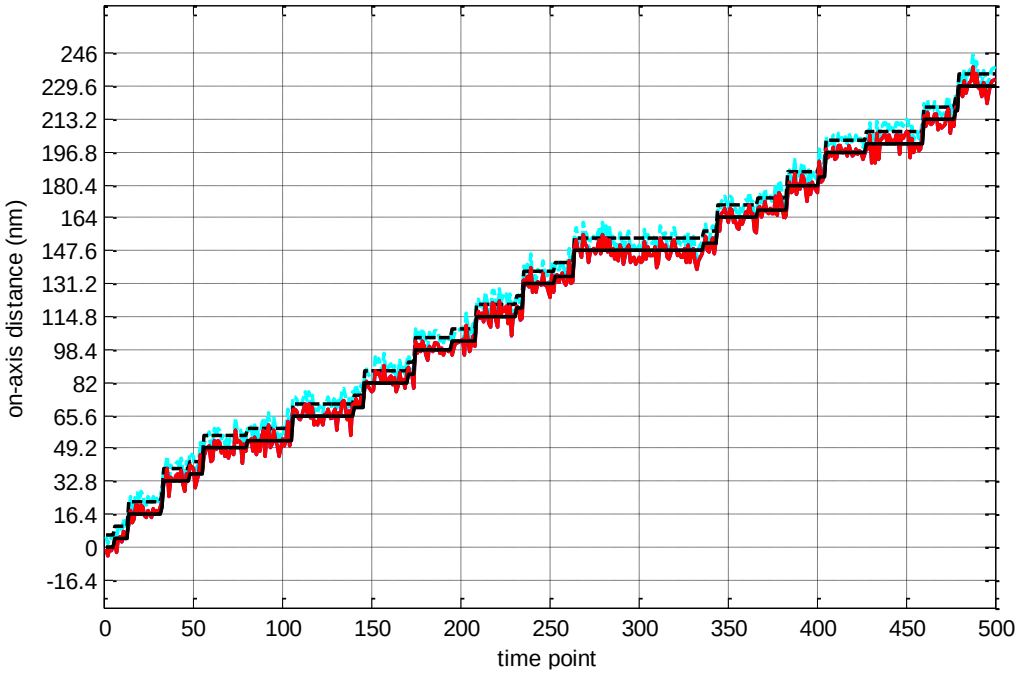

Figure 2.3: Example result of offset detection and correction. Simulated on-axis distance time series from generative hidden Markov model for kinesin stepping plus an arbitrary offset, ω, (dotted cyan line), and offset underlying piecewise-constant signal (dotted black line). Identical time series (red line) and underlying piecewise-constant signal (solid black line) corrected by estimated offset, ω'.

2.1.5 Iterative Continuous Viterbi Algorithm (ICV)

Thus far, the value of δ used in generating observation sequences has also been assumed when constructing a modified HMM emission matrix for input to the Viterbi algorithm. For experimental single-molecule motility data, the true δ value is unknown. In order for the Viterbi algorithm to function as a reliable step detection algorithm, it is necessary for

the modified HMM emission matrix to be constructed using an accurate δ value. To solve this problem, we can iterate the continuous Viterbi algorithm over many different possible δ values. The Viterbi path iteration that yields the lowest mean squared error fit with respect to the input observation sequence, **y**, should be produced by the best approximation of the true δ value. Therefore, the Viterbi results that yield the least mean squared error fit are interpreted as the model-dependent step detection results (see example results, Figure 2.4).

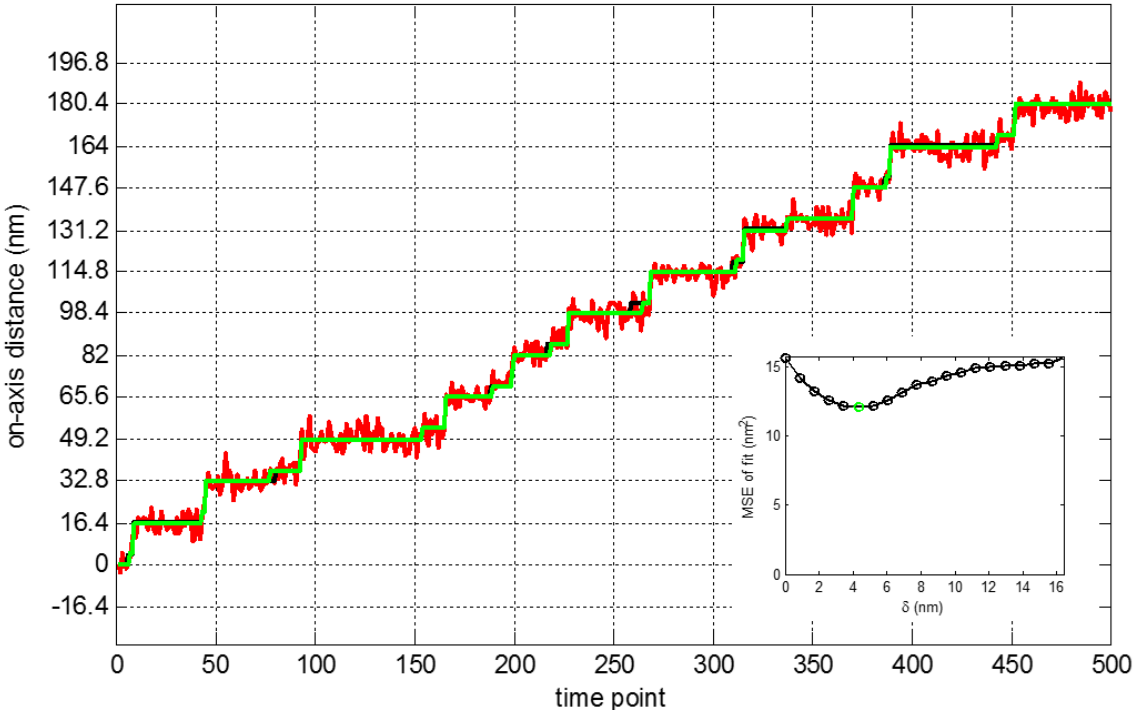

Figure 2.4: Example of iterative continuous Viterbi (ICV) algorithm step detection results (green line) on simulated on-axis distance time series (red line) produced by the generative hidden Markov model for kinesin stepping with δ = 4 nm. Underlying piecewise-constant signal from simulation shown with black line. Figure inset: mean squared errors of Viterbi results for each of the twenty δ values tested. Green spot indicates the optimal value, δ' (4.3 nm), i.e. δ with minimum mean squared error. The hidden state path given this δ' = 4.3 nm is the sequence returned by the iterative continuous Viterbi algorithm.

2.2 Theoretical Kinesin Motor Model

Let the following hidden Markov model be the generative model for observed on-axis positon of a theoretical kinesin motor based on the conservative model (Figure 1.2) imaged at f = 1000 frames/second for T = 500 frames:

$$N = 4; \quad U_n = \frac{1}{N};$$

$$\lambda = 0.15; \quad \eta = 0.99;$$

$$k_1 = \lambda; \quad k_2 = 15\ \mathrm{mM}_{\mathrm{ATP}}^{-1}\,\lambda; \quad k_3 = 10\,\lambda; \quad k_4 = \lambda;$$

$$\mathbf{A} = \begin{bmatrix} 1-k_1 & (\eta)k_1 & 0 & (1-\eta)k_1 \\ (1-\eta)k_2 & 1-k_2 & (\eta)k_2 & 0 \\ 0 & (1-\eta)k_3 & 1-k_3 & (\eta)k_3 \\ (\eta)k_4 & 0 & (1-\eta)k_4 & 1-k_4 \end{bmatrix}$$

$$\delta = 5\ \mathrm{nm}; \quad \sigma = 4\ \mathrm{nm};$$

$$\mathbf{B^M} = \begin{bmatrix} 0 & \sigma^2 \\ \delta & \sigma^2 \\ 16.4 & \sigma^2 \\ 16.4 & \sigma^2 \end{bmatrix}$$

Where $k_1 \approx k_{detach}$, $k_2 \approx k_{on}^{ATP}$, $k_3 \approx k_{hydrolysis}$, $k_4 \approx k_{detach}$ (see Figure 1.2), and $\mathbf{B^M}$ denotes the emission matrix during the mobile sequence of stepping (see Figures 1.3, 1.5).

2.2.1 Detecting the ATP-Waiting State

Given observation sequences generated by this model with arbitrary offset, ω, we can show that by performing offset correction followed by iterating the continuous Viterbi algorithm over modified hidden Markov models with varying δ values as described in Section 2.1.5 (ICV algorithm), it is possible to uncover certain parameters of the stepping cycle of this theoretical kinesin motor.

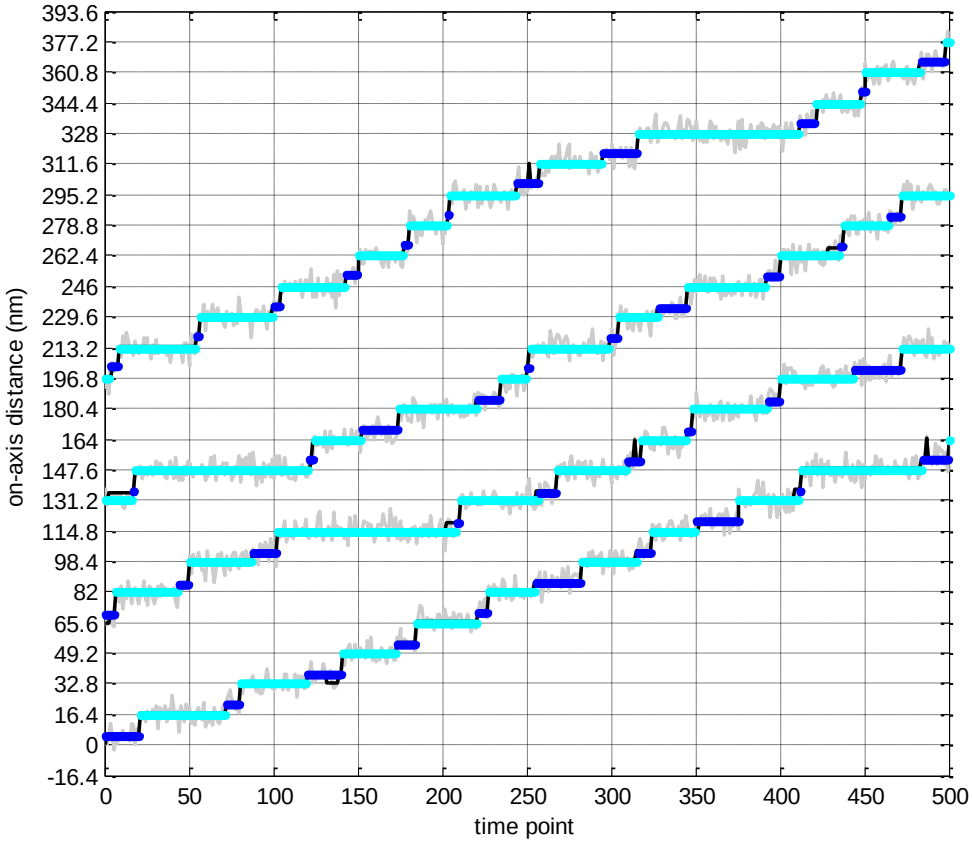

Figure 2.5: On-axis position observation sequences (grey lines) produced by the theoretical kinesin generative model at [ATP] = 50 uM. Iterative continuous Viterbi (ICV) step detection results have been separated into even and odd plateau groups. The plateau group with the greater mean plateau size within a given trace is designated as the long plateaus group (long = cyan lines; short = blue lines). Underlying piecewise-constant signal shown with black lines. Traces have been shifted after step detection to avoid overlay.

In general, with a large enough set of observation sequences at a given ATP concentration (see Figure 2.5), the combined ICV step detection results will converge to functions of the generative model parameters. The combined ICV results for step size and plateau size from ten independent observation sequences at [ATP] = 50 uM are shown in Figure 2.6.

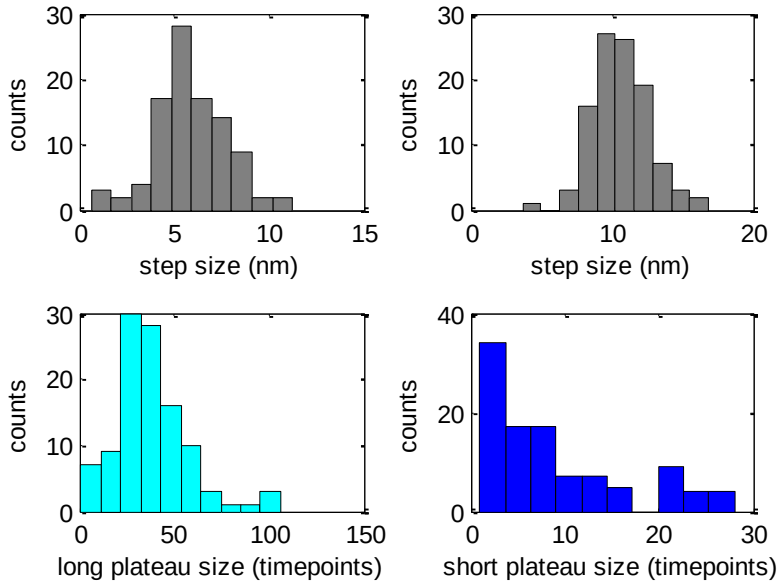

Figure 2.6: Step size and plateau size results from iterative continuous Viterbi (ICV) fitting of ten observation sequences (as in Figure 2.5) produced by the generative model at [ATP] = 50 uM. Plateau sizes have been grouped into long and short plateau sizes (i.e. dwell times of the compressed and ATP-waiting states, respectively). Step sizes have been grouped according to the identity of the plateau that preceded them. Mean plateau sizes are 37.8 time points (long) and 8.9 time points (short). Step size modes from kernel density estimation are 5.4 nm (following long plateaus) and 10.5 nm (following short plateaus).

We can see that the modes of step sizes match well to the two expected step sizes; $\delta = 5$ nm and $16.4 - \delta = 11.4$ nm. Additionally, we can show that the means of the long and short plateau size distributions match well to the expectations for dwell times. The expected duration spent in hidden state, n, of the generative model before transitioning out will be the time constant, τ_n (units = time points). The value for τ_n is given by the inverse of the sum of all transition probabilities that result in leaving state n:

$$\tau_n = \frac{1}{\sum_{i \neq n} a_{n,i}} = \frac{1}{1 - a_{n,n}}$$

As defined by the generative model for the theoretical kinesin motor:

$$\tau_n = \frac{1}{k_n}$$

Recall that in the modified model used in ICV step detection, consecutive hidden states of the generative model are compressed, while the ATP-waiting state (state 2_M) is left independent. Therefore, the time constant of the compressed state will be relatively long,

τ_{long}, due to being the sum of several first-order-process time constants, while the time constant of the ATP-waiting state will be relatively short, τ_{short}. Hidden state sequences returned by the ICV algorithm are forced to alternate between the compressed state and the ATP-waiting state. Therefore, even and odd plateau sizes can be meaningfully grouped (see plateau size distributions, Figure 2.6), and the means of these distributions should match to τ_{long} and τ_{short}. For the results in Figure 2.6, we see that they do:

$$\tau_{\text{short}} = \tau_2 = \frac{1}{k_2} = \frac{1}{15\ \text{mM}_{\text{ATP}}^{-1}\ (0.050\ \text{mM}_{\text{ATP}})\ \lambda} = 8.8\bar{8}\ \text{time points}$$

$$\approx \text{E}[\mathbf{p}_{\text{short}}] = 8.9$$

$$\tau_{\text{long}} = \tau_{3\text{M}} + \tau_{4\text{M}} + \tau_{1\text{S}} + \tau_{2\text{S}} + \tau_{3\text{S}} + \tau_{4\text{S}} + \tau_{1\text{M}} = 36.8\bar{8}\ \text{time points}$$

$$\approx \text{E}[\mathbf{p}_{\text{long}}] = 37.8\ \text{time points}$$

We can also see that the short plateau size distribution appears to be exponentially distributed while the long plateau size distribution resembles a higher order gamma distribution (Figure 2.6). This is what should be expected given the processes that define the underlying piecewise-constant signal.

Further support that the short plateaus represent the ATP-waiting state can be provided by repeating this process of observation sequence generation and ICV analysis across a range of ATP concentrations. As [ATP] is increased, the rate of ATP binding, k_2, increases proportionally, so short plateau durations should become even shorter. Figure 2.7 shows the results of this analysis. We can see that in the low [ATP] range, the inverse of mean short plateau sizes, k_{short} (i.e. k_2), increases proportionally and falls on the expected line defined by the generative model parameters. As [ATP] reaches about 100 uM, the values for k_{short} flatten out due to the ICV algorithm failing to fit exceedingly short plateaus consistently. Nevertheless, a clear relationship between mean of short plateau sizes and [ATP] is evident, which indicates that short plateaus contain the ATP-waiting state as a hidden state.

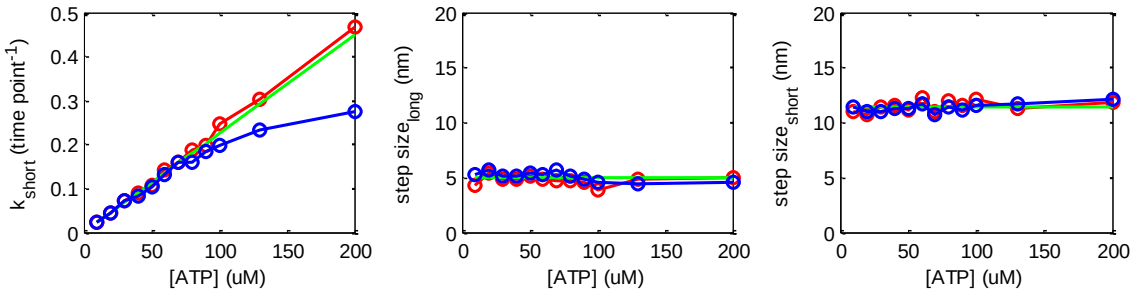

Figure 2.7: Iterated continuous Viterbi (ICV) results across a range of ATP concentrations. The left plot shows the inverse of mean short plateau size as a function of [ATP] ($k_{short} = 1/E[\mathbf{p}_{short}]$). The middle and right plots show kernel density estimation modes of step sizes that follow long plateaus and short plateaus, respectively, as a function of [ATP]. Blue lines indicate ICV step detection results. Red lines indicate results given the true hidden state path. Green lines indicate the expected values given the generative model parameters.

2.2.2 ATPγS Experiments to Probe Neck-Linker Docking

It is possible that a kinesin motor may have a different generative model than the one described at the beginning of this Section 2.2. For example, a theoretical kinesin motor may require ATP hydrolysis before neck-linker docking occurs (compare Figures 2.8, 1.2):

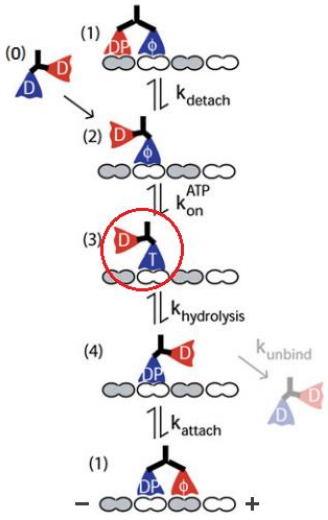

Figure 2.8: Alternate kinesin stepping model in which ATP hydrolysis is required before neck-linker docking.

This model would be defined by the following, slightly altered emission matrix where b_{31} is now set to δ. Additionally, this will result in state 3_M of the generative model transferring from the large compressed state to the formerly stand-alone ATP-waiting

state. That is, the short plateaus will now consist of the ATP-waiting state (2_M) as well as the hydrolysis-waiting state (3_M).

$$\mathbf{B^M} = \begin{bmatrix} 0 & \sigma^2 \\ \delta & \sigma^2 \\ \delta & \sigma^2 \\ 16.4 & \sigma^2 \end{bmatrix}$$

The slow-hydrolysable ATP analog, ATPγS, can be used to test if this alternate model applies for a given theoretical kinesin. In the subsequent analysis we will assume hydrolysis of ATPγS is 5-fold slower than ATP; $k_{3,ATP} = 5*k_{3,ATP\gamma S}$, and that ATP$\gamma$S and ATP have equal binding rates; $k_{2,ATP} = k_{2,ATP\gamma S}$. As nucleotide concentration (ATP or ATPγS) is increased, the duration of the ATP-waiting state (2_M) decreases since nucleotide binding will occur more rapidly. Therefore, in the alternate model, where short plateaus consist of both 2_M and 3_M, this increase of nucleotide concentration will lead to minimized 2_M state contributions towards plateau size. At high enough nucleotide concentrations, the 3_M state (hydrolysis-waiting state) will then dominate the plateau size. Consequently, if a given kinesin motor steps according to the alternate model, then high ATPγS concentrations will yield much longer plateau sizes than high ATP concentrations (i.e. k_{short} in ATPγS < k_{short} ATP). If the given motor instead steps according to the original model, then no difference in trends of short plateau sizes should be evident between ATP and ATPγS (see Figure 2.9).

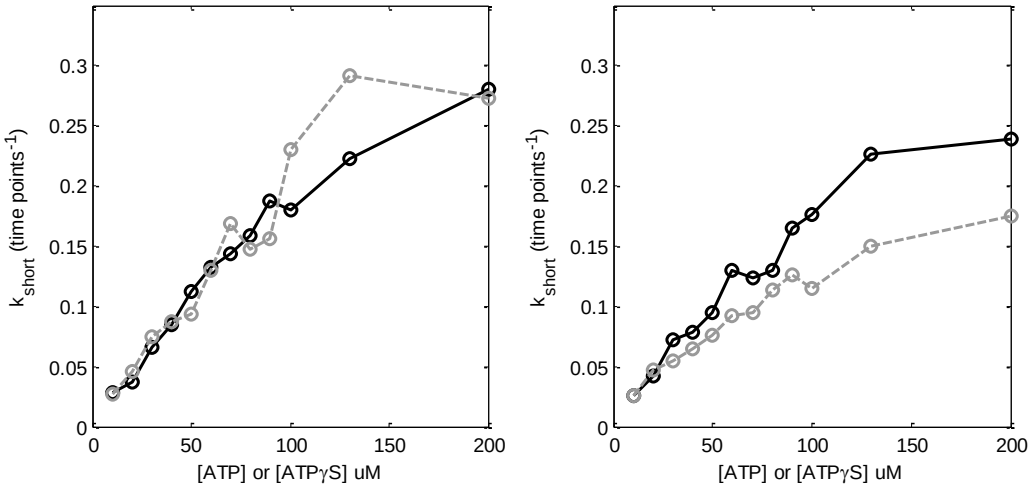

Figure 2.9: Inverse mean of short plateau sizes ($k_{short} = 1/E[\mathbf{p}_{short}]$) as a function of [ATP] (solid black) and [ATPγS] (dotted grey). Left plot indicates ICV results from sets of observation sequences produced by the original model in which ATP/ATPγS binding causes immediate neck-linker docking. Right plot indicates ICV results from observation sequences in which neck-linker docking follows hydrolysis of bound-head ATP/ATPγS.

2.3 Discussion

Given sets of experimental single-molecule on-axis position traces at different ATP concentrations, this ICV analysis should theoretically be able to identify the nature of the ATP-waiting state. That is, it should identify an associated displacement distance, δ, with considerable accuracy. It should also be able to elucidate whether or not subsequent neck-linker docking occurs immediately with binding of ATP, or if ATP hydrolysis is required first. Together, this knowledge would be critical for determining the complete mechanism of different kinesin motors.

The key advantage of the ICV algorithm applied to single-molecule kinesin stepping data, is that this step detection algorithm is able to keep track of the "phase" of plateaus. That is, by assuming the microtubule lattice spacing, odd and even plateau sizes are in theory guaranteed to represent two separate plateau size populations (ATP-waiting state plateaus and compressed state plateaus in the case of the original model, Figure 1.2). Step sizes are also separable in this way. This property allows for these distributions to be analyzed individually, which renders the process of inferring characteristics of the stepping mechanism drastically simpler and more accurate. It is not possible for a model-independent step detection algorithm to achieve this property because even a single false

positive or missed step will lose track of the plateau phase. Since analyzing odd and even plateaus (or step sizes) will be meaningless in this case, we would be forced to deal with mixed distributions that will generally have considerable overlap.

The key shortfall of the current ICV algorithm implementation is that the microtubule lattice must be assumed beforehand, and that the spacing does not allow fluctuations. The processes of offset detection and emission matrix construction currently require an assumed integer-multiple-repeat for lattice spacing which, in this thesis, has been assumed to be exactly 16.4 nm up until this point. Because this value is not perfectly accurate, the mean values of the modified hidden Markov model emission matrix used in the ICV algorithm are guaranteed to diverge from the true microtubule lattice centers given a long enough observation sequence. This will result in the loss of plateau phase fidelity. Although phase is kept successfully with shorter traces, relaxing the constraint of an integer-multiple-repeat for the emission matrix mean values is a feasible next step for improving the ICV algorithm.

Until this point, only the on-microtubule-axis position of the labeled motor domain has been considered as the observation sequence. One of the attractive features of the continuous Viterbi algorithm is its ability to be adapted relatively easily to accept a two-dimensional observation sequence (on-axis and off-axis position) and use this information simultaneously to determine the most-probable hidden state path. This can be done by changing the univariate normal probability density function shown in Equation 2.1 (Section 2.1.2) to the bivariate normal density function. The emission matrix would then require an additional column to provide the expected values for off-axis position given the current hidden state (0 or ε nm, see Figures 1.3, 1.5), while the same variance parameter may be assumed for both dimensions. By considering off-axis position, it would then be theoretically possible to detect additional hidden states (i.e. a 3-unique-state modified hidden Markov model for ICV detection rather than the 2-unique-state model). The 3_M and 4_M states would form a new state discernable from the compressed state (4_M to 1_S transition detectable by off-axis transition from ε to 0 nm), and critical information related to k_{attach} could be inferred with ATPγS experiments in a similar fashion to the neck-linker docking analysis described above.

Chapter 3

Results: Cellulose Synthesis Complex

The cellulose synthesis work presented here was recently published [1]. N.C.D. developed the t-test-based step detection algorithms and the photobleach rate estimation and correction process. Y.C. developed the Bayesian Information Criterion (BIC)-based algorithms, Gaussian Mixture Model fitting process, and created the figures. Y.C. and C.T.A. performed raw data collection. All authors contributed to the design of experiments, overall data analysis approach, and writing of the paper.

3.1 Imaging CESA Complexes in *Arabidopsis* Seedlings

To estimate the copy number of GFP-AtCESA3 in membrane-localized particles in living cells of *Arabidopsis thaliana*, 5-to-6-day-old light-grown seedlings expressing GFP-AtCESA3 [49] were mounted in an imaging chamber and recordings of GFP bleaching were carried out in hypocotyl cells containing low densities of GFP-AtCESA3 particles (see Movie S1 of [1] Supplemental Information). Imaging was performed using variable-angle epifluorescence microscopy [67], which like total internal reflection fluorescence (TIRF) microscopy reduces background fluorescence but allows for the imaging of proteins farther from the coverslip, such as those in the plasma membrane of plant cells that are separated from the coverslip by the cell wall [68, 67] (Konopka et al., 2008; Konopka and Bednarek, 2008). To quantify photobleaching rates, time lapse recordings were collected (Movie S1), and fluorescence intensity traces for individual GFP-containing particles were measured using ImageJ (see Section 3.8.2). Instead of exhibiting discrete steps, the intensity changes during photobleaching for many traces appeared to be relatively smooth (Figure 3.1A, Movie S1), suggesting that the number of fluorophores per particle is relatively high.

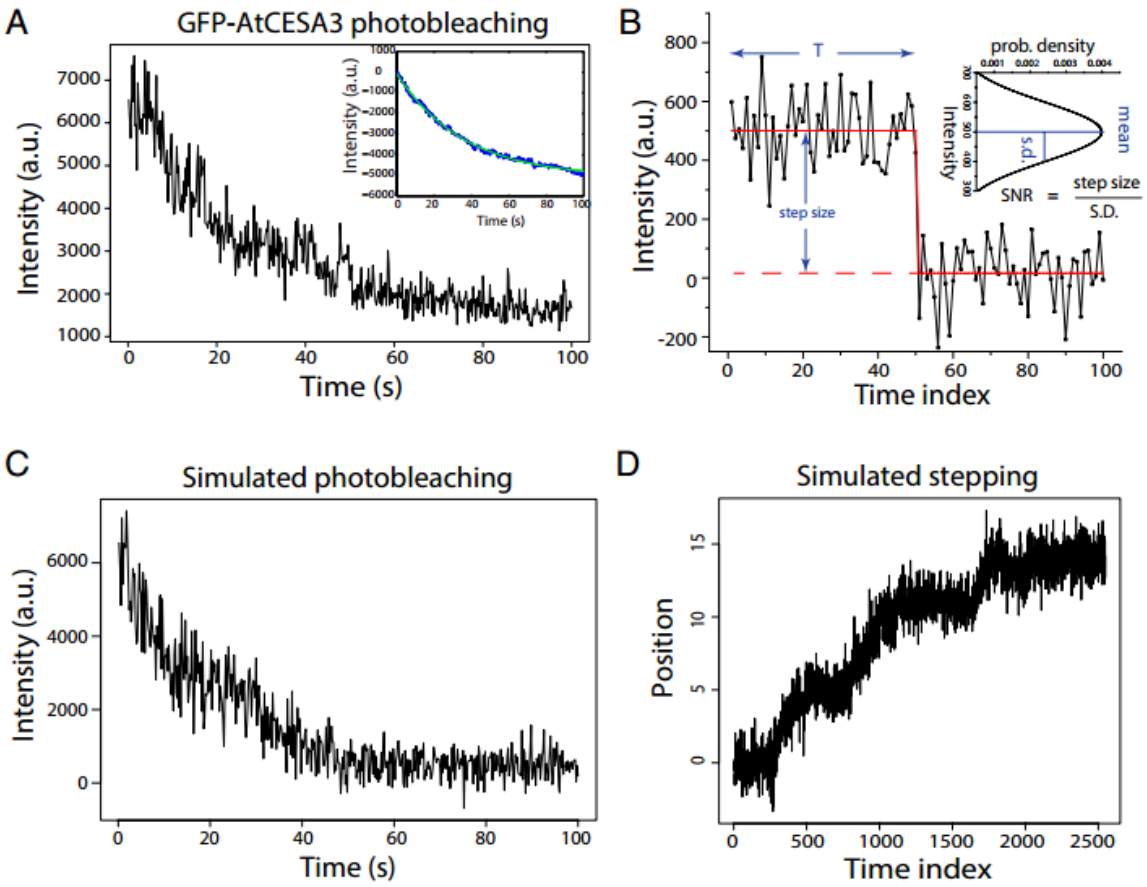

Figure 3.1: In vivo photobleaching of GFP-AtCESA3. (A) Photobleaching trace of a single GFP-AtCESA3 particle in hypocotyl cells of Arabidopsis seedling. Video is recorded at 5 fps and total time is 100 s to allow most GFP to be photobleached. Representative Movie S1 is included in Supplementary Data of [1]. Inset: ensemble average of 77 photobleaching traces with exponential fit to the data. (B) Quantitative model describing photobleaching. The fluorescence signal is assumed to fall over time with constant step sizes, matching the quantal fluorescence of a single GFP. The GFP fluorescence and the background signal are treated as Gaussian distributions, Normal (μ, σ^2) and Normal (0, δ^2), respectively. The time before fluorophore bleaching, T, is assumed to be exponentially distributed with mean $\tau = 1/\lambda$ where λ is the photobleaching rate constant. The signal to noise ratio (SNR) is defined as the step size divided by the standard deviation. (C) Simulated photobleaching trace from 12 fluorophores with $\mu = 500$ a.u., $\sigma = \delta = 250$ a.u. (D) Simulated stepping data such as a kinesin walking along a microtubule in and optical trap experiment, with $\mu = 1$, $\sigma = 1$ and 10% backward steps. (Figure from [1], created by Y.C. and N.C.D.)

The photobleaching rate constant for GFP-AtCESA3 was estimated by ensemble averaging all of the photobleaching collected traces and fitting a single exponential function using MATLAB's nonlinear least squares method (Figure 3.1A inset). The fitted rate of 0.0278 ± 0.0003 s^{-1} (mean ± SEM of fit, N = 77 traces) is the expected rate of photobleaching events regardless of the true number of independent photobleaching units present.

The experimental background noise was estimated by analyzing the distribution of the final plateau variance (as defined by the Tdetector2 step detection algorithm; see below) for the 77 measured traces. As expected, the distribution had more than one mode (Figure S1 of [1]), due to the fact that complete photobleaching had not occurred in some of the traces. Therefore the lowest variance mode was defined as the background variance, while the next mode indicates the sum of the background variance plus the variance associated with one fluorophore. To allow for more precise quantitative analysis of bleaching for multiple fluorophores, we developed a statistical method of photobleaching analysis, as described below.

3.2 Generating Simulated Fluorescence Photobleaching Data

Fluorescence intensity from a single fluorophore is typically described as a Gaussian distribution [69] with mean intensity μ and variance σ^2 (Figure 3.1B, inset panel). While intensity fluctuations at low photon counts are better modeled as a Poisson distribution, added signal variance due to rapid fluorophore blinking events, fluctuations in the background signal, and camera read noise justify the assumption that the signal is Gaussian. We postulate that the fluorophores are independent of one another and thus the intensity fluctuations for each fluorophore are uncorrelated with those of neighboring fluorophores. Thus, when *n* fluorophores are localized in a diffraction-limited spot, the overall intensity will be the sum of the mean intensities ($I_{tot} = n*\mu$), and the overall variance will be the sum of the variances plus the variance of the background, δ^2 ($\sigma_{tot}^2 = n*\sigma^2 + \delta^2$). Notably, in photobleaching traces the variance scales with signal intensity, and if background fluctuations are low and/or signal variance is high, then variance is proportional to intensity. This situation contrasts with typical positional step detection problems (for instance, identifying step displacements for motor proteins), where the variance is independent of position and is thus constant for each step [23]. As a result of this scaled variance, with each intensity drop during a photobleaching experiment, there will be an associated decrease in the signal variance.

Another aspect of multi-fluorophore photobleaching data that complicates the identification of bleaching steps is the fact that the frequency of photobleaching events for an ensemble of fluorophores changes over time. Photobleaching is typically modeled as a first order process with rate λ and characteristic bleach time T, where $\lambda = 1/T$. Thus, the time it takes for a single fluorophore in a set to bleach will follow an exponential distribution with mean of T. If there are n fluorophores in a diffraction-limited spot, then the mean time before the first bleaching event will be much faster because any of the fluorophores can bleach. Assuming that photobleaching events are independent of one another, the time before the first bleaching event will also follow an exponential distribution, with a rate equal to $n*\lambda$, and the mean time before the first photobleaching event will be T/n. Thus, at the beginning of an experiment, bleaching events will be more frequent and will be associated with larger signal variance, making it difficult to identify individual events.

To assess the ability of step detection algorithms to detect photobleaching events, we simulated a photobleaching signal for a complex containing 12 GFP fluorophores (Figure 3.1C), each having a mean intensity μ and variance σ^2 that approximated the GFP-AtCESA3 intensity trace shown in Figure 3.1A. In parallel, we simulated a signal having a uniform stepping rate and a constant variance, similar to motor protein displacement signals (Figure 3.1D). Datasets with various SNR values were generated to represent a range of possible experimental scenarios. For motor stepping data (Figure 3.1D), the SNR is defined as ratio of step size over the standard deviation (μ/σ). Defining SNR for bleaching traces, however, is complicated by the fact that the variance changes with the number of active fluorophores. Thus, the SNR for the photobleaching data was defined as the mean intensity μ of a single fluorophore divided by its standard deviation σ, (μ/σ). The variance of the background signal, δ^2, was chosen to equal the variance of a single fluorophore, σ^2. Different SNR values were achieved by setting $\mu = 500$ a.u. and varying the standard deviation. To objectively identify each bleaching event, we developed multiple step detection algorithms that use statistical analysis to detect photobleaching events and compared their performance using the simulated data.

3.3 Using Step Detection Algorithms to Identify Bleaching Events

To analyze our photobleaching data, we developed two step detection algorithms that use statistical tests to identify steps (see Sections 3.8.3 – 3.8.8). For each method, approaches were developed that assumed the different plateau regions in the signal had either equal or unequal variances. The first method is based on the Bayesian Information Criterion (BIC; [70]) and predicts steps purely based on statistical information in the data. Kalafut and Visscher used this approach for step detection previously, but assumed that the variance within each step was constant [62]. We modified this implementation to allow for changes in variance. A second algorithm was developed based on the two-sample t-test with or without assumed equal variance. These four algorithms are named *Bdetector1* and *Bdetector2* for the BIC-based methods assuming equal or unequal variance respectively, and *Tdetector1* and *Tdetector2* for the t-test based methods assuming equal or unequal variance.

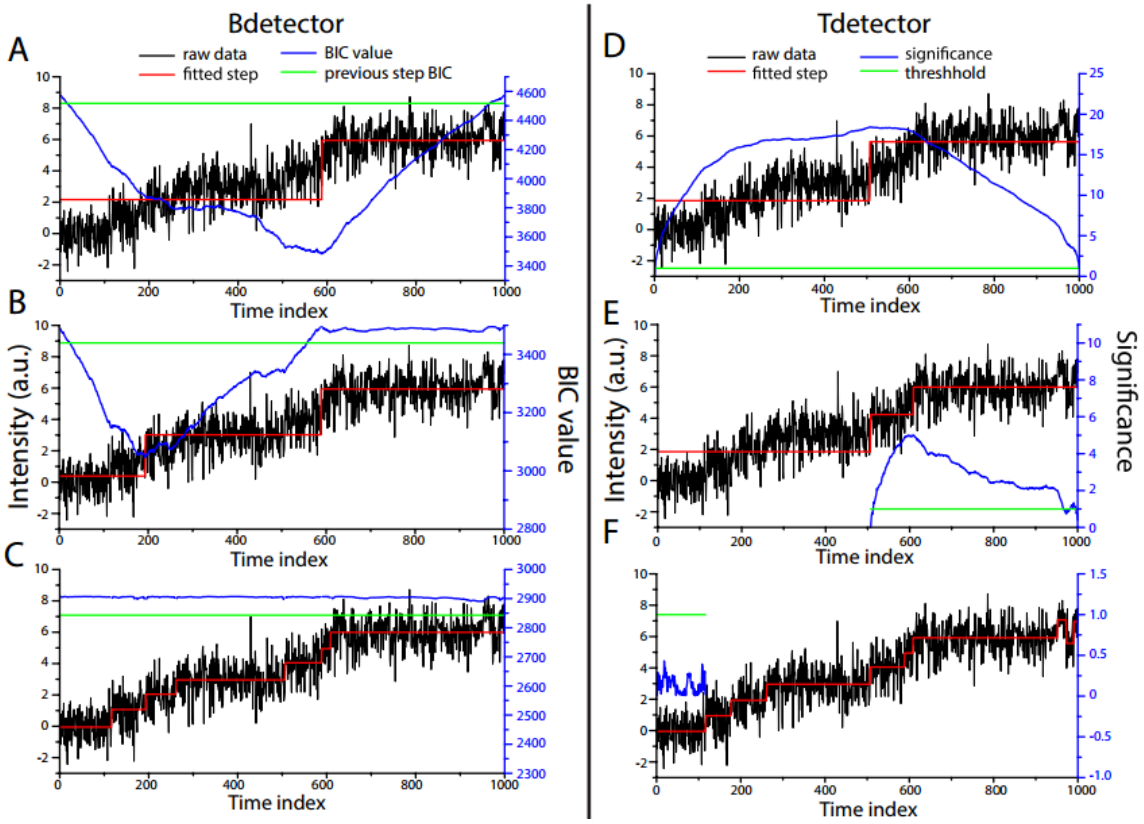

Figure 3.2: Step detection algorithms. (A-C): Bdetector algorithm. (A) To fit the first step, Bdetector scans all possible change points and calculates a corresponding BIC value at each position (blue line). If the minimum BIC is lower than the BIC value for not adding a step (green line), a step is added (red line) at the position where the minimum BIC occurs. (B) Keeping the first step, Bdetector rescans all possible change

points and calculates new corresponding BIC values (blue line), and adds a second step at the position of the minimum BIC (red line). This process is iteratively repeated. (C) When the minimum BIC value for adding an additional step (blue line) is not lower than the current BIC value (green line), the program terminates. (D-F): Tdetector algorithm where, in contrast to the BIC, a higher significance for the t-test indicates a better fit. (D) To add the first step, the significance at each possible change point is calculated (blue line) and is compared to the threshold (green line). Provided it is above the significance threshold, a step is added at the point of maximum significance (red line). (E) The data are split into two segments at the detected change point and the procedure is repeated for each segment (splitting the right segment into two in this case). This process is repeated for each new segment until adding a step does result in a significance value greater than the threshold. The algorithm then moves on to another segment. (F) When adding a change point fails to raise the significance above the threshold for every segment, the program terminates. (Figure from [1], created by Y.C. with assistance from N.C.D.)

Both pairs of algorithms use a conceptually similar step detection approach of iteratively searching for change points until no statistically significant step can be added (Figure 3.2 and Supplemental Movie S2 of [1]). The algorithms are summarized as follows:

(1) The data are scanned, and for each potential time at which a step may occur, the mean and variance is calculated for the time preceding the step and the time following the step.

(2) Using these means and associated variances, a BIC value (Bdetector) or the significance from a two-sample t-test (Tdetector) is calculated and used to identify the optimal step. The optimal step is the one that leads to the lowest BIC value (Bdetector) or the largest significance (Tdetector). If no step leads to a BIC value smaller than the current one or a significance value above a defined threshold then no step is chosen.

(3) The process is repeated until no additional statistically significant steps can be detected, at which point it terminates.

To validate their performance, the step detection algorithms were first tested on simulated stepping data having SNR values from 0.4 to 5 (Figure 3.3). The step times were sampled from an exponential distribution with an expected value of 100 time points per plateau, with 90% of steps being a unit step increase and 10% being a unit step decrease. At high SNR values, the mean predicted step size was close to the actual value, but with diminishing SNR, an additional peak corresponding to twice the unitary step size emerged (Figure 3.3A, and Figure S2 of [1]). We defined two metrics, sensitivity and precision to assess the performance of the algorithms. Sensitivity is defined as the

proportion of the true steps that are identified by the step detection algorithm. Precision is defined as the proportion of identified steps that are true steps (see Section 3.8.10). Overfitting will lead to high sensitivity and low precision (false positives), while underfitting results in high precision but low sensitivity (missed events). With SNR values above 2, all four algorithms performed well and had both high sensitivity and precision values (Figure 3.3, B and C). Reasonable predictions were obtained at SNR values between 1 and 2, but sensitivity and precision both fell sharply for SNR values below 1. The BIC-based algorithms displayed a tradeoff between sensitivity and precision, with Bdetector1 (constant variance) having higher sensitivity and Bdetector2 (unequal variance) having higher precision (Figure 3.3, B and C: blue and green plots). In contrast, for the two-sample t-test methods both Tdetector1 (assumed constant variance) and Tedector2 (assumed unequal variance) performed similarly (Figure 3.3, B and C: red and black plots).

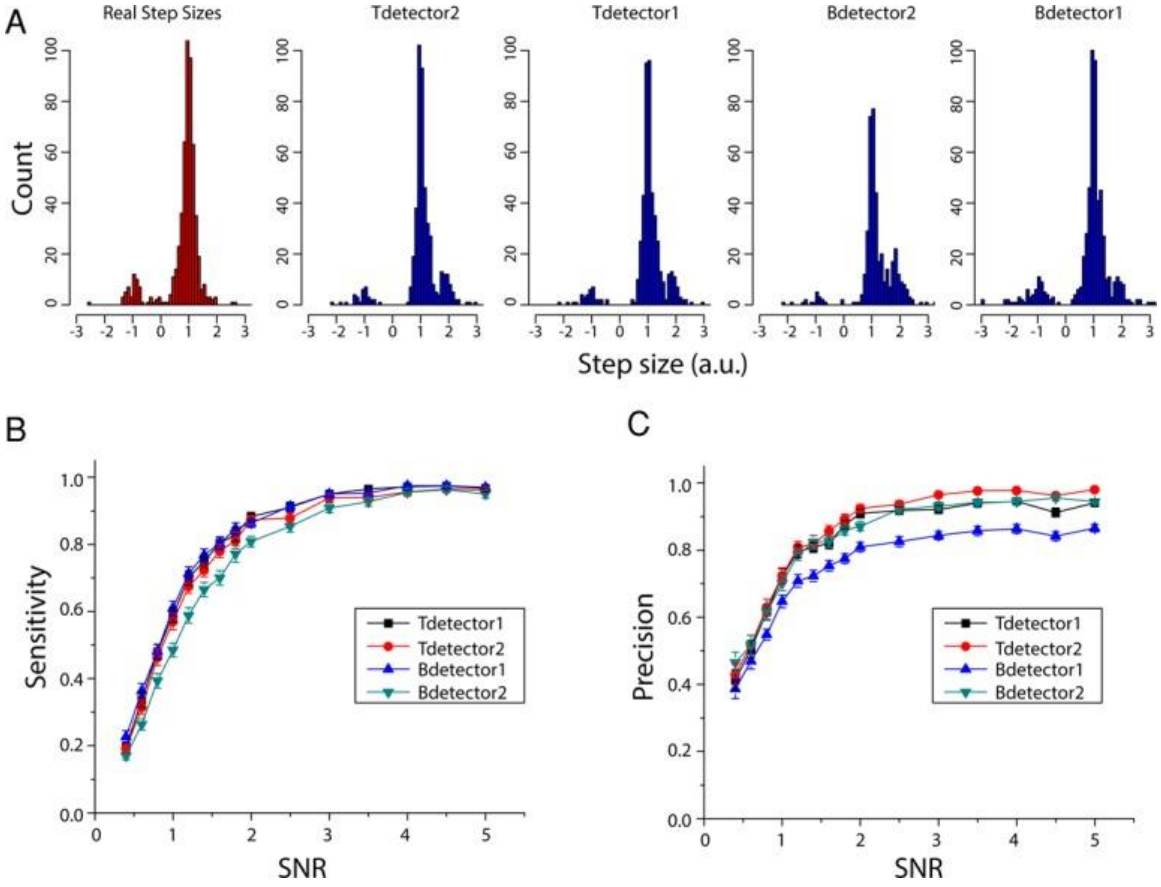

Figure 3.3: Detecting steps in simulated stepping data. (A) Histograms of step sizes predicted by all step detection algorithms. The simulated data have uniform step sizes of 1 with 10% backward steps and SNR

of 1. Real step sizes are calculated by comparing the means of plateau regions on either side of a step. The mode at +1 represents forward steps and the mode at -1 represents backward steps. The four algorithms detect unitary forward and backward steps, but also have modes centered at +2, corresponding to twice the single step size and representing missed steps. (B) Sensitivity plots for the four algorithms. The missed steps corresponding to the lower sensitivity of Bdetector2 can be seen in (A) by the population centered at +2 step size. (C) Precision plots for the four algorithms. Bdetector1 had problems with overfitting, resulting in lower precision and a number of steps between 0 and 1 in (A). (Figure from [1], created by Y.C. with assistance from N.C.D.)

After benchmarking the step detection algorithms on the stepping data, the algorithms were used to detect unitary steps in the simulated photobleaching data. For ease of comparison, the step size was fixed at 500 a.u. for all simulated data and the variance was altered to achieve different SNR values. As seen in Figure 3.4A, both algorithms identified similar steps in the simulated photobleaching data with SNR = 2. Considering the performance at different SNR values, the methods assuming unequal variance (Bdetector2 and Tdetector2) resulted in higher precision but lower sensitivity compared with the methods assuming equal variance (Bdetector1 and Tdetector1, Figure 3.4, B and C).

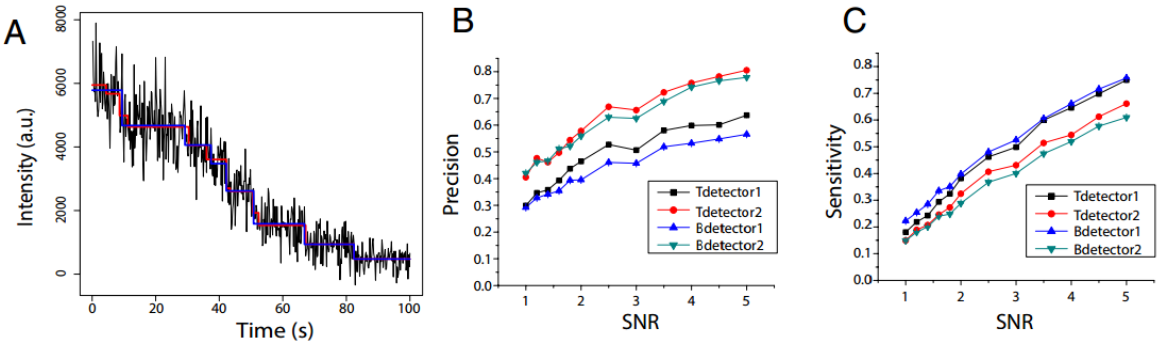

Figure 3.4: Detecting steps in simulated photobleaching data. (A) Simulated photobleaching data (black) with step detection by the Tdetector2 (red) and Bdetector2 (blue) algorithms. (B, C) Precision and sensitivity plots for the four algorithms. The two algorithms not assuming equal variance (Bdetector2 and Tdetector2) gave better precision but missed events, whereas Bdetector1 and Tdetector1 gave better sensitivity but led to false positives. (Figure from [1], created by Y.C. with assistance from N.C.D.)

For estimating subunit numbers from photobleaching data, the most important factor is properly estimating the amplitude of a quantal photobleaching event (the first mode). Hence, a loss in sensitivity corresponding to missed steps (resulting in higher modes) is acceptable. In contrast, the falsely identified steps corresponding to low precision can lead to underestimating the quantal photobleaching amplitude. Based on these considerations, the two methods assuming constant variance were inferior to the methods

assuming unequal variance. The Tdetector2 algorithm performed the best overall and was chosen for the subsequent analyses described below.

3.4 Determining Unitary Step Size from Step Detection Results

After identifying steps, the next task in analyzing photobleaching data is to use the identified step amplitudes to extract the amplitude of a unitary photobleaching event. The total subunit number is subsequently estimated by dividing the initial (high) fluorescence amplitudes by this quantal unit. We initially focused on results from the simulated dataset shown in Figure 3.4A having a SNR = 2 and a GFP copy number of 12. A histogram of step amplitudes predicted by the Tdetector2 algorithm suggests the presence of at least two modes (Figure 3.5A). The simplest method of estimating the unitary step size is to fit the binned histogram data with multiple Gaussian functions corresponding to the different modes. However, estimation by this method is strongly dependent on bin size (Figure 3.5A and B), and there are no existing objective methods for identifying the optimal bin size.

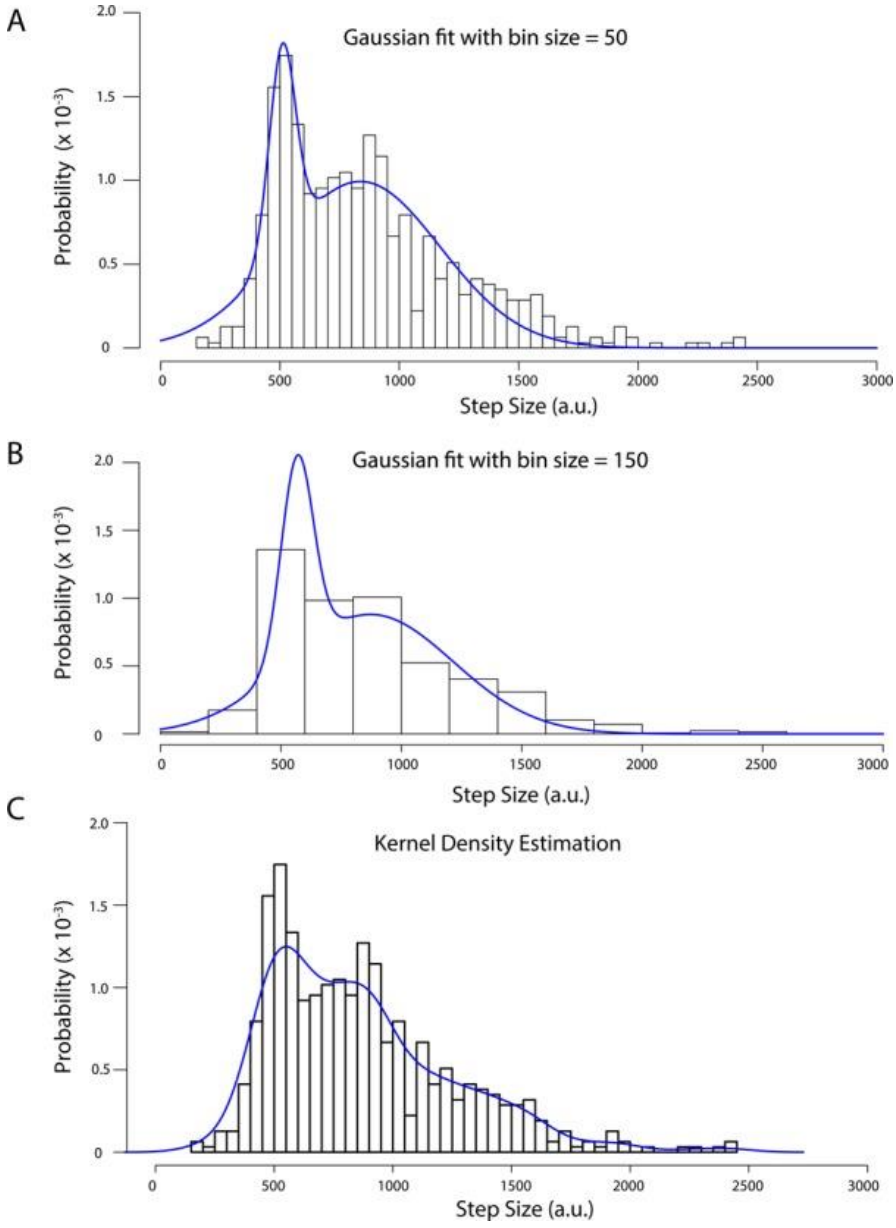

Figure 3.5: Comparing methods of fitting photobleaching step size distributions to extract unitary step size. Histograms represent step size distributions from Tdetector2 applied to simulated photobleaching data with copy number 12 and SNR = 2. The distribution is made up of 570 detected steps. (A) Fit of two Gaussian functions to the data using a bin size of 50. Fit parameters are μ_1 = 510 a.u., σ_1 = 55, μ_2 = 836 a.u., and σ_2 = 335. (B) Fit of two Gaussian functions to the data using a bin size of 150. Fit parameters are μ_1 = 568 a.u., σ_1 = 67, μ_2 = 873 a.u., and σ_2 = 342. In both cases fits to more than two Gaussians did not converge. (C) Identifying modes by KDE. A histogram with bin size 50 is plotted for the purpose of visual comparison but is not used for fitting. Smooth curve is the estimation of multiple Gaussians (kernels) by KDE. (Figure from [1], created by Y.C.)

Kernel Density Estimation (KDE) is a non-parametric method of density estimation that can be used to identify modes without requiring data binning. In short, each step represents a probability of 1/N, where N is total number of steps, and a Gaussian centered

at each step is used to estimate the distribution of this 1/N probability, resulting to a total of N Gaussians. The overall probability density is obtained by the sum of these N Gaussians [71]. Although the main peak from the KDE is obvious, it is difficult to retrieve information for subsequent modes because there are poorly separated (Figure 3.5C).

Density estimation by a Gaussian Mixture Model (GMM) can provide predictions of peak position for each mode in a way that avoids the drawbacks of KDE. In this method the distribution of steps is estimated by a mixture of Gaussians and the means and variances of these Gaussians are obtained by maximizing the expected posterior probability, computationally achieved by expectation–maximization (EM) algorithms [72]. However, one uncertainty of this method is choosing the number of Gaussians (K) to be fit to the data, which can alter the fitting results. To provide an objective method for choosing the number of Gaussians, the step amplitude data were fit using the Gaussian Mixture Model by an increasing number of Gaussians and the Bayesian Information Criterion (BIC) value associated with each fit was determined. The optimal number of Gaussians was defined as the number that gave the lowest BIC value, which for the simulated photobleaching data was 5 (Figure 3.6A and B). The different peaks were assumed to be multiples of the unitary photobleaching amplitude, and the mean unitary step size was calculated as a weighted average of each peak, giving a value of 528.3 a.u. This estimate is within 6% of the step size value of 500 a.u. that was chosen for this simulated photobleaching data.

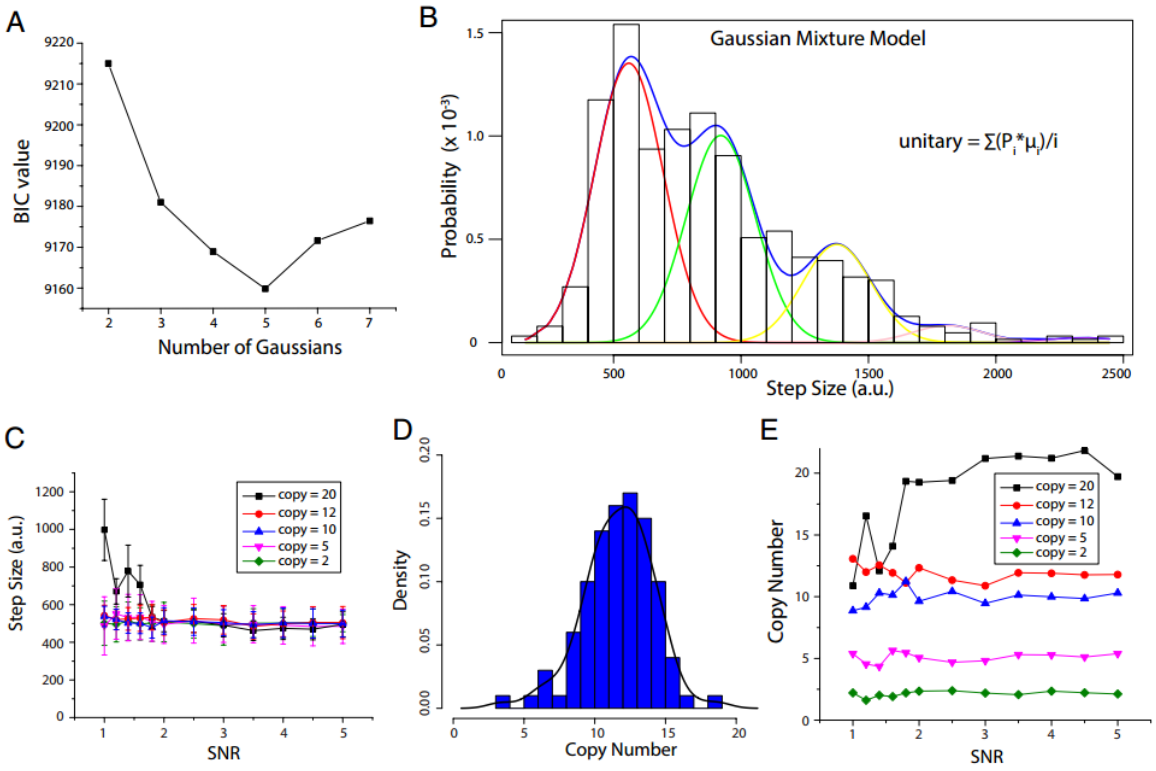

Figure 3.6: Step size and copy number determination for simulated photobleaching data. (A) BIC values using different numbers of Gaussians in the GMM density estimation for the same distribution used in Figure 3.5. The best fit (smallest BIC value) was achieved with 5 Gaussians. (B) Corresponding fit of 5 Gaussians to the step size data (histogram is for display only and is not used by the GMM procedure). Red, green, yellow, pink, and purple traces represent the five Gaussians in the GMM fit, with corresponding means of 560, 921, 1376, 1811, 2343 a.u., and relative weights of 0.461, 0.341, 0.162, 0.028, and 0.008. The standard deviation, which is assumed to be identical for all modes, is 135.9 a.u. Blue line is the overall density. The unitary step size is calculated as $\Sigma_{(i = 1 \text{ to } k)} ((P_i * \mu_i)/i)$, where P_i and μ_i are the relative weight and the mean, respectively, of the i^{th} peak, resulting in a value of 528.3 a.u. (C) Predicted unitary step size as a function of SNR and copy number, demonstrating good performance for copy numbers of below 12 at SNR of 1 and above, and for copy number of 20 at SNR of 2 and above. Actual step size in simulated data was 500 a.u. (D) Predicted copy number from simulated photobleaching data with SNR of 2 and copy number 12. Peak position from KDE (black line) corresponds to mean copy number of 12.3. (E) Predicted copy number across different SNR ratios. Similar to the step size estimate, a break point at SNR below 2 was seen for prediction on copy number 20. (Figure from [1], created by Y.C. with assistance from N.C.D.)

To further assess the performance of this method in estimating copy number from diverse photobleaching data, we performed identical analyses on simulated bleaching data with copy numbers from 2 to 20 at a range of SNR values (Figure 3.6C). Strikingly, for simulated data with copy numbers below 12, the analysis method predicts the value of the unitary step within 10% even down to an SNR of 1 (Figure 3.6C). With a copy number of 20, predicted step sizes are within 7% of the true step size for SNR of 2 and above, but rise toward twice the true step size at lower SNR values. Based on these results, the ability of this method to estimate copy numbers from photobleaching data is limited for

data with both very high copy numbers (20 and above) and low SNR values (below 2). In these cases, the design of the photobleaching experiment should be further optimized to improve the SNR.

3.5 Using Unitary Step Size to Estimate Fluorophore Copy Number

The final task in estimating the number of fluorophores in a complex is to calculate the amplitude of the overall fluorescence drop by taking the difference between the initial fluorescence and the value of the final plateau and dividing by the unitary step size. Accurately estimating the total amplitude of the photobleaching signal can be challenging, however, due to uncertainties in measuring the initial fluorescence amplitude and uncertainties in whether the final plateau represents full bleaching. The first few time points of photobleaching traces have the most variability due to the fast rate of photobleaching and high signal variance associated with a large number of fluorophores. Simply averaging over the first few points reduces the noise but also leads to underestimating the true maximum fluorescence. To avoid introducing any bias, we chose to simply take the initial fluorescence value as the maximum for each trace.

The proportion of fluorophores that are expected to bleach during the finite acquisition time can be estimated by fitting an exponential to the ensemble average of the photobleaching traces (see Section 3.8.9). The simulated photobleaching data had a duration of 100 s and, because it was modeled on the experimental data, was well fit by an exponential with a rate constant of 0.0278 s^{-1}. Thus, 93.9% of the fluorophores are expected to bleach (see Equation 3.1), and the overall intensity drop of the simulated data was corrected upward by dividing by 0.939. Dividing the total intensity drop of each trace by the unitary step size results in a distribution of copy numbers with a mean of 12.3 estimated by KDE (Figure 3.6D), within 3% of the correct copy number of 12. Copy number errors were within 10% for SNR = 1 and above for copy numbers of below 12, and for SNR = 1.8 and above for a copy number of 20 (Figure 3.6E).

3.6 Estimating Copy Number for Kinesin-4XGFP

To validate the ability of the developed methods to estimate copy numbers from a protein with a known number of GFP subunits, we engineered a kinesin construct containing four GFPs (see Section 3.8.1). Proteins were attached to the coverslip surface through non-specific interactions and imaged using TIRF microscopy [28]. Steps were fit to the 71 acquired photobleaching traces using the Tdetector2 algorithm (Figure 3.7A), resulting in 455 detected steps. The step size distribution was fit using the Gaussian Mixture Model and based on the calculated BIC values, the optimal number of modes was determined to be four (Figure 3.7B). When the step size distribution was fit using four modes, the corresponding unitary step size was determined to be 60.8 a.u. (Figure 3.7C). Based on this step size and the standard deviation of noise in the traces, the SNR was calculated to be 1.1 for these measurements.

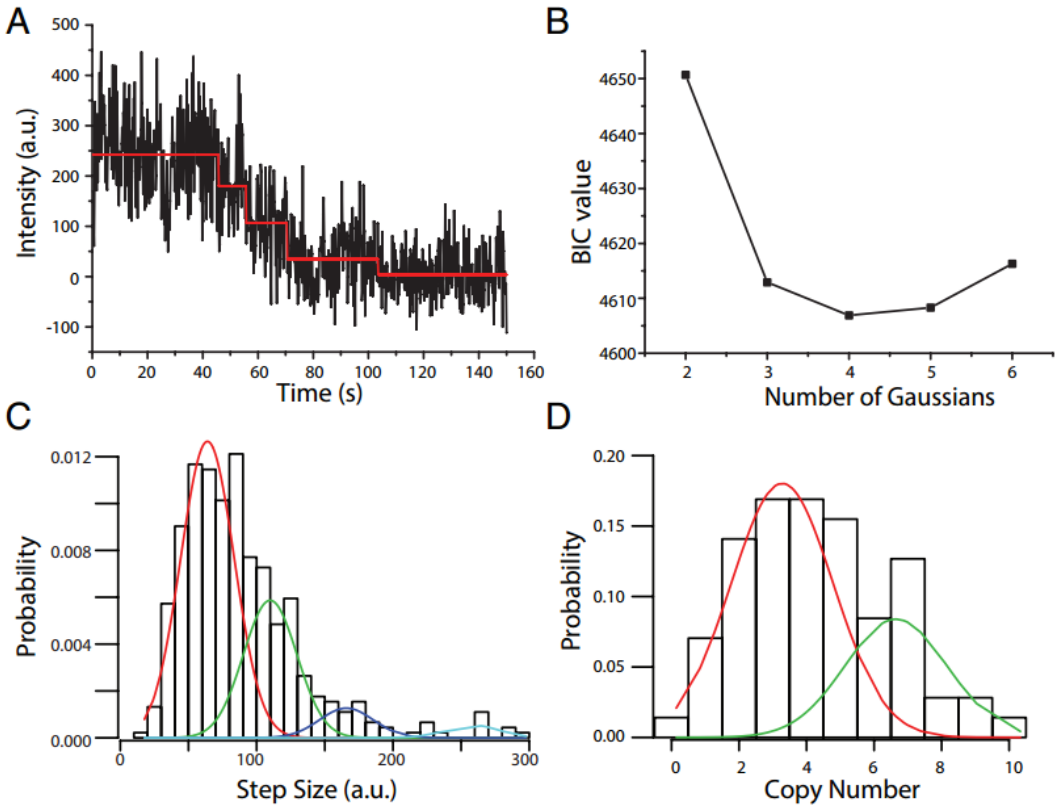

Figure 3.7: Estimating copy number for kinesin-4xGFP. (A) Trace of kinesin-4xGFP bleaching (black) with steps fitted by Tdetector2 (red). (B) The BIC search leads to a best fit of $k = 4$ Gaussians for fitting the step size distribution. (C) Estimating the unitary step size (60.8 a.u.) from the step size distribution (455 total detected steps). The mean values of the four modes were 63.9, 109.9, 165.8, and 258.1 a.u., relative weights were 0.622, 0.289, 0.062, and 0.027, and the SD was 19.6 a.u. (D) Copy number distribution. There were two peaks, centered at 3.28 and 6.65. These peaks are consistent with the binomial nature

leading to a slight shift from four toward lower copy number and with a double-aggregate population at roughly twice the copy number of the first peak. Histograms (black boxes) are also plotted in C and D for reference but not used in the GMM fitting. (Figure from [1], created by Y.C. with assistance from N.C.D.)

The resulting copy number distribution can be influenced by several factors. First, the probability that a GFP will fluoresce is not expected to be unity, which leads to the distribution having a binomial nature. Second, the probability of observing every single bleaching event during an experiment is less than unity due to the finite acquisition time, meaning that the number of acquired bleaching events from each sub-population of fluorescing GFPs will itself be binomially distributed. Third, due to normal intensity fluctuations, the overall intensity drop for each trace will have an associated error value simply from the fluorescence fluctuations. Fourth, it is difficult to rule out the presence of a small percentage of aggregates in the sample or pairs of complexes residing in the same diffraction-limited spot. Due to these factors, the expected copy number distribution will be a binomial distribution broadened by Gaussian noise. As a conservative approach, we chose to fit the copy number distribution using the Gaussian Mixture Model.

To estimate fluorophore copy number, the total intensity drop for each photobleaching trace was calculated by taking the difference of the initial point and the mean value of the final plateau. Each intensity drop was then divided by the estimated unitary step size of 60.8 a.u. to generate a copy number estimate. The fit to the copy number distribution shows two peaks at 3.28 and 6.65 (Figure 3.7D). Given an expected copy number of four, these peaks are consistent with the binomial nature leading to a slight shift towards lower copy number for the first mode, and the second mode corresponding to pairs of complexes either due to aggregates or to two surface-bound complexes being within the same diffraction-limited spot. These results demonstrate that the method can give an accurate prediction of minimum protein copy number even in a data set having a SNR of 1.1.

3.7 Estimating Copy Number for GFP-AtCESA3

After developing an objective method for estimating subunit copy number for protein complexes tagged with large numbers of fluorophores and assessing its performance on simulated photobleaching data, we applied the technique to a set of photobleaching data for GFP-AtCESA3 particles (Figure 3.8A). Based on the trend of BIC values (Figure 3.8B), a model consisting of six Gaussians was used to estimate the distribution of predicted step sizes, and the final estimate for a single step was calculated to be 445.4 a.u. (Figure 3.8C). This step size indicates that the SNR is roughly 2 to 2.5, within the range that our methods can reliably uncover copy number. However, in the final copy number histogram, instead of seeing a single mode as for the simulated data, two modes, one around 10 and the other around 20, are apparent (Figure 3.8D). This factor of two suggests that a subpopulation of the analyzed particles might be composed of two complexes within the focal limited spot, either because there are two populations of CSCs in cells or because pairs of CSCs occasionally exist in close proximity, especially when they are immobile as was the case for this dataset. A fit consisting of two Gaussians identifies peaks at 9.56 and 23.5 copies. Considering that protein misfolding, incomplete maturation of GFP, and bleaching events occurring before data acquisition can all potentially lead to underestimating the true number of GFPs present , we conclude that the 10 copies is a lower limit for the estimated number of GFP-AtCESA3 subunits in each CSC particle.

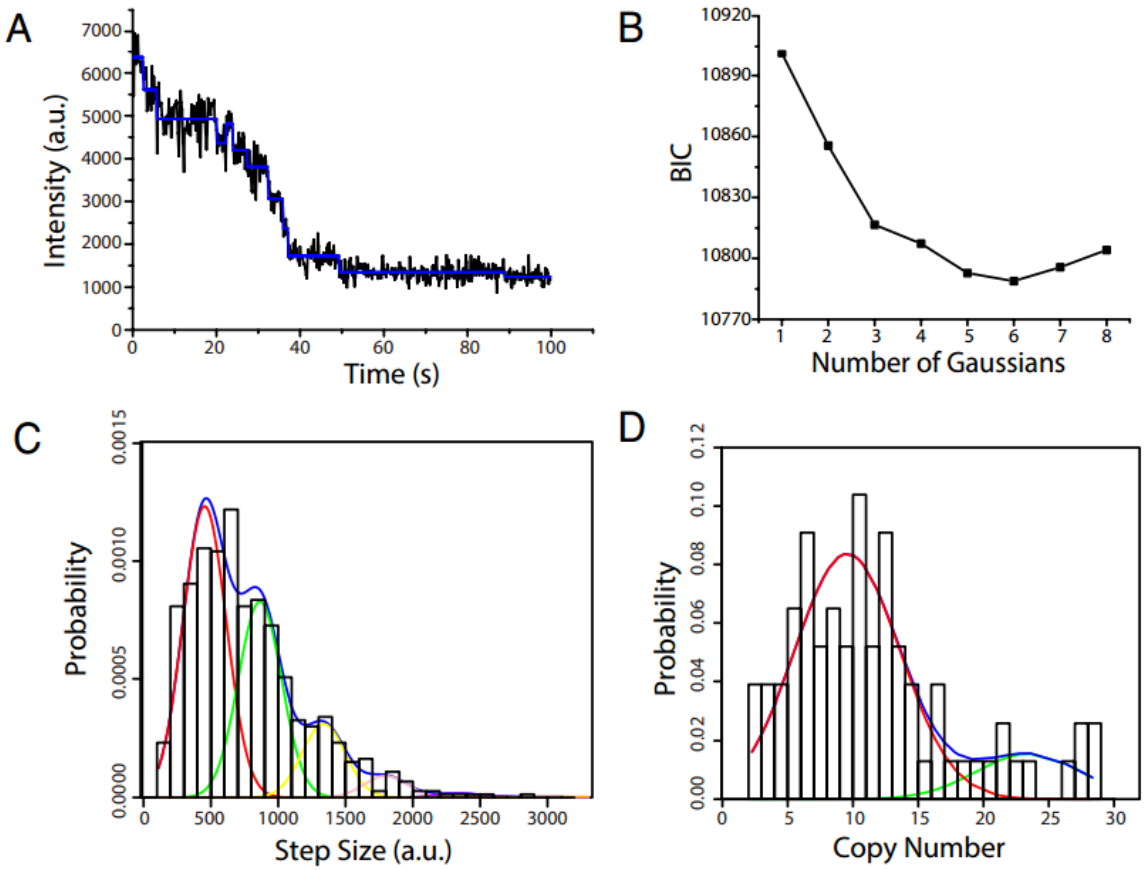

Figure 3.8: Copy number estimation for GFP-AtCESA3 particles. (A) Trace of GFP-AtCESA3 photobleaching (black) with steps fitted by Tdetector2 (blue). (B) BIC values for step detection at increasing number of Gaussians, showing the minimum at $k = 6$. (C) Estimation of unitary step size (445.4 a.u.) by GMM based on 730 total detected steps. Step size distribution was fitted by six Gaussians, shown in red, green, yellow, pink, and purple. Mean values were 453, 864, 1337, 1799, 2335, and 3082 a.u., relative weights were 0.4953, 0.3325, 0.1252, 0.0367, 0.0074, and 0.0027, and the SD was 160 a.u. Overall fit from GMM is shown in blue. Histogram (black boxes) is also plotted for reference but not used in the GMM fitting. (D) Copy number distribution for GFP-AtCESA3 particles. Two peaks are evident from the histograms, and fitting two Gaussians (red and green curves) gives means of 9.56 and 23.5 and ratio of 0.844 and 0.156, with SD of 4.03. (Figure from [1], created by Y.C. with assistance from N.C.D.)

3.8 Materials and Methods

3.8.1 Photobleaching Experiments

Arabidopsis thaliana seeds of the genotype AtCESA3^{je5} GFP-CESA3 [49] were surface-sterilized for 20 min in 30% bleach + 0.1% SDS, washed 4X with sterile water, and stored in sterile 0.15% agar at 4 °C for 3 days before being sown on square petri plates containing MS medium (2.2 g/L Murashige and Skoog salts (Caisson Laboratories, Logan, UT) + 0.6 g/L 2-(*N*-morpholino)-ethanesulfonic acid (MES, Research Organics, Cleveland, OH) + 8 g/L agar-agar (Research Organics), + 10 g/L sucrose, pH 5.6). The plates were incubated in a 22 °C growth chamber under 24h illumination for 5-6 days before use in microscopy experiments. Seedlings were mounted on glass slides between two pieces of permanent double-stick tape (3M, St. Paul, MN), 30 µL of sterile water was added to the seedling, and a 24 x 40 mm #1.5 coverslip was adhered to the tape to generate an imaging chamber. Seedlings were imaged on a Nikon TE2000 microscope in variable-angle mode with a 60X 1.4 NA oil immersion objective and a 100 mW 488 nm excitation laser. Hypocotyl cells containing sparse GFP-AtCESA3-positive particles were imaged using a Photometrics Cascade 512b camera in streaming mode using maximum gain with 200 msec exposure time for 500-600 frames, during which time many particles bleached to background levels.

As a control, *Drosophila* kinesin heavy chain truncated at residue 559 was modified to have GFP at both the N- and C-termini, generating a dimer containing four GFP fluorophores. The protein was bacterially expressed and Ni column purified as previously described [28]. Surface-immobilized fluorophores were imaged by TIRF illumination [28] and acquired in an identical manner to the GFP-AtCESA3 data.

3.8.2 Image Analysis

Image stacks were processed in ImageJ (http://imagej.nih.gov/ij/) as follows. First, the Background Subtract tool (10 pixel radius, sliding paraboloid) was used to subtract background fluorescence from each frame in the stack. Next, an Average Projection of the stack was generated and used to select 7-pixel-radius circular regions of interest (ROI) surrounding immobile GFP-AtCESA3 particles. Finally, photobleaching traces

were generated from the background-subtracted image stack by measuring the total pixel intensity of each ROI for every frame of the stack. A total of 77 particles were analyzed.

3.8.3 Tdetector1 Algorithm

The *Tdetector1* algorithm carries out a modified, iterative two-sample t-test that assumes the expected variance throughout the entire input signal to be constant. As stated previously, it also assumes that the input is a piecewise-constant signal hidden in normally distributed white noise. There are no user-defined parameters, and the only input to the algorithm is the signal in the form of a vector or series of values, X.

To begin, the algorithm must calculate the variance of the corrupting white noise, σ^2, of the input signal. The conventional method of calculating variance ($\mathrm{Var}(X) = \mathrm{E}[(X-\mu)^2]$) cannot be used because the data is expected to contain steps that would result in a large overestimation of the corrupting variance. Instead a pairwise difference calculation must be used (Equation 3.1). Pairwise differences that are significantly greater in magnitude compared to the rest (possibly due to a large step there) are discounted from the calculation (see Section 3.8.5 for further details).

$$\sigma^2 \approx \frac{\sum_i^{(L-1)}(x_{i+1}-x_i)^2}{2(L-1)} \qquad (3.1)$$

Where X = input signal, σ^2 = variance of corrupting noise in X, L = length of X, i = index of X.

The first round of the step detection process iterates through every possible way of splitting X into two sections and calculates the difference of means (DOM) of those two sections. Each DOM is then rated for significance based on the expected distribution of DOMs that would result from splitting a normal random vector of the same length, with no steps, at that respective index (given in Equation 3.2). This process is similar to comparing to the t-distribution as in a two-sample t-test.

$$\mathrm{DOMs} \sim \mathrm{N}\left(0, \sigma^2\left(\frac{1}{i}+\frac{1}{L-i}\right)\right) \qquad (3.2)$$

Where σ^2 = variance of corrupting noise in X, L = length of current subset of X (for first round of step detection: L = length of entire X series), i = index of splitting.

If there is a calculated DOM that is significant (see Section 3.8.6) compared to the normal distribution shown in Equation 3.2, then the null hypothesis – that the observed DOM is due to variations of a normal random vector without a step – is rejected, the two sections are declared as two separate plateaus, and a possible step is declared at that index. For each round of step detection, only the most significant DOM results in a declared step. After the first round of step detection, the process is repeated on each new plateau, and any new plateaus from a round of step detection will go through the same process until no new plateaus are declared.

Finally, the algorithm undergoes a step-checking phase that performs DOM significance testing for all adjacent plateaus declared. MATLAB code for the Tdetector algorithm is included in Appendix A.

3.8.4 Tdetector2 Algorithm

The *Tdetector2* algorithm is nearly identical to Tdetector1, except that it assumes that different sections of the data have different expected variances (as found in photobleaching traces where higher numbers of unbleached fluorophores lead to higher variances). Again, it assumes the input is a piecewise constant signal hidden in normally distributed white noise, and it requires only a single series of data, X, as input to the algorithm.

The first task of the algorithm is to find sections of the data that have significantly different variances from one another. To accomplish this, it first calculates the variance of corrupting noise throughout all of X using the same process described for Tdetector1 (Equation 3.1). Next, it uses the same process that the Tdetector1 algorithm uses to test each possible DOM for significance, but instead of comparing means it tests each possible difference of variances (DOV) for significance. The expected distribution of DOVs is approximated as normal, with a variance (Equation 3.3, derivation in Section 3.8.7) that depends on nearly the same variables defining the variance of DOMs in Equation 3.2. The only difference is that σ^2 is always the corrupting variance of the entire X vector in Equation 3.2, while in Equation 3.3 it is the corrupting variance of only the subset of X that is currently being split into two sections.

$$\mathrm{DOVs} \sim \mathrm{N}\left(0, \sigma^4 \left[\frac{i^2+i-3}{(i-1)^2} + \frac{(L-i)^2+(L-i)-3}{((L-i)-1)^2} - 2\right]\right) \qquad (3.3)$$

Where σ^2 = variance of corrupting noise in current subset of X, L = length of current subset of X, i = index of splitting.

As in the iterative step fitting process of Tdetector1, this variance-sectioning continues to declare and test new plateaus until no new significant variance sections are declared. Once the algorithm has completed the variance-sectioning process, it begins the same step detection process as in the Tdetector1 algorithm, with two exceptions: (1) For DOM significance testing, Tdetector2 uses σ^2 = mean corrupting variance of the current subset of X in Equation 3.2 rather than the corrupting variance of the entire X series; and (2) Once the most significant index of splitting has been determined, the resulting DOM is again tested for significance with respect to a slightly different distribution of DOMs shown by Equation 3.4 (similar to Welch's t-test [73]). This distribution takes into account the possibility of unequal variances between the two sections. If both tests have shown significance with respect to their distributions, then a step and two new plateaus are declared at that index.

$$\mathrm{DOMs} \sim \mathrm{N}\left(0, \frac{{\sigma_1}^2}{i} + \frac{{\sigma_2}^2}{L-i}\right) \qquad (3.4)$$

Where ${\sigma_1}^2$ = corrupting variance of the first section, ${\sigma_2}^2$ = corrupting variance of the second section, L = length of current subset of X, i = index of splitting.

3.8.5 Calculation of Variance of Corrupting Noise

Let X be a vector of L independent random variables with a mean of 0, and variance of σ^2. Let Y be a piecewise-constant vector of L values, containing a step of amplitude d between indexes i and i+1. Now let the sum of these two vectors, $Z = X + Y$, represent a data vector given to the Tdetector step detection algorithm (see Figure 3.9).

$$X = [x_1, x_2, \dots, x_{L-1}, x_L], Y = [0,0, \dots, d, d], Z = [x_1, x_2, \dots, x_{L-1} + d, x_L + d]$$

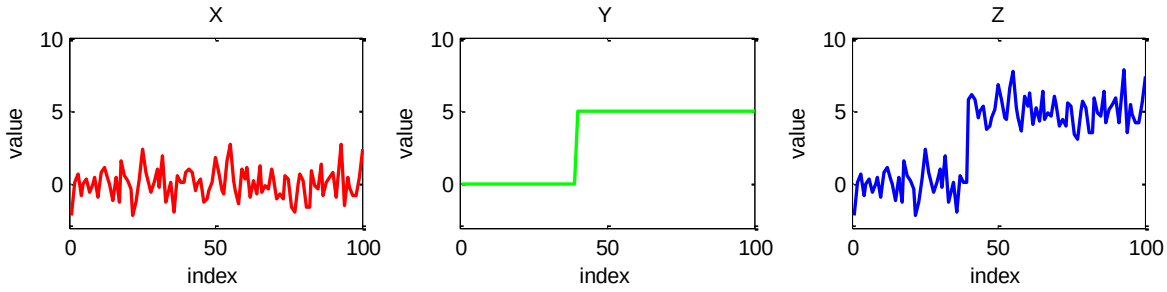

Figure 3.9: Plots of theoretical X, Y, Z vectors where $\sigma^2 = 1$, d = 5, L = 100, and i = 40

The goal is to estimate σ^2 (the variance of the corrupting noise, X), but we are given only the vector Z. Using the conventional calculation of variance on Z would yield an answer composed of both σ^2 and the value of d (step amplitude of Y).

$$\mathrm{Var}(Z) = \mathrm{Var}(X + Y)$$

$$= \mathrm{Var}(X) + \mathrm{Var}(Y)$$

$$= E[(X - E[X])^2] + E[(Y - E[Y])^2]$$

$$= \sigma^2 + \frac{i(L-i)}{L^2} d^2$$

If Z contained more than one step, $\mathrm{Var}(Z)$ would be an even greater overestimation of σ^2. Therefore, a method aimed at calculating the variance of only the corrupting noise – a pairwise difference calculation – should be used instead. Generally speaking, it calculates variance based on the difference between neighboring data points rather than the difference of each data point from the mean. The following demonstrates how one-half of the expected value of squared pairwise differences of X equates to the variance of X, σ^2.

$$\frac{\sum_{n=1}^{(L-1)} (x_{n+1} - x_n)^2}{2(L-1)} = \frac{E[(x_{n+1} - x_n)^2]}{2} = \frac{E[x_{n+1}^2 - 2x_{n+1}x_n + x_n^2]}{2}$$

Since X is an independent random vector with a mean of zero:

$$= \frac{E[x_{n+1}^2] - 2E[x_{n+1}]E[x_n] + E[x_n^2]}{2} = \frac{E[x_{n+1}^2] + E[x_n^2]}{2} = \frac{\sigma^2 + \sigma^2}{2} = \sigma^2$$

This yields Equation 3.1 given in the Tdetector1 Algorithm Section 3.8.3:

$$\mathrm{Var}(X) = \frac{\sum_{n=1}^{(L-1)} (x_{n+1} - x_n)^2}{2(L-1)}$$

This equation holds only if all values in X have an expected value of zero. If it is instead applied to Z, a piecewise constant step function hidden in noise, then the equation does not give $\mathrm{Var}(Z)$, but rather a value composed of the variance of corrupting noise and a relatively small contribution from d (step amplitude of Y).

$$\frac{\sum_{n=1}^{(L-1)} (z_{n+1} - z_n)^2}{2(L-1)} = \sigma^2 + \frac{1}{2(L-1)} d^2$$

As is, this approach yields a much better estimate of the variance of corrupting noise than simply using the variance of Z (when $L \geq 4$). However, an even better estimation of σ^2 can be obtained by performing an iterative outlier analysis on the pairwise difference values of Z before taking their mean. If the magnitude of any pairwise difference is significantly greater than the rest, then we can hypothesize that it is due to a step in the data vector, consider it an outlier, and therefore exclude it from the average. More specifically, if its magnitude is greater than three times the standard deviation of pairwise differences of X ($\sqrt{2}\sigma$) then it should be excluded. Of course we do not know the value of σ, so we use the current best estimate. This process is iterated until there are no outliers remaining. Iterations are necessary because each time an outlier is removed, the value of σ changes slightly. The following Table 3.1 describes the iterative process explicitly.

Table 3.1: Pseudo/MATLAB code of iterative pairwise difference outlier removal

```
L = length(Z);

% construct pairwise differences of Z vectors
for i = 1:L-1
    pdz(i) = Z(i+1) - Z(i);
    pdz2(i) = (Z(i+1) - Z(i))^2;
end

while true
    % current estimate of sigma of X
    sigmaC = (mean(pdz2)/2)^0.5;

    % remove outlier values from pdz vectors
    pdz2(abs(pdz) > 3*(2^.5)*sigmaC) = [];
    pdz(abs(pdz) > 3*(2^.5)*sigmaC) = [];

    % new estimate of sigma of X
    sigmaN = (mean(pdz2)/2)^0.5;

    if sigmaN == sigmaC
        break
    end
end

% final sigma estimate
sigma = sigmaN;
```

3.8.6 Difference of Means Significance Testing

A difference of means (DOM) is declared significant by the Tdetector algorithm if its absolute value is greater than a certain value (the *multiplier*) times the standard deviation of its respective DOM distribution (Equation 3.2). The multiplier determines the frequency of incorrect rejections of the null hypothesis (i.e. false positives). For a given data series of length, L, there are L-1 ways to split the data into two sections, hence that many DOM values being tested for significance (i.e. opportunities for a false positive to occur).

We want the probability that a given data vector will return a false positive to be 0.05, but choosing the corresponding multiplier is analytically difficult due to the fact that DOM values are not independent of one another. If they were independent, the relation would be simple; given L-1 opportunities for a false positive, the probability, p, that a single DOM should yield a false positive must be:

$$p = 1 - (0.95)^{\frac{1}{L-1}}$$

The normal distribution standard deviation multiplier (as a function of L) that would yield this probability can be calculated using the inverse error function as follows.

$$\text{multiplier}(L) = -\sqrt{2}\,\text{erfinv}\left(-(0.95)^{\frac{1}{L-1}}\right)$$

This relation was used as guidance for estimating multiplier values empirically. Multiplier values in the range of this relation were tested on several generated random vectors of different lengths L in order to achieve a 0.05 false positive probability. The resulting empirical multiplier lookup table is as follows (Table 3.2).

Table 3.2: Empirically calculated standard deviation multiplier lookup table for DOM significance testing. Data vector lengths, L, are rounded values of $2^{(n/2)}$ where n = 0,1,2, …, 26. Multipliers between given L values can be linearly interpolated with good reliability. The last two L values in the table are untested extrapolations of the trend.

L	**multiplier**		**L**	**multiplier**
1	0.0000		181	3.1207
2	1.9600		256	3.1500
3	2.1700		362	3.1975
4	2.3400		512	3.2400
6	2.4700		724	3.2801
8	2.6000		1024	3.3048
11	2.6563		1448	3.3183
16	2.7500		2048	3.3252
23	2.8156		2896	3.3295
32	2.9000		4096	3.3311
45	2.9406		5793	3.3328
64	3.0000		8192	3.3332
91	3.0422		10000	3.3333
128	3.1000		1e+10	3.3333

3.8.7 Differences of Variances

Let X be a vector of L independent normally distributed random variables with a mean of 0, and variance of σ^2.

$$X = [x_1, x_2, x_3, \dots, x_L]$$

Next, if X is split into two sections, X_A and X_B, of length N and M respectively,

$$X_A = [x_1, x_2, x_3, \dots, x_N], \; X_B = [x_{N+1}, x_{N+2}, x_{N+3}, \dots, x_{N+M}]$$

Then both $X_A$and $X_B$will have their own sample variance. The difference of these two variances is referred to here as the DOV.

$$\mathrm{DOV} = \mathrm{Var}(X_A) - \mathrm{Var}(X_B)$$

If this process was repeated on many randomly generated X vectors of length L, split into two sections of lengths N and M, and a DOV was calculated each time, then the resulting collection of DOVs would have a variance itself.

$$\mathrm{Var}(\mathrm{DOV}) = \mathrm{Var}\big(\mathrm{Var}(X_A) - \mathrm{Var}(X_B)\big)$$

We wish to know $\mathrm{Var}(\mathrm{DOV})$ in order to test for the significance of a given DOV calculated from a data vector. Even though $X_A$and X_B as we have stated in this derivation are not expected to contain steps, we must represent their variance with the pairwise difference method (Equation 3.1) because that is how variance values for DOV of a given data vector will be calculated (see Appendix C for more detail).

$$\mathrm{Var}(\mathrm{DOV}) = \mathrm{Var}\left(\frac{\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2}{2(N-1)} - \frac{\sum_{n=N+1}^{(N+M-1)}(x_{n+1} - x_n)^2}{2(M-1)}\right)$$

These two terms, $\mathrm{Var}(X_A)$ and $\mathrm{Var}(X_B)$, are independent of one another, therefore:

$$\mathrm{Var}(\mathrm{DOV}) = \mathrm{Var}\left(\frac{\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2}{2(N-1)}\right) + \mathrm{Var}\left(\frac{\sum_{n=N+1}^{(N+M-1)}(x_{n+1} - x_n)^2}{2(M-1)}\right)$$

We can simplify the variances above, $\mathrm{Var}\big(\mathrm{Var}(X_A)\big)$ and $\mathrm{Var}\big(\mathrm{Var}(X_B)\big)$, to functions of the population variance of X, σ^2, that depend on lengths N and M respectively, using the

conventional formula ($\mathrm{Var}(X) = \mathrm{E}[(X - \mathrm{E}(X))^2] = \mathrm{E}[X^2] - (\mathrm{E}[X])^2$). The simplification of $\mathrm{Var}\big(\mathrm{Var}(X_A)\big)$ is as follows.

$$\mathrm{Var}\left(\frac{\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2}{2(N-1)}\right)$$

$$= \mathrm{E}\left[\left(\frac{\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2}{2(N-1)}\right)^2\right] - \left(\mathrm{E}\left[\frac{\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2}{2(N-1)}\right]\right)^2$$

$$= \mathrm{E}\left[\left(\frac{\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2}{2(N-1)}\right)^2\right] - (\sigma^2)^2$$

$$= \mathrm{E}\left[\left(\frac{1}{2(N-1)}\right)^2\left(\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2\right)^2\right] - \sigma^4$$

$$= \left(\frac{1}{2(N-1)}\right)^2 \mathrm{E}\left[\left(\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2\right)^2\right] - \sigma^4$$

$$= \left(\frac{1}{4(N-1)^2}\right) \mathrm{E}\left[\left(\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2\right)^2\right] - \sigma^4$$

Next, we can simplify the term highlighted in blue to a multiple (defined by length N) of the squared population variance of X, σ^4.

$$\mathrm{E}\left[\left(\sum_{n=1}^{(N-1)}(x_{n+1} - x_n)^2\right)^2\right] = \mathrm{E}\left[\left(x_1^2 - 2\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right) + 2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right) + x_N^2\right)^2\right]$$

$$= E\left[\begin{array}{c} x_1^4 - 2x_1^2\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right) + 2x_1^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right) + x_1^2 x_N^2 \ldots \\ -2x_1^2\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right) + 4\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right)^2 - 4\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right)\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right) - 2x_N^2\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right) \ldots \\ +2x_1^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right) - 4\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right)\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right) + 4\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right)^2 + 2x_N^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right) \ldots \\ +x_1^2 x_N^2 - 2x_N^2\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right) + 2x_N^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right) + x_N^4 \end{array}\right]$$

When expanded, the expected value of each term within all red terms will be equal to zero. This is because each term will contain at least one value raised to the first power (x_n^1), which has an expected value of zero, resulting in the expected value of that entire term being equal to zero. Therefore all red terms above can be dropped.

$$= E[x_1^4] + 2E\left[x_1^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right)\right] + E[x_1^2 x_N^2] + 4E\left[\left(\sum_{n=1}^{(N-1)} x_n x_{n+1}\right)^2\right]$$
$$+ 2E\left[x_1^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right)\right] + 4E\left[\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right)^2\right] + 2E\left[x_N^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right)\right]$$
$$+ E[x_1^2 x_N^2] + 2E\left[x_N^2\left(\sum_{n=1}^{(N-2)} x_{n+1}^2\right)\right] + E[x_N^4]$$

Recall: $\sigma^2 = E[x_n^2] - E[x_n]^2 = E[x_n^2]$, and $E[x_n^2 x_m^2] = E[x_n^2]E[x_m^2] = \sigma^4$ since $x_n \perp x_m$ where $n \neq m$

Note: the 4th central moment of a normal random variable, $E[x_n^4] = 3\sigma^4$

$$E\left[\left(\sum_{n=1}^{(N-1)} (x_{n+1} - x_n)^2\right)^2\right]$$
$$= 3\sigma^4 + 2(N-2)\sigma^4 + \sigma^4 + 4(N-1)\sigma^4 + 2(N-2)\sigma^4$$
$$+ 4[(N-2)^2 - (N-2) + 3(N-2)]\sigma^4 + 2(N-2)\sigma^4 + \sigma^4 + 2(N-2)\sigma^4 + 3\sigma^4$$
$$= (4N^2 + 4N - 12)\sigma^4$$

This term can now be plugged back in for the blue highlighted term:

$$\mathrm{Var}\left(\frac{\sum_{n=1}^{(N-1)}(x_{n+1}-x_n)^2}{2(N-1)}\right)=\left(\frac{1}{4(N-1)^2}\right)E\left[\left(\sum_{n=1}^{(N-1)}(x_{n+1}-x_n)^2\right)^2\right]-\sigma^4$$

$$=\left(\frac{1}{4(N-1)^2}\right)(4N^2+4N-12)\sigma^4-\sigma^4$$

$$=\left(\frac{N^2+N-3}{(N-1)^2}\right)\sigma^4-\sigma^4$$

$$=\left(\frac{N^2+N-3}{(N-1)^2}-1\right)\sigma^4$$

Now this term for $\mathrm{Var}(\mathrm{Var}(X_A))$ and its counterpart representing $\mathrm{Var}(\mathrm{Var}(X_B))$ in terms of M can be plugged into the $\mathrm{Var}(DOV)$ equation.

$$\mathrm{Var}(DOV)=\mathrm{Var}\left(\frac{\sum_{n=1}^{(N-1)}(x_{n+1}-x_n)^2}{2(N-1)}\right)+\mathrm{Var}\left(\frac{\sum_{n=N+1}^{(N+M-1)}(x_{n+1}-x_n)^2}{2(M-1)}\right)$$

$$=\left(\frac{N^2+N-3}{(N-1)^2}-1\right)\sigma^4+\left(\frac{M^2+M-3}{(M-1)^2}-1\right)\sigma^4$$

$$=\left[\frac{N^2+N-3}{(N-1)^2}+\frac{M^2+M-3}{(M-1)^2}-2\right]\sigma^4$$

Instead of using N and M, we can represent the lengths of X_A and X_B with i and $(L-i)$ respectively. This yields the variance shown in Equation 3.3.

$$\mathrm{Var}(DOV)=\left[\frac{i^2+i-3}{(i-1)^2}+\frac{(L-i)^2+(L-i)-3}{((L-i)-1)^2}-2\right]\sigma^4$$

3.8.8 Bdetector Algorithms

The Bdetector1 algorithm is identical to the method described in [62], with the algorithm implemented in R (http://www.r-project.org). The Bdetector2 algorithm was developed by modifying Bdetector1 to allow for changing variance, as follows:

For a data with points x_i (i is from 1 to N), if k steps are fitted at position l_1, l_2, … , l_k, and for notational simplicity, let $l_0 = 0$, and $l_{k+1} = N$, then the maximum likelihood estimator for mean and variance are:

$$\mu_j = \frac{1}{l_j - l_{j-1}} \sum_{i=l_{j-1}}^{l_j} x_i$$

Where j = 1, …, k + 1

$$\sigma_j^2 = \frac{1}{l_j - l_{j-1}} \sum_{i=l_{j-1}}^{l_j} (x_i - \mu_j)^2$$

Recall that the BIC for a statistical model is calculated as:

$$BIC = -2 * logL + p * ln(N)$$

Where *logL* is the log-likelihood of a model, and *p* is the number of parameters to estimate.

Thus, the BIC for fitting k steps will be:

$$BIC = \sum_{j=1}^{k+1} (l_j - l_{j-1}) * ln(\sigma_j^2) + N * ln(2\pi) + N + p * log(N)$$

Where $p = 2*(k+1) = 2k + 2$.

To add a step, Bdetector2 scans each potential step position and calculates a BIC value. If the difference between the minimal BIC value and BIC from not adding a step is greater than 5 [74] a new step is added at the position that leads to smallest BIC value. While holding all previous steps, this process is then repeated to detect subsequent steps. Bdetector2 terminates when no more steps that result in a lower BIC value can be added.

3.8.9 Photobleaching Rate Estimation

By ensemble averaging many photobleaching traces and fitting to an exponential, the photobleaching rate constant can be estimated with high accuracy. Because each GFP photobleaches independently of one another, the rate constant for the exponential decay of the ensemble average will be the same as the first-order bleaching rate of a single GFP.

Comparing the photobleaching rate constant to the total acquisition time also allows for a correction due to photobleaching events that are expected to be missed due to the finite acquisition time of the experiment. Based on the known acquisition time and calculated photobleaching rate, Equation 3.1 calculates the fraction of photobleaching events that are expected to occur during acquisition. This number is critical because the final copy number is estimated by dividing the total intensity drop for each photobleaching trace by the experimentally-determined unitary step size. If the photobleaching trace has not fallen all the way to background, then copy number will be underestimated. Hence, to correct for missed photobleaching events, the total intensity drop for each trace is corrected by dividing by the expected fraction of observed events given by:

$$\text{Fraction observed} = 1 - e^{-ak} \qquad (3.1)$$

Where a = acquisition time in seconds, k = fitted photobleach rate in inverse seconds

According to our fitted photobleaching rate (0.0278 ± 0.0003 s-1) and acquisition time (a = 100 s), we expect to observe ~ 93% of the photobleaching process.

3.8.10 Definition of Sensitivity and Precision for Step Detection Algorithms

The ability of each step detection algorithm to correctly identify steps was tested using simulated data with added white noise containing steps at known indexes. Each algorithm was given the same collection of simulated data, and then the indexes at which each algorithm declared steps were compared to the true step indexes. If a declared step index was within a certain range of a true step index, then it was regarded as a correct declared step (i.e. if Equation 3.2 is satisfied). The range was defined by a constant percentage multiplier (0.05) of the two true plateau lengths on either side of a true step index.

$$-\|0.05\ p_1\| \leq (i_{\text{declared}} - i_{\text{true}}) \leq \|0.05\ p_2\| \qquad (3.2)$$

Where p_1 = # of data points in plateau that precedes the true step, p_2 = # of data points in plateau that follows the true step, $i_{declared}$ = index of the declared step, i_{true} = index of a true step.

Once a declared step is defined as correct, the true step to which it was matched is no longer allowed to be matched to again. This means that if there are multiple declared steps within a certain range of the true step, only one of those declared steps is allowed to be defined as correct.

The sensitivity of an algorithm was defined as the fraction of true steps that have a declared step within their range (detected true steps). The precision of an algorithm was defined as the fraction of declared steps that are correct:

$$\text{sensitivity} = \frac{\text{detected true steps}}{\text{total true steps}}$$

$$\text{precision} = \frac{\text{correct declared steps}}{\text{total declared steps}}$$

Underfitting the data will result in low sensitivity and generally higher precision, while overfitting will result in low precision and generally higher sensitivity.

3.8.11 Density Estimation

Least-squares fitting on binned histogram data was carried out in R with nonlinear least-squares fitting. Center of bins and bin height are used. For Kernel Density Estimation, bandwidth is as specified by Scott [75]. The "normalmixEM" function in the R package "mixtools" [76] was used to implement the Gaussian Mixture Model, and the variance of each Gaussian was assumed to be the same while means were unconstrained. The BIC value, is calculated based on the log-likelihood of each fitting, and was used to objectively determine the number of Gaussians to use in the final model.

3.9 Discussion

Determining the stoichiometry of proteins in large multi-subunit membrane complexes by biochemical methods is challenging, and despite producing a highly abundant and useful biopolymer, the molecular makeup of the cellulose synthesis complex, one such protein complex, has remained enigmatic. The goal of this work was to quantify the number of CESA subunits in cellulose synthesis complexes by non-destructive in vivo photobleaching. Plant seedlings expressing GFP-AtCESA3 were imaged using variable-angle epifluorescence microscopy and the fluorescence intensities of individual GFP-AtCESA3-containing particles were recorded as the signals bleached to near background levels. However, despite efforts to maximize the SNR, individual photobleaching steps were not easily identified by eye, preventing an objective estimate of CESA copy number. This hurdle motivated us to develop a set of statistical tools to estimate unitary step size and fluorophore copy number from photobleaching data involving many fluorophores.

Using imaging to quantify subunit copy number for intact protein complexes in vivo provides a method to probe the quaternary structure of these complexes that circumvents the difficulty and potential disruption of the complex inherent in biochemical purification. For copy numbers under five, it is often easy to simply estimate the number of steps by eye [58, 77]. In other cases, it is possible to estimate unitary step intensity by measuring the amplitude of the last step, but that approach ignores much of the rich information present in the data. Because small errors in the estimation of the unitary step intensity can propagate to larger errors in the copy number estimation, it is important to use as much of the available information as possible to achieve the best possible estimate for unitary photobleaching. In our photobleaching data analysis, we identified three major challenges to accurately measuring high copy numbers: 1) detecting steps in traces having non-uniform variances due to the summed fluctuations of multiple fluorophores, 2) precisely identifying the unitary step size from step size distribution densities, and 3) accurately quantifying the total intensity drop corresponding to bleaching for all of the subunits in the complex. We developed a solution for each of these challenges, and we hope that this set of tools will be adopted as “best practices” for analyzing photobleaching data in other systems with high protein copy number.

While signal variance in molecular motor stepping data is independent of the motor position, photobleaching data present the unique challenge of signal variance that scales with intensity. Previous step detection methods have used the approach of constructing pairwise distance distributions to estimate unitary step size for each step [23, 59], but assumed a constant variance. This variance is important because it is used in tests to determine statistical significance. Applying step detection algorithms that assume constant variance to photobleaching data results in overfitting of steps in early time points when both the signal and variance are high. Thus the technique developed here to estimate the time-dependent variance of the signal was a key advance that improved the performance of both the BIC-based and t-test-based step detection algorithms over those assuming constant variance.

The step detection algorithms output a step size distribution density that needs to be analyzed to extract the unitary step size. We found Kernel Density Estimation to be a vastly superior approach over the traditional technique of binning the data and fitting multiple Gaussians because it eliminated the decision of what bin size to use. However, one weakness of KDE was fitting to higher modes. The Gaussian Mixture Model proved to be the optimal tool for identifying the modes of step intensity and assigning them proper weights. The multiple modes of step sizes can be explained by at least two reasons. First, it is possible that two or more fluorophores can bleach at the same time, resulting in larger steps. This probability grows with increasing copy number. Second, a step detection algorithm might group two steps into one when fitting the two steps separately is not statistically significant. This can happen when noise is high, which also often correlates with high copy numbers. The probability of observing single steps consisting of multiple bleaching events is represented by the proportion of each mode in the GMM density estimation.

The final technique that we developed was a best estimate of the total photobleaching amplitude, taking into account the bleaching rate. From the ensemble average, a photobleaching rate constant could be readily extracted. This parameter will vary with excitation intensity, cellular conditions, and other factors, and so needs to be measured for each experiment. If the duration of the experiment is longer than five times the

photobleaching time constant, then it is expected that 99% of the signal has bleached, minimizing the need for any correction. However, long acquisition times are not always possible due to stage or sample drift, camera memory, and underlying cellular dynamics. Hence, correcting for the expected maximum amplitude is important to avoid underestimating copy number.

While the statistical analysis indicated an average copy number of 10 GFP-CESA3 in the observed complexes, we consider this to be a lower limit for the following reasons. First, the GFP-AtCESA3 transgene is present in a background of the partial-loss-of-function AtCESA3^{je5} allele of AtCESA3 [49], meaning that endogenous non-fluorescent AtCESA3 can potentially still be expressed and comprise a portion of each CSC. Second, the time required for microscope focus adjustments necessary to pinpoint the focal plane of the membrane means that some GFP molecules might bleach before images are recorded. Third, it is impossible to rule out the presence of GFP molecules that are misfolded or have not matured (though the estimated 15 minute maturation time constant for eGFP is expected to be sufficiently fast for the present measurements [78]). To improve upon this initial result, we are engineering plants that contain GFP-AtCESA3 expressed in a CESA3 null background. We are also exploring the use of slow-bleaching versions of fluorescent proteins in order to minimize pre-bleaching. Slow bleaching will also improve the ability of step detection algorithms to detect early bleaching steps. An additional uncertainty is whether the two peaks in the copy number distribution indicate that some particles are aggregates of multiple complexes or that two different populations of CSCs exist. To distinguish these two hypotheses, future experiments will focus on photobleaching analysis of motile GFP-AtCESA particles, which presumably represent single CSCs.

In conclusion, we have developed a reliable method for determining copy number in multi-subunit complexes from in vivo photobleaching data. The statistical analysis combines step detection and density estimation to accurately determine the unitary photobleaching step and takes into consideration the bleaching rate constant when determining the maximum fluorescence signal. This method is generic and can be used to estimate the stoichiometry of other membrane-bound complexes and can be applied to

fluorophores other than GFP. Because the signal variance and unitary step size are calculated directly from the raw data, it is not necessary to carry out new controls for different fluorophores, but fluorophores that display more prominent and prolonged dark states such as YFP are expected to have lower SNR, which may set an upper limit on maximum copy numbers that can be reliably estimated. These algorithms can also be adapted to analyze molecular motor stepping data. Applying this method to in vivo photobleaching data gave a lower limit of 10 copies of GFP-AtCESA3 in cellulose synthesis complexes.

Chapter 4

Conclusion

The model-independent algorithms presented in this thesis provide a less biased and higher-precision approach to the step detection problem compared with other methods used in single-molecule microscopy time series analysis [61, 62] (see Section 3.3). These algorithms provide a solution to the significant problem of changing variance within a stepped time series signal and can be used alongside other analysis methods presented here in order to determine copy numbers of multi-subunit complexes from single-molecule photobleaching data.

This thesis also shows that the iterative continuous Viterbi (ICV) algorithm, a model-dependent step detection approach, is a powerful method for uncovering parameters of a generative hidden Markov model of kinesin stepping. This is due to the fact that the ICV algorithm has the critical capability of keeping "phase" of plateaus within a given observation sequence. Therefore, different populations of plateau size and different populations of step size can be analyzed individually, which drastically improves the simplicity and accuracy of subsequent analyses. In tandem with developing advances in high temporal and spatial resolution single-molecule microscopy imaging technologies, this algorithm provides a promising method for elucidating unresolved mechanisms of the kinesin stepping cycle.

Appendix A:

Tdetector Step Detection Algorithm

The following MATLAB code is a current working version of the *Tdetector1* and *Tdetector2* algorithms [1]. The text can be saved as an m file and then it can be called from the workspace just as with a native MATLAB function.

```
%% TDETECTOR Step Detection Algorithm (Tdetector1 and Tdetector2)
% [Y,out] = tdetector(X,var_option)
%
% REQUIRED INPUT:
% ---------------------------------------------------------------------
% X: vector of a piecewise constant function hidden in white noise
%
% OPTIONAL INPUT:
% ---------------------------------------------------------------------
% var_option (Tdetector1 or Tdetector2):
%   [1] assume the corrupting variance of X is constant throughout (default)
%   [2] assume the corrupting variance of X changes throughout
%
% OUTPUTS:
% ---------------------------------------------------------------------
% Y: column vector showing step function fit of X (same length as X)
% out: structure containing information of fitting
%   di: column vector of the declared step indexes (index of each new plateau)
%   ssz: column vector of each step size
%   psz: column vector of each plateau size
%   vx: column vector of the variance at each index
%
% for additional info, visit:
% <a href="matlab:web('http://www.bioe.psu.edu/labs/Hancock-
Lab/tdetector.html','-browser')">http://www.bioe.psu.edu/labs/Hancock-
Lab/tdetector.html</a>

% NOTES:
% ---------------------------------------------------------------------
% - The "out" structure output and the "var_option" input do not have to be
%   included when calling the function. Y = tdetector(X); is valid.
%
% - Cell titles for secondary functions in the code below include the
%   var_options that utilize that respective function in parentheses.

% EXAMPLES:
% ---------------------------------------------------------------------
% 1. Demonstrate Tdetector1:
%
% X = randn(1000,1);
% X(200:end) = X(200:end) + 5;
% X(400:end) = X(400:end) + 5;
% X(600:end) = X(600:end) + 5;
% plot(X,'b'); hold on
% [Y,out] = tdetector(X);
% disp('declared step indexes:');disp(out.di)
% plot(Y,'g');
%
% 2. Demonstrate Tdetector2:
```

```
%
% X = [1*randn(199,1);2*randn(200,1);3*randn(200,1);8*randn(401,1);];
% X(200:end) = X(200:end) + 5;
% X(400:end) = X(400:end) + 5;
% X(600:end) = X(600:end) + 5;
% plot(X,'b'); hold on
% [Y,out] = tdetector(X,2);
% disp('declared step indexes:');disp(out.di)
% plot(Y,'g');

% Nathan Deffenbaugh
% ncd50561234@gmail.com, ncd5056@psu.edu
% (2014 June 10)

%% tdetector

function [Y,out] = tdetector(X,var_option)

% check the var_option, store as VO
VO = 1; % (default to constant variance)
if exist('var_option','var')
    if var_option == 2
        VO = 2;
    end
end

% define full data length
Lo = length(X);

% calculate corrupting variance or variance sections (vx)
if VO == 1
    SIG = getSig(X,1);
    out.vx = SIG^2*ones(Lo,1);
else
    vx = varSect(X);
    out.vx = vx';
end

% define the empirical multiplier lookup table and linearly interpolate
multTab =
[1,0;2,2;3,2.17;4,2.34;6,2.47;8,2.60;11,2.656250;16,2.75;23,2.815625;32,2.90;45
,2.940625;64,3;91,3.0421875000;128,3.10;181,3.1207031250;256,3.15;362,3.1975471
6981100;512,3.24;724,3.280080;1024,3.304768;1448,3.318336;2048,3.325240;2896,3.
329480;4096,3.331096;5793,3.332793;8192,3.3331644000;1e4,3.333300;1e10,3.333300
];
multTab = interp1(multTab(:,1),multTab(:,2),1:Lo);

% step detecting loop
plats_array = [1,Lo];
found = [];
while ~isempty(plats_array)
    Bound = plats_array(end,:);
    % look for a step in this current section of the data
    if VO == 1
        [step_index,status] = detectStep1(X(Bound(1):Bound(2)),Bound(1),SIG,
multTab);
    else
        [step_index,status] =
detectStep2(X(Bound(1):Bound(2)),Bound(1),vx(Bound(1):Bound(2)), multTab);
    end
    % if a significant step is detected
    if status == 1
        found(end+1,1) = step_index;
```

```
        plats_array(end+1,:) = [plats_array(end,1),step_index-1];
        plats_array(end+1,:) = [step_index,plats_array(end-1,2)];
        plats_array(end-2,:) = [];
    elseif status == -1
        plats_array(end,:) = [];
    end
end

% sort the found steps
found = [found;1;Lo];
found = sortrows(found);

% check found steps and build Y vector
if VO == 1
    [checked] = checkSteps(found, X, SIG, multTab, VO);
else
    [checked] = checkSteps(found, X, vx, multTab, VO);
end
checked = [1;checked;Lo+1];
for ii = 1:(length(checked)-1)
    Y(checked(ii):checked(ii+1)-1) = mean(X(checked(ii):checked(ii+1)-1));
end
Y = Y';

% calculate step sizes
step_sizes = zeros(length(checked) - 2,1);
for ii = 2:length(checked)-1
    step_sizes(ii-1) = Y(checked(ii)) - Y(checked(ii) - 1);
end
out.ssz = step_sizes;

% calculate plateau sizes (how many indexes exist between each found step)
out.psz = checked(2:end) - checked(1:end-1);

% output declared step indexes
out.di = checked(2:end-1);

end

%% getSig (1,2)

function [SIG] = getSig(Xs,expnt)

% define pairwise difference vectors
diff1 = diff(Xs);
diff2 = diff(Xs).^2;

while true
    % current estimate of sigma of X
    sigmaC = (mean(diff2)/2)^0.5;

    % remove outlier values from diff vectors
    diff2(abs(diff1) > 3*(2^.5)*sigmaC) = [];
    diff1(abs(diff1) > 3*(2^.5)*sigmaC) = [];

    % new estimate of sigma of X
    sigmaN = (mean(diff2)/2)^0.5;

    if sigmaN == sigmaC
        break
    end
end
% empirical correction for underestimation
```

```
SIG = sigmaN*1.015;
SIG = SIG^expnt;

end
```

```
%% getSigLoop (2)
% getSig function altered slightly to improve speed during a loop

function [SIG] = getSigLoop(Xs,expnt,sd2,ld2)

% Sig Equation
diff1 = diff(Xs);
diff2 = diff(Xs).^2;

% initiate sum to subtract
sts = 0;
% initiate length to subtract
lts = 0;

STOP = 0;
while (STOP == 0)
    SIG = ((sd2-sts)/(ld2-lts)/2)^0.5;

    sd2 = sd2-sts;
    ld2 = ld2-lts;

    icurrpeaks = (abs(diff1) > (3*(2^.5))*SIG);
    currpeaks = diff2(icurrpeaks);
    % sum to subtract
    sts = sum(currpeaks);
    % length to subtract
    lts = length(currpeaks);

    % zero out peaks
    diff1(icurrpeaks) = 0;

    if (lts == 0)
        break
    end
end
SIG = SIG*1.015;
SIG = SIG^expnt;

end
```

```
%% varSect (2)

function [vx] = varSect(X)

% define full data length
Lo = length(X);

% variance sectioning loop
plats_array = [1,Lo];
found = [];
while ~isempty(plats_array)
    Bound = plats_array(end,:);
    [step_index,status] = detectVars(X(Bound(1):Bound(2)),Bound(1));

    if status == 1
        found(end+1,1) = step_index;
        plats_array(end+1,:) = [plats_array(end,1),step_index-1];
        plats_array(end+1,:) = [step_index,plats_array(end-1,2)];
```

```
        plats_array(end-2,:) = [];
    elseif status == -1
        plats_array(end,:) = [];
    end
end

% sort the found variance steps
found = [found;1;Lo];
found = sortrows(found);

% check found variance steps
[checked] = checkVars(found,X);
checked = [1;checked;Lo+1];

% build vx vector
for ii = 1:(length(checked)-1)
    vx(checked(ii):checked(ii+1)-1) = getSig(X(checked(ii):checked(ii+1)-1),2);
end

end
```

```
%% detectVars (2)

function [mxi,status] = detectVars(Xs,i_1)
% in order for any pairwise difference value to be > z*(2^.5)sig and hence
excluded,
%  the length, n, of the diff vector must be n >= z^2 + 1; L >= 2(z^2 + 2). For
%  z = 3, L >= 22. Requiring L >= 22 is necessary to ensure that large pairwise
%  differences due to true steps in the data do not influence the calculated
%  variance of that section.

% define L and default values
L = length(Xs);
status = -1;
mxi = 0;

if (L >= 22)
    d2 = diff(Xs).^2;

    % get sigma of noise
    SIG = getSig(Xs,1);

    % DOV significance rating
    Asd2 = sum(d2(1:9));
    Bsd2 = sum(d2(11:end));
    RVD = zeros(L-2,1);
    for ii = 11:L-10
        Asd2 = Asd2 + d2(ii-1);
        Bsd2 = Bsd2 - d2(ii);

        A = Xs(1:ii);
        B = Xs(ii+1:end);

        VA = getSigLoop(A,2,Asd2,ii-1);
        VB = getSigLoop(B,2,Bsd2,L-ii-1);
        DOV = VA - VB;

        LA = ii;
        LB = L - ii;
        sigma_squared = ( ((LA^2 + LA -3)/((LA-1)^2)) + ((LB^2 + LB -3)/((LB-
1)^2)) - 2 )*SIG^4;

        RVD(ii) = DOV/(sigma_squared^.5)/3;
```

```
    end

    % find the index of the max RVD (rated difference of variance)
    mxi = find(abs(RVD) == max(abs(RVD)));
    mxi = max(mxi);

    % determine the status of the section based on the RVD
    if (abs(RVD(mxi)) > 1)
        status = 1;
        % globalize the step index
        mxi = mxi + i_1;
    end
end

end

%% detectStep1 (1)

function [mxi,status] = detectStep1(Xs,i_1,SIG,multTab)

% define L and default values
L = length(Xs);
status = -1;
mxi = 0;

if (L >= 2)

    % declare sigma multiplier
    mult = multTab(L);

    % DOM significance rating
    RMD = zeros(L,1);
    m1 = 0;
    m2 = sum(Xs);
    for ii = 1:L-1

        m1 = m1 + Xs(ii);
        m2 = m2 - Xs(ii);
        DOM = m2/(L-ii) - m1/(ii);
        sigma = SIG*(1/ii + 1/(L-ii))^.5;

        RMD(ii+1) = DOM/(sigma*mult);

    end

    % find the index of the max RMD (rated difference of mean)
    mxi = find(abs(RMD) == max(abs(RMD)));
    mxi = max(mxi);

    % determine the status of the section based on the RMD
    status = -1;
    if (abs(RMD(mxi)) > 1)
        status = 1;
        % globalize the step index
        mxi = mxi + i_1 - 1;
    end
end

end

%% detectStep2 (2)
```

```
function [mxi,status] = detectStep2(Xs,i_1,vx,multTab)

% define L and default values
L = length(Xs);
status = -1;
mxi = 0;

if (L >= 2)

    % get sigma of noise
    SIG = mean(vx)^.5;

    % declare sigma multiplier
    mult = multTab(L);

    % DOM significance rating
    RMD = zeros(L,1);
    m1 = 0;
    m2 = sum(Xs);
    for ii = 1:L-1

        m1 = m1 + Xs(ii);
        m2 = m2 - Xs(ii);
        DOM = m2/(L-ii) - m1/(ii);
        sigma = SIG*(1/ii + 1/(L-ii))^.5;

        RMD(ii+1) = DOM/(sigma*mult);

    end

    % find the index of the max RMD (rated difference of mean)
    mxi = find(abs(RMD) == max(abs(RMD)));
    mxi = max(mxi);

    % define the RMD_vx value
    sigma_vx = (( sum(vx(1:(mxi - 1)))/((mxi - 1)^2) ) + (
sum(vx((mxi):end))/((L-mxi+1)^2) ))^.5;
    DOM = mean(Xs(1:(mxi - 1))) - mean(Xs(mxi:end));
    RMD_vx = DOM/(mult*sigma_vx);

    % determine the status of the section based on the RMD and RMD_vx
    status = -1;
    if (abs(RMD(mxi)) > 1 && abs(RMD_vx) > 1)
        status = 1;
        % globalize the step index
        mxi = mxi + i_1 - 1;
    end
end

end

%% checkVars (2)

function [checked] = checkVars(found, rx)

% shift last index
found(end) = found(end) + 1;

% initialize
checked = [];
cc = 0;
endW = 0;
```

```
% if there are no found sections to check, do not enter checking loop
if (length(found) == 2)
    endW = 1;
end

% variance section checking loop
ii = 1;
while (endW == 0)
    ii = ii + 1;
    % check variance steps now based on adjacent variance plateaus
    [step_index,status] = detectVars(rx(found(ii-1):found(ii+1)-1),found(ii-
1));

    % if the step is still significant based on adjacent plateaus, store it to
the checked vector
    if (status == 1)
        cc = cc + 1;
        checked(cc,1) = step_index;
    % else, remove it from the found vector
    else
        found(ii) = [];
        ii = ii - 1;
    end

    % if there are no more steps to check, end this while loop
    if ((ii + 1) == length(found))
        endW = 1;
    end
end

end

%% checkSteps (1,2)

function [checked] = checkSteps(found, rx, noise_input, multTab, VO)

% store noise_input
if VO == 1
    SIG = noise_input;
else
    vx = noise_input;
end

% shift last index
found(end) = found(end) + 1;

% initialize
checked = [];
cc = 0;
endW = 0;

% if there are no found steps to check, do not enter checking loop
if (length(found) == 2)
    endW = 1;
end

% step checking loop
ii = 1;
while (endW == 0)
    ii = ii + 1;
    % check steps now based on adjacent plateaus (depending on variance option,
VO)
    if VO == 1
```

```
        [step_index,status] = detectStep1(rx(found(ii-1):found(ii+1)-
1),found(ii-1),SIG, multTab);
    else
        [step_index,status] = detectStep2(rx(found(ii-1):found(ii+1)-
1),found(ii-1),vx(found(ii-1):found(ii+1)-1), multTab);
    end

    % if the step is still significant based on adjacent plateaus, store it to
the checked vector
    if (status == 1)
        cc = cc + 1;
        checked(cc,1) = step_index;
    % else, remove it from the found vector
    else
        found(ii) = [];
        ii = ii - 1;
    end

    % if there are no more steps to check, end this while loop
    if ((ii + 1) == length(found))
        endW = 1;
    end
end

end
```

Appendix B:

Iterative Continuous Viterbi Algorithm

The following MATLAB code performs the iterative continuous Viterbi (ICV) Hidden Markov Model step detection algorithm.

```
%% icv (iterative continuous-Viterbi algorithm)
% [ret,R,o] = icv(y,[Fss,delta_incr,skip_offset])
%
% Inputs
% ------------------------------------------
% y: piecewise constant vector corrupted by noise (emission values from a
%    hidden markov model of a stepping motor)
%
% optional inputs ...
%
% Fss: full step size of motor (16.4 nm default)
% delta_incr: number of delta increments tested (20 default)
% det_offset: if set to 0 then offset detect is skipped (1 default)
%
% Outputs:
% ------------------------------------------
% ret: vector of step trace HMM fit that returned lowest mean squared error
%      (same size as input y). Values are locked to means of this optimal
%      HMM emission matrix.
%
% R: matrix of delta values tested (column 1) and their
%    respective mean squared errors of fitting (column 2)
%
% o(1): structure containing other information of HMM fitting
%   mvar: mean variance of input y (calculation from tdetector function)
%   off: offset of input y based on expected full step size of Fss or 16.4
%        by default (calculation from odetect function)
%   A: HMM transition matrix
%   U: HMM initial probabilities matrix
%   lambda: HMM inverse of expected plateau length (data points)
%   N: HMM length of transition matrix
%   toc: analysis run time
%   opti: optimal delta value (lowest mean squared error)
%   di: declared indexes of stepping from optimal HMM fit (Y)
%   Y: optimal HMM fit, similar to "ret" output, but with each plataeu as
%      the mean of input data (y) in that section
%   ssz: step sizes from Y fit
%   psz: plateau sizes from Y fit
%   even: structure of even plateau sizes and step sizes
%   odd: structure of odd plateau sizes and step sizes
%
% o(n).x: vector of HMM fit for each tested delta value. The highest
%   value of n = delta_incr, or 20 by default. Whichever tested delta
%   yielded the lowest mean squared error will have its o(n).x = ret.

%% icv (requires Tdetector function)

function [ret,R,o] = icv(y,Fss,delta_incr,det_offset)
% perform preliminary step detection (Tdetector) for offset detector inputs
[tdet_x,tdet_o] = tdetector(y);
o(1).mvar = mean(tdet_o.vx);
```

```
% shift data to make tdet_x(1) = 0
y = y - tdet_x(1);

% set input defaults
if nargin < 2
    Fss = 16.4; % default full step size (microtubule binding site spacing)
end
if nargin < 3;
    delta_incr = 20; % default increment number (arbitrary)
end
if nargin < 4;
    det_offset = 1; % default to offset detection
end

% perform offset detection, and offset input y vector
if det_offset == 1
    od = odetect(y,tdet_o,Fss);
    y = y + od(1).off;
    o(1).off = od(1).off - tdet_x(1);
    o(1).od = od;
end

% begin ICV analysis ...
tic
T = length(y);

% build estimated transition matrix
N = 2*(ceil(max(y)/Fss) - floor(min(y)/Fss) + 1);
lambda = N/T;
A = HMMbuildTransition(lambda,N);

% initial probs
U = 1/N*ones(1,N);

% build emission matrix and perform continuous-Viterbi (iterative)
delta = linspace(0,Fss,delta_incr);
R(:,1) = delta;
for ii = 1:length(delta)
    B = HMMbuildEmission(delta(ii),0,tdet_o.vx(1)^0.5,N,Fss);
    [o(ii).s,R(ii,2)] = vitec(y,A,B,U);
    o(ii).x = B(o(ii).s,1);
    o(ii).delta = delta(ii);
end
% R(:,2) = 1./R(:,2);

o(1).A = A;
o(1).U = U;
o(1).lambda = lambda;
o(1).N = N;
o(1).toc = toc;

% return optimal delta value
[dum,ind] = min(R(:,2));
o(1).opti = o(ind).delta;
ret = o(ind).x;

% calculate declared step indexes
o(1).di = find(diff(ret) ~= 0) + 1;
step_indexes = [1;o(1).di;T+1];

% calc noise corrupted step sizes
for ii = 1:(length(step_indexes)-1)
```

```
    Y(step_indexes(ii):step_indexes(ii+1)-1) =
mean(y(step_indexes(ii):step_indexes(ii+1)-1));
end
o(1).Y = Y';
step_sizes = zeros(length(step_indexes) - 2,1);
for ii = 2:length(step_indexes)-1
    step_sizes(ii-1) = Y(step_indexes(ii)) - Y(step_indexes(ii) - 1);
end
o(1).ssz = step_sizes;

% calculate plateau sizes (how many indexes exist between each found step)
o(1).psz = step_indexes(2:end) - step_indexes(1:end-1);

% sort even and odd index step sizes and plateaus
inds = 1:length(o(1).ssz);
o(1).even.ssz = o(1).ssz(find(rem(inds,2)*-1+1));
o(1).odd.ssz = o(1).ssz(find(rem(inds,2)));
inds = 1:length(o(1).psz);
o(1).even.psz = o(1).psz(find(rem(inds,2)*-1+1));
o(1).odd.psz = o(1).psz(find(rem(inds,2)));


end
```

```
%% odetect (offset detector)
% [ret] = odetect(x,o,ss);
%
% Inputs:
% ----------------------------------
% x: piecewise constant signal corrupted by noise. This is also the
%    emission values from a Hidden Markov Model
% o: output structure from the Tdetector algorithm.
% ss: step size for which the x data offset is to be calculated
%
% Outputs:
% ----------------------------------
% ret: structure containing the following:
%

function ret = odetect(x,o,ss)
% acceptable range parameter
arp = 0.05;
arv = arp.*ones(size(o.psz));

% normalize fit to start from zero and by unit step size
Q = [0;cumsum(o.ssz)]./ss;

for i = 1:length(Q)
    % shift Q so that the ith plateau is at zero
    q = Q - Q(i);
    % calc q absolute error
    qae = abs(q - round(q));
    % truth list if each plat is within acceptable range
    z = arv >= qae;
    ret(i).z = z;
    % declare plateau scores
    ret(i).zs = sum(o.psz(find(z))); % z score (plat point count)
    ret(i).zc = sum(z); % z count
end

ret(1).Q = Q;
% cumulative plateau size
cps = [0;cumsum(o.psz)];
```

```
% initialize optimal plateau score
opt_zs = 0;
for i = 1:length(o.psz)
    ret(i).plat = x(cps(i) + 1:cps(i+1));
    % check for optimal reference plateau
    if (ret(i).zc >= 2) && (ret(i).zs >= opt_zs)
        % update
        ret(1).opt = i;
        opt_zs = ret(i).zs;
    end
end
% if no synced plateaus found
if opt_zs == 0; ret(1).opt = 0; end

% define Qr
Qr = round(Q - Q(ret(1).opt));
ret(1).Qr = Qr;

% calculate offset
if size(x,1) == 1;
    x = x'; % make sure x is a column vector
end
plat_vect = [];
plat1s = find(ret(ret(1).opt).z);
for i = 1:length(plat1s)
    plat_vect = [plat_vect;ret(plat1s(i)).plat - Qr(plat1s(i))*ss];
end
ret(1).plat_vect = plat_vect;
ret(1).plat1s = plat1s;
ret(1).mpv = mean(plat_vect);
% calc offset (what to shift so that a plateau exactly on mpv would be an
% integer multiple of the ss)
ret(1).off = round(ret(1).mpv/ss)*ss - ret(1).mpv;

% organize plateaus within acceptable range for plotting
ARP = [];
for ii = 1:length(ret(ret(1).opt).z)
    if ret(ret(1).opt).z(ii) == 1
        pp = mean(ret(ii).plat).*ones(size(ret(ii).plat));
    else
        pp = Inf*ret(ii).plat;
    end
    ARP = [ARP;pp];
end
ret(1).arp = ARP;

end

%% HMMbuildTransition
% builds HMM transition matrix for kinesin stepping
% A = HMMbuildTransition(g,K)

function A = HMMbuildTransition(lambda,N)

eta = 0.999;
A = (1 - lambda)*eye(N);
A(1,2) = lambda;
for i = 2:N-1
    A(i,i+1) = eta*lambda;
    A(i,i-1) = (1-eta)*lambda;
end
A(N,N-1) = lambda;
if sum(sum(A,2)) ~= N
```

```
    disp 'TRANSITION MATRIX ERROR'
end

end

%% HMMbuildEmission
% builds HMM emission matrix for kinesin stepping

function B = HMMbuildEmission(delta,offset,Nstd,N,Fss)
B = Nstd*ones(N,2);
B(1,1) = 0;

for i = 2:N
    if rem(i,2) == 0
        B(i,1) = B(i-1,1) + delta;
    else
        B(i,1) = (i-1)/2*Fss;
    end
end

B(:,1) = B(:,1) + offset;

end

%% vitec (viterbi-continuous algorithm)
% [x,r] = vitec(y,A,B,U)
%
% INPUTS:
% -------------------------------------
% y: observation series (1xT or Tx1 vector)
% A: transition matrix (NxN array)
%    where A(i,j) is the probability of transitioning from state i to
%    state j. P(x(t) = j | x(t-1) = i)
% B: emission matrix (Nx2 array)
%    where y(i) ~ N( B(x(i),1), B(x(i),2) ), that is ~ normal distribution
%    with mean = B(x(i),1), and std = B(x(i),2)
% U: initial probability vector (Nx1 or 1xN vector)
%    where U(i) = P(x(1) = i)
%
% OUTPUTS:
% -------------------------------------
% x: most probable state sequence (1xT vector)
% r: rating, (sum of squared residuals)/T

function [x,r] = vitec(y,A,B,U)

T = length(y);
N = length(A);

T1 = zeros(N,T);
T2 = T1;

% calculate P(x_i|y_1)
for i = 1:N
    T1(i,1) = U(i)*pdens(B(i,:),y(1));
    T2(i,1) = 0;
end

% construct T1 and T2
for i = 2:T
    for j = 1:N
        % [max,argmax] = max();
        [T1(j,i),T2(j,i)] = max( T1(:,i-1).*A(:,j)*pdens(B(j,:),y(i)) );
```

```
        % T1(j,i) is the relative probability that x(i) = j, given y(1:i)
        % T2(j,i) is the most likely x(i-1) given that x(i) = j
    end
    T1(:,i) = T1(:,i)/sum(T1(:,i));
end

% find most likely final state
[dum,z(T)] = max(T1(:,T));
x(T) = z(T);

% trace back through most likely path
for i = T:-1:2
    z(i-1) = T2(z(i),i);
    x(i-1) = z(i-1);
end

% calculate rating
if size(y,1) ~= 1;
    y = y';
end
bx = B(x,1);
r = sum((bx' - y).^2)/T;

end

%% pdens [called by: vitec]

function ret = pdens(b,y)
ret = 1/(b(2)*(2*pi)^.5).*exp( -((y-b(1)).^2) / (2*b(2)^2) );
end
```

Appendix C:

Differences of Variance; Pairwise Differences vs. χ^2

In the *Tdetector* algorithms, the variance of corrupting noise of the input signal or sections of the input signal are calculated using the pairwise difference method described in Section 3.8.5. Therefore, when the difference of variances (DOV) between two sections are being tested for significance, it is necessary to test according to the Var(DOV) given in Equation 3.3 (as derived in Section 3.8.7) rather than simply the Var(DOV) from comparing two χ^2 distributions given in Equation A.1:

$$\mathrm{Var}\left(\mathrm{DOV}_{\chi}\right) = \sigma^2_{\mathrm{DOV}\,\chi^2} = \sigma^4\left(\frac{2}{i} + \frac{2}{L-i}\right) \qquad \text{(A.1)}$$

The differences in the two ways of calculating Var(DOV) are evident in Figure A.1 as produced by MATLAB code in Table A.1.

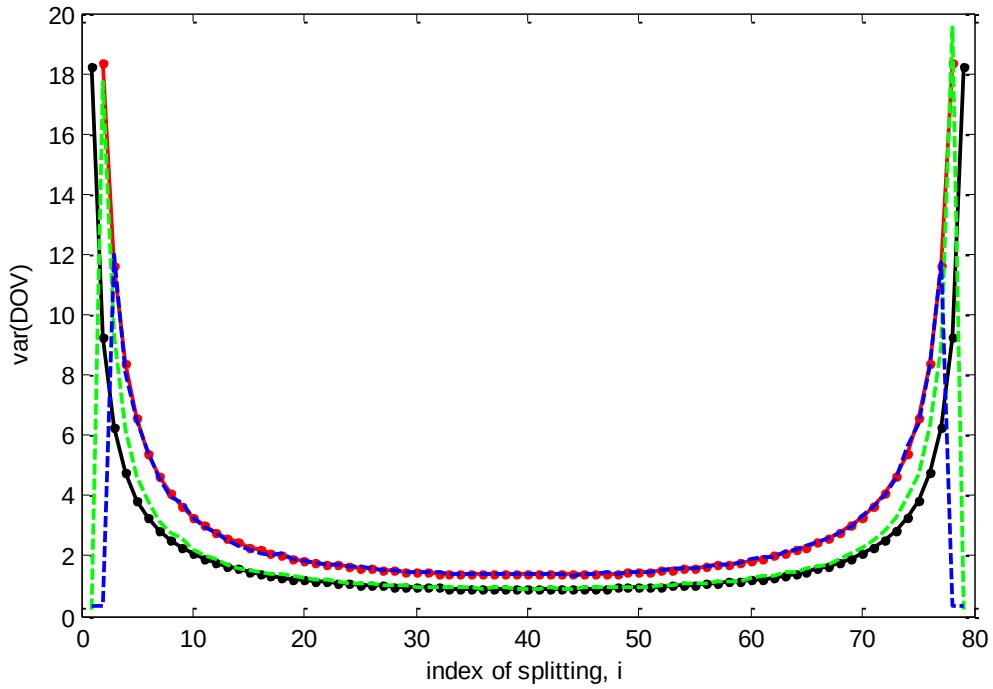

Figure A.1: Var(DOV) calculations on generated random vectors with respect to different indices of splitting. Analytical Var(DOV) from Equation 3.3 (red line), analytical Var(DOV) from Equation A.1 (black line), Var(DOV) from pairwise difference calculation (blue dotted line), Var(DOV) from conventional calculation (green dotted line).

Table A.1: MATLAB code used to generate and calculate Var(DOV) for different indices of splitting.

```
% define length of X vector and index of splitting, i
L = 80;
i = 1:L-1;
sig2 = 3;

% analytical calculations of expected var(DOV)
VDOV_pds = sig2^2*((i.^2+i-3)./(i-1).^2 + ( (L-i).^2+(L-i)-3 )./((L-i)-1).^2 - 2);
VDOV_Chi = sig2^2*(2./(i) + 2./(L-i));

% generate random X vectors, split at i, then calculate variance of DOVs
h = waitbar(0,'Please wait...');
for i = 1:L-1
    % iteratations for each index of splitting
    ites = 10000;
    X = sig2^.5*randn(L,ites);
    % split X
    Xa = X(1:i,:);
    Xb = X(i+1:end,:);

    % calc DOV by conventional method and pairwise difference method
    dov_chi = var(Xa) - var(Xb);
    dov_pds = (mean(diff(Xa).^2)/2) - (mean(diff(Xb).^2)/2);

    % store the variance of DOVs
    vdov_chi(i) = var(dov_chi);
    vdov_pds(i) = var(dov_pds);
    waitbar(i / L)
end
close(h)

% plotting
figure
plot(VDOV_pds,'.-r')
hold on
plot(VDOV_Chi,'.-k')
plot(vdov_chi,'--g')
plot(vdov_pds,'--b')
xlabel('index of splitting, i')
ylabel('var(DOV)')
```

REFERENCES

[1] Y. Chen, N. C. Deffenbaugh, C. T. Anderson and W. O. Hancock, "Molecular counting by photobleaching in protein complexes with many subunits: best practices and application to the cellulose synthesis complex," *Molecular Biology of the Cell,* vol. 25, no. 22, pp. 3630-42, 2014.

[2] N. Hirokawa, K. Pfister, H. Yorifuji, M. C. Wagner, S. T. Brady and G. S. Bloom, "Submolecular domains of bovine brain kinesin identified by electron microscopy and monoclonal antibody decoration," *Cell,* vol. 56, no. 5, p. 867–78 , 1989.

[3] J. Howard, Mechanics of Motor Proteins and the Cytoskeleton, Sunderland, MA: Sinauer Associates, 2001.

[4] S. Hisanga, H. Murofushi, K. Okuhara, R. Sato, Y. Masuda, H. Sakai and N. Hirokawa, "The molecular-structure of adrenal-medulla kinesin," *Cell Motil. Cytoskeleton,* vol. 12, pp. 265-272, 1989.

[5] J. M. Scholey, J. Heuser, J. T. Yang and J. S. B. Goldstein, "Identification of globular mechanochemical heads of kinesin," *Nature,* vol. 338, pp. 355-357, 1989.

[6] J. Kerssemakers, J. Howard, H. Hess and S. Diez, "The distance that kinesin-1 holds its cargo from the microtubule surface measured by fluorescence interference contrast microscopy," *Proc. Natl. Acad. Sci. U.S.A.,* vol. 103, pp. 15812-15817, 2006.

[7] Z. Wang, M. Feng, W. Zheng and D. Fan, "Kinesin is an evolutionarily fine-tuned molecular ratchet-and-pawl device of decisively locked direction," *Biophysical Journal,* vol. 93, pp. 3363-3372, 2007.

[8] R. D. Vale, "The molecular motor toolbox for intracellular transport," *Cell,* vol. 4, no. 112, pp. 467-480, 2003.

[9] G. M. Jeppesen and J. K. H. Hoerber, "The mechanical properties of kinesin-1: a holistic approach," *Biochemical Society Transactions,* vol. 40, pp. 438-443, 2012.

[10] Y. L. Wong and S. E. Rice, "Kinesin's light chains inhibit the head- and microtubule-binding activity of its tail," *Proc Natl Acad Sci U.S.A.,* vol. 26, no. 107, pp. 11781-11786, 2010.

[11] H. Miki, Y. Okada and N. Hirokawa, "Analysis of the kinesin superfamily; insights into the structure and function," *Trends Cell Biol,* vol. 15, pp. 467-476, 2005.

[12] H. Miki, M. Setou, K. Kaneshiro and N. Hirokawa, "All kinesin superfamily protein, KIF, genes in mouse and human," *Proc Natl Acad Sci U.S.A.,* vol. 98, pp. 7004-7011, 2001.

[13] R. D. Vale, T. S. Reese and M. P. Sheetz, "Identification of a novel force-generating protein, kinesin, involved in microtubule-based motility," *Cell,* vol. 42, no. 1, pp. 39-50, 1985.

[14] D. G. Cole, D. R. Diener, A. L. Himelblau, P. L. Beech, J. C. Fuster and J. L. Rosenbaum, "Chlamydomonas kinesin-II-dependent intraflagellar transport (IFT): IFT particles contain proteins required for ciliary assembly in Caenorhabditis elegans sensory neurons," *J Cell Biol,* vol. 141, pp. 993-1008, 1998.

[15] J. M. Scholey, "Intraflagellar transport," *Annual Rev Cell Dev Biol,* vol. 19, pp. 423-443, 2003.

[16] G. Bergnes, K. Brejc and L. Belmont, "Mitotic kinesins: prospects for antimitotic drug discovery," *Curr Top Med Chem,* vol. 5, no. 2, pp. 127-145, 2005.

[17] C. J. Lawrence, R. Dawe, K. R. Christie, D. W. Cleveland, S. C. Dawson, S. A. Endow, L. S. B. Goldstein, H. V. Goodson, N. Hirokawa, J. Howard, R. L. Malmberg, J. R. McIntosh, H. Miki, T. J. Mitchison, Y. Okada, A. Reddy, W. M. Saxton, M. Schliwa, J. M. Scholey, R. D. Vale, C. E. Walczak and L. A. Wordeman, "A standardized kinesin nomenclature," *Journal of Cell Biology,* vol. 167, no. 1, pp. 19-22, 2004.

[18] E. Chevalier-Larsen and E. L. Holzbaur, "Axonal transport and neurodegenerative disease," *Biochimica et Biophysica Acta (BBA) - Molecular Basis of Disease,* vol. 1762, no. 11-12, p. 1094–1108, 2006.

[19] T. U. Mayer, T. M. Kapoor, S. J. Haggarty, R. W. King, S. L. Schreiber and T. J. Mitchison, "Small Molecule Inhibitor of Mitotic Spindle Bipolarity Identified in a Phenotype-Based Screen," *Science,* vol. 286, no. 5441, pp. 971-974, 1999.

[20] K. E. Sawin, K. LeGuellec, M. Philippe and T. J. Mitchison, "Mitotic spindle organization by a plus-end-directed microtubule motor," *Nature,* vol. 359, pp. 540-543, 1992.

[21] T. M. Kapoor, T. U. Mayer, M. L. Coughlin and a. T. J. Mitchison, "Probing spindle assembly mechanisms with monastrol, a small molecule inhibitor of the mitotic kinesin, Eg5," *Journal of Cell Biology,* vol. 150, no. 5, pp. 975-988, 2000.

[22] R. Sakowicz, J. Finer, C. Beraud, A. Crompton, E. Lewis, A. Fritsch, Y. Lee, J. Mak, R. Moody, R. Turincio, J. Chabala, P. Gonzales, S. Roth, S. Weitman and K. Wood, "Antitumor activity of a kinesin inhibitor," *Cancer Research,* vol. 64, no. 9, pp. 3276-80, 2004.

[23] K. Svoboda, C. F. Schmidt, B. J. Schnapp and S. M. Block, "Direct observation of kinesin stepping by optical trapping interferometry," *Nature,* vol. 365, pp. 721-727, 1993.

[24] C. L. Asbury, A. N. Fehr and S. M. Block, "Kinesin moves by an asymmetric hand-over-hand mechanism," *Science,* vol. 302, pp. 2130-2134, 2003.

[25] A. Yildiz, M. Tomishige, R. D. Vale and P. R. Selvin, "Kinesin walks hand-over-hand," *Science,* vol. 303, pp. 676-678, 2004.

[26] M. J. Schnitzer and S. M. Block, "Kinesin hydrolyses one ATP per 8-nm step," *Nature,* vol. 388, pp. 386-390, 1997.

[27] D. L. Coy, M. Wagenbach and J. Howard, "Kinesin Takes One 8-nm Step for Each ATP That It Hydrolyzes," *Journal of Biological Chemistry,* vol. 274, pp. 3667-3671, 1999.

[28] S. Shastry and W. O. Hancock, "Neck linker length determines the degree of processivity in kinesin-1 and kinesin-2 motors," *Current Biology,* vol. 20, pp. 939-943, 2010.

[29] W. O. Hancock and J. Howard, "Kinesin's processivity results from mechanical and chemical coordination between the ATP hydrolysis cycles of the two motor domains," *Proceedings of the National Academy of Sciences,* vol. 96, no. 23, p. 13147–13152, 1999.

[30] S. S. Rosenfeld, P. M. Fordyce, G. M. Jefferson, P. H. King and S. M. Block, "Stepping and stretching. How kinesin uses internal strain to walk processively," *Journal of Biological Chemistry,* vol. 278, pp. 18550-56, 2003.

[31] S. Shastry and W. O. Hancock, "Interhead tension determines processivity across diverse N-terminal kinesins," *Proceedings of the National Academy of the Sciences, USA,* vol. 108, no. 39, pp. 16253-16258, 2011.

[32] D. D. Hackney, "Evidence for alternating head catalysis by kinesin during microtubule-stimulated ATP hydrolysis," *Proc Natl Acad Sci U.S.A,* vol. 15, no. 91, pp. 6865-9, 1994.

[33] G. Muthukrishnan, Y. Zhang, S. Shastry and W. O. Hancock, "The processivity of kinesin-2 motors suggests diminished front-head gating," *Current Biology,* vol. 19, pp. 442-447, 2009.

[34] M. T. Valentine and S. P. Gilbert, "To step or not to step? How biochemistry and mechanics influence processivity in Kinesin and Eg5," *Curr Opinion Cell Biol,* vol. 19, pp. 75-81, 2007.

[35] S. M. Block, "Kinesin motor mechanics: binding, stepping, tracking, gating, and limping," *Biophysical Journal,* vol. 92, pp. 2986-2995, 2007.

[36] D. D. Hackney, "The tethered head domain of a kinesin-microtubule complex catalyzes reversible synthesis of bound ATP," *Proc Natl Acad Sci U.S.A.,* vol. 102, pp. 18338-18343, 2005.

[37] N. R. Guydosh and S. M. Block, "Direct observation of the binding state of the kinesin head to the microtubule," *Nature,* vol. 461, pp. 125-128, 2009.

[38] S. Uemura, K. Kawaguchi, J. Yajima, M. Edamatsu, Y. Y. Toyoshima and S. Ishiwata, "Kinesin–microtubule binding depends on both nucleotide state and loading direction," *Proc Natl Acad Science U.S.A.,* vol. 99, no. 9, pp. 5977-5981, 2002.

[39] J. O. Andreasson, S. Shastry, W. O. Hancock and S. M. Block, "The mechanochemical cycle of mammalian kinesin-2 KIF3A/B under load," *Current Biology,* vol. (In Press), 2015.

[40] D. Axelrod, " Cell-substrate contacts illuminated by total internal reflection fluorescence," *Journal of Cell Biology,* vol. 89, no. 1, pp. 141-145, 1981.

[41] F. Ruhnow, D. Zwicker and S. Diez, "Tracking single particles and elongated filaments with nanometer precision," *Biophysical Journal,* vol. 100, pp. 2820-2828, 2011.

[42] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *The Annals of Mathematical Statistics,* vol. 37, no. 6, pp. 1554-1563, 1960.

[43] J. Ortega-Arroyo and P. Kukura, "Interferometric scattering microscopy (iSCAT): new frontiers in ultrafast and ultrasensitive optical microscopy," *Phys Chem Chem Phys,* vol. 14, pp. 15625-15636, 2012.

[44] H. Ueno, S. Nishikawa, R. Iino, K. Tabata, S. Sakakihara, T. Yanagida and H. Noji, "Simple dark-field microscopy with nanometer spatial precision and microsecond temporal resolution," *Biophysical Journal,* vol. 98, pp. 2014-2023, 2010.

[45] A. Carroll and C. Somerville, "Cellulosic biofuels," *Annu Rev Plant Biol,* vol. 60, pp. 165-182, 2009.

[46] H. McFarlane, A. Doring and S. Persson, "The cell biology of cellulose synthesis," *Annu Rev Plant Biol,* vol. 65, p. 69–94, 2014.

[47] C. Haigler and R. Brown, "Transport of rosettes from the golgi apparatus to the plasma membrane in isolated mesophyll cells of Zinnia elegans during differentiation to tracheary elements in suspension culture," *Protoplasma,* vol. 134, p. 111, 1986.

[48] N. Taylor, R. Howells, A. Huttly, K. Vickers and S. Turner, "Interactions among three distinct CesA proteins essential for cellulose synthesis," *Proc Natl Acad Sci USA,* vol. 100, p. 1450–1455, 2003.

[49] T. Desprez, M. Juraniec, E. Crowell, H. Jouy, Z. Pochylova, F. Parcy, H. Hofte, M. Gonneau and S. Vernhettes, "Organization of cellulose synthase complexes involved in primary cell wall synthesis in Arabidopsis thaliana," *Proc Natl Acad Sci USA,* vol. 104, p. 15572–15577, 2007.

[50] S. Persson, A. Paredez, A. Carroll, H. Palsdottir, M. Doblin, P. Poindexter, N. Khitrov, M. Auer and C. Somerville, "Genetic evidence for three unique components in primary cell-wall cellulose synthase complexes in Arabidopsis," *Proc Natl Acad Sci USA,* vol. 104, p. 15566–15571, 2007.

[51] A. Fernandes, L. Thomas, C. Altaner, P. Callow, V. Forsyth, D. Apperley, C. Kennedy and M. Jarvis, "Nanostructure of cellulose microfibrils in spruce wood," *Proc Natl Acad Sci USA,* vol. 108, p. E1195–E1203, 2011.

[52] L. Thomas, V. Forsyth, A. Sturcova, C. Kennedy, R. May, C. Altaner, D. Apperley, T. Wess and M. Jarvis, "Structure of cellulose microfibrils in primary cell walls from collenchyma," *Plant Phys,* vol. 161, p. 465–476, 2013.

[53] L. Sethaphong, C. Haigler, J. Kubicki, J. Zimmer, D. Bonetta, S. DeBolt and Y. Yingling, "Tertiary model of a plant cellulose synthase," *Proc Natl Acad Sci USA,* vol. 110, p. 7512–7517, 2013.

[54] G. Guerriero, J. Fugelstad and V. Bulone, "What do we really know about cellulose biosynthesis in higher plants," *J Integr Plant Biol,* vol. 52, pp. 161-175, 2010.

[55] J. Lai-Kee-Him, H. Chanzy, M. Muller, J. Putaux, T. Imai and V. Bulone, "In vitro versus in vivo cellulose microfibrils from plant primary wall synthases: structural differences," *J Biol Chem,* vol. 277, p. 36931–36939, 2002.

[56] C. Cifuentes, V. Bulone and A. Emons, "Biosynthesis of callose and cellulose by detergent extracts of tobacco cell membranes and quantification of the polymers synthesized in vitro," *J Integr Plant Biol,* vol. 52, pp. 221-233, 2010.

[57] S. Fujii, T. Hayashi and K. Mizuno, "Sucrose synthase is an integral component of the cellulose synthesis machinery," *Plant Cell Physiol,* vol. 51, pp. 294-301, 2010.

[58] M. Ulbrich and E. Isacoff, "Subunit counting in membrane-bound proteins," *Nat Methods,* vol. 4, p. 319–321, 2007.

[59] M. Leake, J. Chandler, G. Wadhams, F. Bai, R. Berry and J. Armitage, "Stoichiometry and turnover in single, functioning membrane protein complexes," *Nature,* vol. 443, p. 355–358, 2006.

[60] B. Carter, M. Vershinin and S. Gross, "A comparison of step-detection methods: how well can you do," *Biophysical Journal,* vol. 94, pp. 306-319, 2008.

[61] J. Kerssemakers, E. Munteanu, L. Laan, T. Noetzel, M. J. ME and M. Dogterom, "Assembly dynamics of microtubules at molecular resolution," *Nature,* vol. 442, p. 709–712, 2006.

[62] B. Kalafut and K. Visscher, "An objective, model-independent method for detection of non-uniform steps in noisy signals," *Comput Phys Commun,* vol. 179, pp. 716-723, 2008.

[63] Y. Sowa, A. Rowe, M. Leake, T. Yakushi, M. Homma, A. Ishijima and R. Berry, "Direct observation of steps in rotation of the bacterial flagellar motor," *Nature*, vol. 437, pp. 916-919, 2005.

[64] A. Snijders, N. Nowak, R. Segraves, S. Blackwood, N. Brown, J. Conroy, G. Hamilton, A. Hindle, B. Huey, K. Kimura, S. Law, K. Myambo, J. Palmer, B. Ylstra, J. Yue, J. Gray, A. Jain and D. P. D. Albertson, "Assembly of microarrays for genome-wide measurement of DNA copy number," *Nature Genetics,* vol. 29, pp. 263-264, 2001.

[65] E. Page, "A test for a change in a parameter occurring at an unknown point," *Biometrika,* vol. 42, p. 523–527, 1955.

[66] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *Information Theory, IEEE Transactions,* vol. 13, no. 2, pp. 260-269, 1967.

[67] C. Konopka and S. Bednarek, "Variable-angle epifluorescence microscopy: a new way to look at protein dynamics in the plant cell cortex," *Plant J,* vol. 53, p. 186–196, 2008.

[68] C. Konopka, S. Backues and S. Bednarek, "Dynamics of Arabidopsis dynamin-related protein 1C and a clathrin light chain at the plasma membrane," *Plant Cell,* vol. 20, p. 1363–1380, 2008.

[69] J. Lakowicz, Principles of Fluorescence Spectroscopy, New York: Springer, 2010.

[70] G. Schwarz, "Estimating the dimension of a model," *Ann Stat,* vol. 6, p. 461–464, 1978.

[71] B. Silverman, Density Estimation for Statistics and Data Analysis, London: Chapman & Hall, 1986.

[72] A. Dempster, N. Laird and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J R Stat Soc,* vol. 39, pp. 1-21, 1977.

[73] B. L. Welch, "The generalization of "Student's" problem when several different population variances are involved," *Biometrika,* vol. 34, pp. 28-35, 1947.

[74] R. Kass and A. Raftery, "Bayes factors," *J Am Stat Assoc,* vol. 90, p. 773–795, 1995.

[75] D. Scott, Multivariate Density Estimation: Theory, Practice, and Visualization, Hoboken, NJ: John Wiley & Sons, 1992.

[76] T. Benaglia, D. Chauveau, D. Hunter and D. Young, "Mixtools: an R package for analyzing mixture models," *J Stat Software,* vol. 32, pp. 1-29, 2009.

[77] K. Nakajo, M. Ulbrich, Y. Kubo and E. Isacoff, "Stoichiometry of the KCNQ1-KCNE1 ion channel complex," *Proc Natl Acad Sci USA,* vol. 107, p. 2010, 18862–18867.

[78] R. Iizuka, M. Yamagishi-Shirasaki and T. Funatsu, "Kinetic study of de novo chromophore maturation of fluorescent proteins," *Analytical Biochem,* vol. 414, pp. 173-178, 2011.

ACADEMIC VITA

Nathan C. Deffenbaugh

Email: ncd5056@psu.edu, ncd50561234@gmail.com
Telephone: 1 (814) 883-3545

EDUCATION

2013 – 2015	Pennsylvania State University, University Park, PA M.S. Bioengineering (May 2015)
2010 – 2015	Pennsylvania State University, Schreyer Honors College, University Park, PA B.S. Bioengineering (May 2015)

RESEARCH/INTERNSHIP EXPERIENCE

- **Student Researcher**, Department of Biomedical Engineering, Penn State University (May 2012 – May 2015) PI: William O. Hancock, Ph.D.

  Developed novel, high-performance signal processing algorithms for motor protein positional data analysis, and for temporal intensity data analysis of protein complexes. Implemented computational and analytical techniques for modeling kinesin motor protein kinetics. Imaging single-molecule kinesin motility assays using total internal reflection fluorescence microscopy (TIRF) to identify discrete steps along microtubules.

- **Energy and Engineering Internship**, Office of the Physical Plant, Penn State University
  (Jun. 2011 – Dec. 2011)

  Organized control programs for campus HVAC network using Automated Logic Corporation software (WebCTRL). Programmed and updated graphical interfaces for controls of variable air volume units (VAVs) and air handling units (AHUs) for university buildings.

PUBLICATIONS

- Molecular Counting by Photobleaching in Protein Complexes with Many Subunits: Best Practices and Application to the Cellulose Synthesis Complex, Y. Chen, **N.C. Deffenbaugh**, C.T. Anderson, W.O. Hancock. *Mol. Biol. Cell,* September 17, 2014, doi: 10.1091/mbc.E14-06-1146

PRESENTATIONS

- *Talk*: Molecular Counting by Photobleaching in Protein Complexes with Many Subunits: Best Practices and Application to the Cellulose Synthesis Complex, **N.C. Deffenbaugh,** Y. Chen, C.T. Anderson, W.O. Hancock (2014) GE Student Research Summit Professional Networking Dinner, Niskayuna, NY
- *Poster*: Molecular Counting by Photobleaching in Protein Complexes with Many Subunits: Best Practices and Application to the Cellulose Synthesis Complex, **N.C.**

Deffenbaugh, Y. Chen, C.T. Anderson, W.O. Hancock (2014) GE Student Research Summit, Niskayuna, NY

- *Poster*: Investigating the Front-head Gating in KIF3A, G. Chen, **N.C. Deffenbaugh**, D. Arginteanu, W.O. Hancock, (2014) Biophysical Society Meeting, San Francisco, CA
- *Poster*: High Resolution Tracking of Single-Molecule Kinesin Motor Proteins by TIRF Microscopy, **N.C. Deffenbaugh**, W.O. Hancock, (2014) Undergraduate Research at the Capitol – Pennsylvania, Harrisburg, PA
- *Poster*: Biochemical Investigations into the Kinesin-2 Chemomechanical Cycle, W.O. Hancock, **N.C. Deffenbaugh**, D. Arginteanu (2013) Biophysical Society Meeting, Philadelphia, PA
- *Poster*: Novel Computational and Analytical Tools for Modeling Kinesin Protein Biochemistry, **N.C. Deffenbaugh**, W.O. Hancock, (2012) BMES Annual Meeting, Atlanta, GA

TEACHING EXPERIENCE

- BME 301 – Analysis of Physiological Systems, Teaching Assistant (Penn State University, 2014): Instructor of lab periods on signal processing and simulation (MATLAB, Simulink), graded lab assignments, held office hours, held midterm review sessions

SKILLS

- **Programming**: Design of data analysis algorithms; signal/image processing; strong with MATLAB; familiar with C++, Python, LabVIEW; strong with HTML, PHP, JavaScript, SQL
- **Statistics/Mathematics**: Stochastic processes, Monte Carlo methods, Markov models; statistical inference; linear time-invariant systems, Fourier analysis; familiar with theory and implementation of Kalman filters, artificial neural networks, and classification algorithms
- **Microscopy**: Total internal reflection fluorescence microscopy (TIRF); competent in optics theory, practical hardware experience (emission/excitation filters, dichroic mirrors)
- **Experimental Procedures**: Single-molecule kinesin protein motility assays; stopped-flow spectrofluorometer protein kinetics measurements

HONORS/AWARDS

- Summer Translational Cardiovascular Sciences Institute at Penn State, Fellowship (2013)
- Schreyer Honors Scholar, Penn State (2012 – 2015)

ACTIVITIES/MEMBERSHIPS

- Penn State Engineering Graduate Student Council, Webmaster (2014)
- American Society for Cell Biology, Member (2014 – 2015)
- Biophysical Society, Member (2013 – 2015)