A THEORETICAL MODEL OF CONFORMATIONAL TRANSITIONS IN BIOMOLECULES BASED ON SINGLE-MOLECULE FÖRSTER RESONANCE ELECTRON TRANSFER MEASUREMENTS

DIPLOMA THESIS

Department for Theoretical Physics, TU Berlin

SUPERVISORS:
PROF. DR. ECKEHARD SCHÖLL, PHD
INSTITUT FÜR THEORETISCHE PHYSIK, TU-BERLIN

DR. FRANK NOÉ
INSTITUTE FOR MATHEMATICS UND COMPUTER SCIENCE,
FU-BERLIN

submitted by

TORSTEN LÜDGE

Berlin, April 7th 2009

The fundament of life emerges on a nanoscopic scale where biomolecules change their conformation, aggregate and react with each other. One such process that has attracted particular interest by both theorists and experimentalists is the folding process, in which a biopolymer, such as a protein or an RNA compactifies into a relatively well-defined three dimensional fold in which it is biologically active. Understanding essential molecular processes such as folding is at the heart of understanding the mechanism of the cell.

Novel experimental techniques allow the dynamics of single molecules to be studied with nano to millisecond resolution over time periods of milliseconds to minutes. One such technique is the single-molecule Förster resonance energy transfer (FRET) where two dyes are attached to the molecule of interest, and a light signal is measured that allows to make conclusions about the distance of these dyes in time. This is done by exciting the donor dye at its absorption wavelength with an incident laser beam. Then, by the Förster resonance, an energy transfer between the two dyes may occur if the two dyes are close and an acceptor photon will be emitted, otherwise a donor photon will be emitted. The FRET efficiency, i.e. the ratio of donor photons at any time window allows to approximately compute the average distance during that time.

While the mapping of molecular species, i.e. the binning of distances observed in the data, and thus, e.g. estimating the probabilities of the folded and unfolded states is relatively straightforward, the extraction of kinetic information from single molecule FRET data is challenging. The reasons for this lies in the fact that the true dynamics of the system is high-dimensional, and the kinetically relevant free energy minima are only separable in high dimensions. When projecting this dynamics down onto the single distance measured in FRET, different free-energy minima overlap and the extraction of the number and definition of the minima and their interconversion rates becomes a hard inverse problem.

In this work, we propose to use Hidden Markov Models combined with either simple Gaussian output or linear stochastic differential equation (SDE) output as an approach to this problem. When the systems dynamics is in a free energy minimum this corresponds to a metastable state in the time series. The transition events between these different metastable states can be modeled as a (hidden) Markov chain, goverened by an (initially unknown) transition probability matrix. Depending on the hidden state, FRET-efficiencies are drawn from a Gaussian distribution with a certain mean and variance. In the SDE case, subsequent FRET-efficiencies are related by a simple diffusion in a harmonic potential well. Thus, the model can be described as a set of noisy harmonic oscillators coupled by a transition matrix. The unknown parameters (transition matrix, means, variances and diffusion constants) are estimated from the data via a maximum-likelihood method.

The method was applied to single molecule FRET trajectories of RNA folding collected in the laboratory of G. Ulrich Nienhaus at University of Ulm. The method shows to be useful in exhibiting a number of previously hidden and overlapping but kinetically separated metastable states in the data that could not be found by standard analysis methods. Furthermore, the estimated transition matrices provide information on the relative probabilities, and the lifetimes of the different conformations. Due to these results, we believe that, this novel approach to the analysis of single-molecule experimental data is destined to become a useful tool in biophysics.

Zusammenfassung: Die Fundamente des Lebens entstehen auf nanoskopischer Ebene, wo Biomoleküle ihre Konformationen ändern, sich aggregieren und untereinander reagieren. Einer dieser wesentlichen Abläufe, gleichermaßen interessant für Theoretiker und Experimentatoren, ist der Faltungsprozess, bei dem Biopolymere wie Proteine oder Ribonucleinsäuren eine kompakte Gestalt mit einer relativ wohl definierten Faltungsstruktur annehmen, welche dem Molekül seine Funktionalität verleiht. Diese grundlegenden Prozesse zu verstehen ist der Schlüssel zu einem umfassenderen Verständnis der lebenden Zelle.

Neuartige Experimente erlauben Beobachtungen von einzelnen Molekülen mit einer Auflösung von Nano - bis Millisekunden über einen Zeitraum, der von einigen Millisekunden bis zu mehreren Minuten reichen kann. Eine dieser experimentellen Techniken ist die Einzelmolekül Förster Resonanz Energie Transfer Methode (FRET), bei der zwei molekulare Farbstoffmarker an dem untersuchten Molekül befestigt werden, deren Lichtemissionen Rückschlüsse über den Abstand und über Abstandsänderungen der beiden Marker zulassen. Dies geschieht durch Erregen des Donor Moleküls mit einer Laserquelle im Bereich des Absorptionsmaximums des Donors. Anschließend kann Energie durch die Förster Resonanz Kopplung von dem einen Farbstoff zum anderen für den Fall übertragen werden, dass sich die Farbstoffe nahe beieinander befinden und ein Akzeptor Photon emittiert wird. Andernfalls emittiert der Donor die Erregungsenergie. Die FRET Effizienz, also das Verhältnis von Donor zu Akzeptor Photonen während eines gewissen Zeitraums, erlaubt die Bestimmung des mittleren Abstandes von Donor und Akzeptor innerhalb dieses Zeitraums. Während die Einteilung von Molekülunterarten, z.B. durch die in den Daten beobachteten Donor-Akzeptor Abstände relativ unkompliziert ist, indem aus den Häufigkeiten bestimmter Abstände auf die Wahrscheinlichkeiten von gefalteten und ungefalteten Zuständen geschlossen wird, stellt die Gewinnung von kinetischer Information aus Einzelmolekül FRET Daten eine Herausforderung dar. Gründe dafür sind die hochdimensionalen Räume, in denen sich die Dynamik abspielt und kinetisch relevante, freie Energieminima nur dort separierbar sind. Wird diese Dynamik auf eine eindimensionale Abstands- oder FRET-Effizienz-Trajektorie projiziert, überlagern sich verschiedene Minima der freien Energielandschaft, und die Berechnung von einzelnen Minima und ihrer Austauschraten wird zu einem schwer lösbaren inversen Problem.

In dieser Arbeit schlagen wir für die Bewältigung dieser Aufgabe ein Hidden Markov Modell, kombiniert mit entweder Gaußförmigem Output oder stochastischen Differentialgleichungen vor, die den Output bestimmen. Wenn sich das System in einem Minimum der freien Energielandschaft befindet, entspricht dies einem metastabilen Zustand in der Beobachtungszeitreihe. Die Übergänge zwischen verschiedenen metastabilen Zuständen können mit Hilfe von (verborgenen) Markov-Ketten modelliert werden, die durch die anfänglich unbekannte Übergangswahrscheinlichkeitsmatrix bestimmt werden. Abhängig von dem verborgenen Zustand werden FRET-Effizienzen aus einer durch Mittelwert und Varianz bestimmten Gauß-Verteilung gezogen. Im Falle der stochastischen Differentialgleichung werden aufeinander folgende FRET-Effizienzen durch einfache Diffusion in einem harmonischen Potential verknüpft. Das Model kann also als ein System von gekoppelten harmonischen Oszillatoren aufgefasst werden, deren Kopplung durch die Übergangsmatrix bestimmt wird. Die unbekannten Parameter (Übergangsmatrix, Mittelwert, Varianz und Diffusionskonstante) werden durch die Maximum-Likelihood Methode aus den Daten geschätzt. Diese Herangehensweise wurde auf die Einzelmolekül FRET-Trajektorien einer sich faltenden Ribonucleinsäure angewendet, welche im Labor von G. Ulrich Nienhaus an der Universität Ulm aufgenommen wurden. Die Methode hat sich als nützlich erwiesen, um weitere, zuvor unbekannte oder unerkennbare, metastabile Zustände in den Daten aufzudecken, die mit den Standard - Analysemethoden nicht zu erkennen sind. Des Weiteren liefert die Übergangsmatrix Informationen über die relativen Wahrscheinlichkeiten und Lebensdauern von verschiedenen Konformationen. Die Ergebnisse lassen den Schluss zu, dass dieser neuartige Analyse-Ansatz für die Auswertung experimenteller Einzelmoleküldaten zukünftig eine zunehmend größere Rolle spielen wird.

Originality Statement

’I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at Technische Universität Berlin or Freie Universität Berlin or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project’s design and conception or in style, presentation and linguistic expression is acknowledged.’

’Ich erkläre hiermit an Eides statt, dass ich die vorliegende Diplomarbeit sebständig und ohne unerlaubte Hilfe angefertigt, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.’

Signed:____

Contents

CHAPTER 1

Introduction

The undreamt-of breakthrough of molecular biology has made the problem of the origin of life a greater riddle than it was before: we have acquired new and deeper problems. — **Karl R. Popper**[1]

1.1. Outline

The purpose of this thesis is to understand and describe the folding dynamics of a small biological molecule. A historical contemplation in the first chapter will recollect theories and methods that lead to our current state of the art in molecular microscopy, data processing and the description of processes evolving in living systems as for example the cell. The experiment that collected the data of a single molecule folding process, the molecule itself and its crucial function in living organisms will be described in the second chapter. Theoretical principles required for the description of molecular dynamics are probability calculus and stochastics, which are treated in the third chapter together with the basic photo-physical principles of the investigating process and applied to the available data for analysis in chapter four. Difficulties in converting and interpreting the raw data are also mentioned in the anylis chapter. Concluding remarks and an outlook are given in the last chapter. The apendix lists the algorithmic procedures used for the analysis and a summary of the data that has sucessfully been analyzed.

1.2. Physics and Life

Biological systems have a long tradition of being studied by physicists who are using mathematical methods to describe biological phenomena from physical principles. First recorded attempts reach back to antiquity and accompany the medical development at all times (the historically interested Reader may find [1] and the references therein useful). Nevertheless the first *physical* method for this purpose based on mathematical equations was developed by Edward N. Lorenz in the 1960's [2], which originated the chaos theory. Investigations on whole populations of living creatures and their dynamics using computer routines were done in 1976 by Robert May who simulated a fish population P possessing the growth rate R using the formular $\mathrm{P(new)} = \mathrm{R} \cdot \mathrm{P(old)} \cdot (1 - \mathrm{P(old)})$ containing a ressource limit

[1] "Reduction and the Essential Incompleteness of All Science," p 259-284, Studies in the Philosophy of Biology, Francisco Jose Ayala and Theodosius Dobzhansky, eds. University of California Press, 1974. p 271.

through the $1 - \mathrm{P(old)}$ term. In spite of its simple form the formula already exhibits nonlinear (or chaotic) effects e.g. for a start population of 2% and a a growth rate of $3.8\ \frac{\mathrm{Individuals}}{\mathrm{Generation}}$ or higher.

The physical view on the smallest units of life, however, made its major progress in the last decade. Through the massive improvement of experimental design and precisions enabled through tunable optical lasers, single photon counter, extremely sensitive nuclear magnetic resonance spectroscopy, high performance computer clusters, etc, we gained insights into many biological systems in a quantitative manner and with high predictability. Despite their recent developement, new branches in scientific research emerged globally called “Biophysics”, “Bioinformatics” or ”Life Science” and quickly gained a diverse communitiy consisting of biologists, chemists, physicists, physicians, mathematicians and programmers.

Facing the huge amount of data generated by new experimental techniques covering all time scales from femto seconds to years, biophysicists often are in need of efficient analysis methods that sufficiently reduce their data to reveal the specific physical property of interest. A particular tool for investigations in biological cells and their environment is the *single molecule* Förster Resonance Energy Transfer (smFRET) spectroscopy, which was also used in its particular two fluorophore version in this thesis for data acquisition described in section 2.3. Analysing the hereby acquired collection of two-channel photon streams demands a stochastic treatment of the time series through *Hidden Markov Models* (HMM. see section 3.3) or *stochastic differential equations* (SDE, chapter 3.4.2). A major obstacle in the analysis of single molecule dynamics and their measurements under biologically relevant conditions is the separation of unwanted but inevitable effects (such as background noise, saturation or photophysical effects as the triplet absorption (blinking) or bleaching) from the desired signal, i.e. the photon encoded distance trajectory of two distinctive areas of a macro molecule. The FRET spectroscopy method is available since 40 years, yet mostly applied to bulk measurements of whole molecule ensembles. It has only been a few years ago that measurements and observations at the single molecule level are possible . For biologically relevant conditions, experiments require room temperaturs and aqueous solutions of the molecules, which often do not allow further experimental reduction of noise without obscuring the outcome of the measurement. Additionally, for obtaining a prolonged observation duration, low-frequent observation events in turn fundamentally broaden the results by means of Poissonian distribution (e.g. ”shot noise”) [3].

The herein investigated properties are the conformational fluctuations within and amongst the conformational states of a single surface-immobilized, enzymatic RNA molecule, the Diels-Alderase.

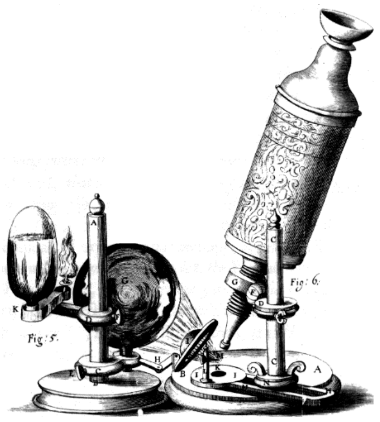
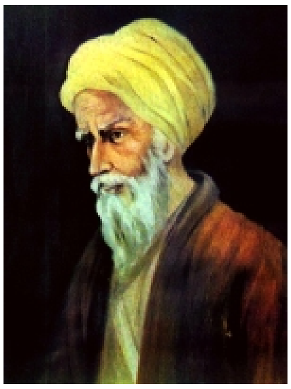


FIG. 1.1. Early compound microscope invented by Anton van Leeuwenhoek to look at cells, bacteria

1.3. Evolution of Microscopy

The physical fundament of the magnifying effect when looking through convexly formed, transparent material is diffraction, which in turn is dependent on the speed of light dependency on the optical density of the medium it is travelling through. The refraction angle of a light beam entering a medium with different optical densities was described by Fermat's *principle of least time* in 1657 [4]and, indeed, this is known as the first elaborate aplication of an extremal principle in physics since the medieval times. The resulting equations were the same as discovered independently by Willebrord van Roijen Snell (Snell's law) and Descartes in 1618 but may have probably first been mentioned in the "Book of Optics" by the Iraqi Muslim scientist Ibn al-Haytham (in Europe latinized as Abu Ali Hasan Ibn Alhacen or Alhazen, see Fig. 1.2.) in the year 1021 AD. However, 3000 years BC, the ancient Egyptians and Mesopotamians already used and constructed lenses (referred to as "reading stones") with the desired magnifying property, but it wasn't until 1595 that the first true microscopic device was made in Middelburg, Holland manufactured by three eyeglass makers: Hans Lippershey (developer of the first real telescope), Hans Janssen and his son Zacharias [5]. In the following 30 years, various names were used for the magnifying device, as for example Galileo called it the "occhiolino" or "little eye", until finally Giovanni Faber coined the name "microscope" (from the Greek: $\mu\iota\kappa\rho o\varsigma$, mikros, "small" and $\sigma\kappa o\rho\epsilon\iota\nu$ skopein, ”to look” or ”see”,). For further details the historically interested reader can explore the contents of the online `http://www.mikroskop-museum.de/` microscope musem.

With the first compound microscopes (see Fig. 1.1.) at hand (since circa 1595), new insights into organic, mineral and other domains, previously having been inaccessible to the eye, opened up whole new dimensions of investigating nature (as evolved simultaneously in telescope-enabled astronomy), since suddenly bacteria, blood and sperm cells, protists like amoeba, etc, could be observed, analysed or manipulated (accordingly in astronomy, moons, rings and phases of extraterrestrial


(a) The islamic physisics Alhazen (965-1040)

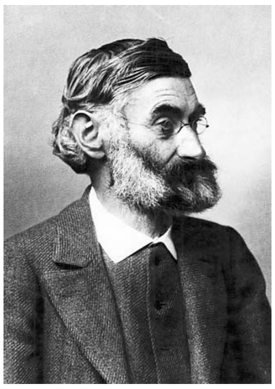

(b) Ernst Karl Abbe (1840-1905)

FIG. 1.2. (a) Author of the first preserved published material about light rays in 1021: the seven volume treatise "*Book of optics*". (b) Founder of the first mathematical foundation of microscope design essential for the later inventions of the Carl Zeiss AG company.

planets, asteroids and nebulae were discovered). A better understanding of optics made it possible to minimize chromatic aberration of the projection lenses and maximize magnification and lead to a better understanding of light possessing a finite velocity (Ole Christensen Rømer 1676), propagating as a wave (Christian Huygens 1678) or beeing polarizable (Augustin-Jean Fresnel, François Arago). Even foreign domains in science were able to benefit from the new devices as for example Robert Brown draw major conclusions while looking through a microscope in 1827 [6]. He observed the permanent movement of pollen in aqueous dispersion, which Einstein later concluded to be clear evidence for the material existence of atoms and molecules[2] in his famous publications in 1905 [7]. The triumphal procession of looking closer and closer into the universe, seemingly, came to an end when Ernst Karl Abbe (see Fig. 1.2) discovered the fundamental diffraction limit for the smallest resolvable diameter in 1872 [8]. This limit, d_{limit}, of visual light microscopy, is completely determined through the wavelength of the light λ, the refraction index n of the lens and the apterture angle α:

$$d_{limit} = \frac{\lambda}{2n \sin\alpha} .$$

Even though light diffraction is the key to magnify the world, it also causes its downfall at the end of the magnification scale. George Airy (an astronomer in the early 19th century) found that any light beam passing a (circular) aperture has a minimum area at the optimal focus creating a minimum focal spot [9]: The airy disc (see Fig. 1.3.). He found that the minimum diameter can be calculated as a linear function of the aperture angle (ratio of lens diameter to focal length) and the

[2] Philosophers like Mach and Oswald and their followers before Einstein regarded atoms as a purly abstract concept without acutal manifestation in reality and, thereforee, had to retreat and develop new views and perspectives after his proof of existence.

wavelength from the intensity derived from the Fraunhofer diffration equation as

$$\mathrm{I}(\theta) = \mathrm{I}_0 \left(\frac{2J_1(\mathrm{k} \cdot \mathrm{d} \sin\theta)^2}{\mathrm{k} \cdot \mathrm{d} \sin\theta} \right) \tag{1}$$

where $\mathrm{I}_0 = \mathrm{I}(0)$ is the maximum intensity of the pattern at the center of the disc,

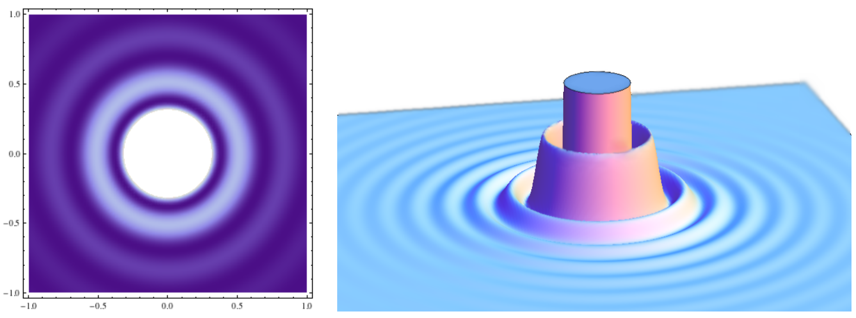


FIG. 1.3. Light intensity distribution of the Airy disc from a focused point light source: *left* as seen with the eye, picture generated with Eqn. 1, *right* the light intensity as function z(x,y)

J_1 denotes the Bessel function of the first kind and order one, $\mathrm{k} = {}^{2\pi}/_{\lambda}$ is the wave number and d and θ denote the two-dimensional spherical coordinates radius and observation angle measured from the image-perpendicular, respectively.

An infinite number of rings exists around the inner disc corresponding to the second maximum, third and so on. The angle θ at which one finds the first intensity minimum is the first zero of J_1, hence at $\mathrm{k} \cdot \frac{\mathrm{d}}{2} \sin\theta \approx 3.817$, which yields the desired linear relation between $\sin\theta$ defining the resolution limit for the wavelength λ

$$\sin\theta = \frac{3.83\lambda}{\pi \cdot \mathrm{d}} = 1.22\frac{\lambda}{\mathrm{d}} \tag{2}$$

Today, the resolution limit assigned to every diffraction-limited device is the Rayleigh criterion which can be formulated using Eqn. 1. It defines half the Airy radius as the smallest displacement of two microscopic objects for being detectable. Any two objects with a distance smaller than ${}^{d}/_{2}$ are perceived as one. One possible way to circumvent the Abbe limit is to change the imaging technique to scanning the media with invisible (e.g. ultra violet) light of lower wavelength or sources other than electromagnetic waves. This, for example, is done by using charged particles in the electron microscope which was co-invented by Ernst Ruska in 1931 (Nobel laureate since 1986), by using needles with mono-atomic cusps to raster a surface with an atomic force microscope, by diffraction imaging through X-rays, or by using the quantum mechanical tunnelling effect in the scanning tunnelling microscope, invented by Heinrich Rohrer and Gerd Binnig at the IBM research laboratory in Rüschlikon 1981. These methods have illuminated the microscopic world to a highly detailed view on cells and organisms. Unfortunately, their experimental conditions do not allow living samples to be analysed ("in vitro" microscopy) because of the high energy applied to the sample. Nevertheless, the insights gained through the methods mentioned before allowed crucial comprehension without which the following inspections would not have been possible.

Another intuitive, though physically incomprehensible explanation for an optical limit is that objects smaller than the wavelength of light (e.g. 400nm) are unable to reflect the illuminating wave, thus appear as transparent or at least blurred. For biophysical research, the scale of interest is the diameter of a single cell and beyond, typically 5-10 $\mu\mathrm{m}$ for eukaryotes (cells bearing a nucleus) down to 1-2 $\mu\mathrm{m}$ for bacteria. A visual image of a single fluorescent molecule would take 20% of a cells size and, hence, is unsuited for obtaining structural information about its components as i.e the eucaryotes nucleus with about one tenth of the cells size. To overcome the diffraction limit, microscopes based on a different fundament than the conventional ”in focus” image generation process were needed and invented as, for example, happened with the confocal microscopy techniques (see Section 1.5.1) by inverting the usual microscope setting. This technique is no longer able to project images onto a screen or directly into the eye and is therefore somewhat abstract, nevertheless it allows deeper insight into the smallest compartments of living systems than optical methods can provide.

1.4. Breaking the Diffraction Limit

Prototypes of sub diffraction microscopes have been constructed since the beginning of the 20th century allowed the mechanical precision necessary for adjustment and focussing. A prominent example is the "ultra microscope" invented by the colloid chemist Richard Adolf Zsigmondy and the Zeiss instrument maker Henry Siedentopf in the year 1905 [10]. It is based on the light scattering effect that particles exhibit if irradiated at a wavelength comparable to their diameter. The effect of electromagnetic radiation emitted perpendicular to the irradiation beam is treated in the *Mie solution* of Maxwelll’s equation (an analytical solution of electromagnetic radiation of sperical particles) and is named after the discoverer John Tyndall. The "Tyndall cone" for example produces the clearly visible laser beam observed when the laser path traveres a gaseous suspension of collodial particles (e.g. smoke particles or fog droplets). The special characteristics of the Mie solution are used in the ultramicroscope to infere the position of the scattering particle at a very high precision (about a third of the diffraction limit). It is, however, indispensable for observation on non-collodial immersions (i.e. crystal-structures or cell-membranes) and for even higher resolution to resort to a different imaging technique other than light projection and intensity measurements, since Abbe’s limit is of fundamental nature. In current research, another circumvention of the Abbe limit is discussed by employing negative refraction index lenses (the so-called ”perfect lens” made of customized metamaterials), but, until now, it is lacking any experimental evidence.

In the last two decades, techniques and methods became available to experimentalists, which were able to obtain visual structures way beyond the Abbe limit. An extensive exploration of state-of-the-art imaging systems is given in [11]. A few other attempts were made to visualize cell compartments passively, as for example the phase-contrast technique developed by the physicist Frits Zernike in 1932 [12], but the majority of images gathered today on sub cellular scales employ an old idea, which is somewhat similar to the one that enabled Heinrich Willhelm Waldeyer in 1888 to observe what he then coined as the chromosomes. Chromophores (colorant particles or molecules) are the key to improved resolution: without staining (applying the marking colorant), most cell constituents appear transparent, hence invisible to the eye. Through a highly specific binding affinity, a molecular dye (aniline colorant in the case of the chromosomes) binds either to a specific or unspecific component of the cell, making it visible and distinct from its surroundings. Experimenters in today's laboratories as Stefan Hell, director of the Max Planck Institute for Biophysical Chemistry in Göttingen and developer of the STED (Stimulated Emission Depletion-technique) [13] are equipped with a sizable catalog of dyes, laser sources, single photon counters, precise filter sets and further optical devices enabling them to produce beautiful images at a resolution that captures single molecule turnovers in biochemical reactions. Other scientists have performed single step motion-capturing of i.e. the myosin/kinesin - actin muscle mechanics or to track single viruses along their infectious way from the cell outside membrane into its nucleus. The method of confocal fluorescence microscopy as depicted in Fig. 1.4. will be explained in detail in the following section.

1.5. Fluorescence microscopy

1.5.1. Confocal Microscopy. The focusing optics of a confocal microscope is, in principle, the same as in a usual microscope, the major differences are the inverted position of the sample and the light source, an additional beam splitter and a pinhole at the conjugated focal spot (B_1 in Fig. 1.4.). Only the fluorescent light coming from the very small (down to femto-litre) focal volume reaches the detector (*orange lines*), whereas fluorescent light from other parts (*dotted lines* in Fig. 1.4.) is blocked by the second screen. Moving the focal spot across the sample generates a surface image (*XY-plane*) and, additionally, changing the focal length (*Z*-axis) can generate a three dimensional image as long as the exciting and fluorescent light can travel through the fixed sample.

The so-called confocal microscopy was pioneered by Marvin Minsky in 1955 [14]. Whereas for the first cell compartment microscope images, physicians simply had to pour their dyes into the Petri dish, dilute and mix it with the object under study in order to stain their desired substance, the today's task of attaching a photo-active dye to a single molecule often determines the most expensive task (in terms of

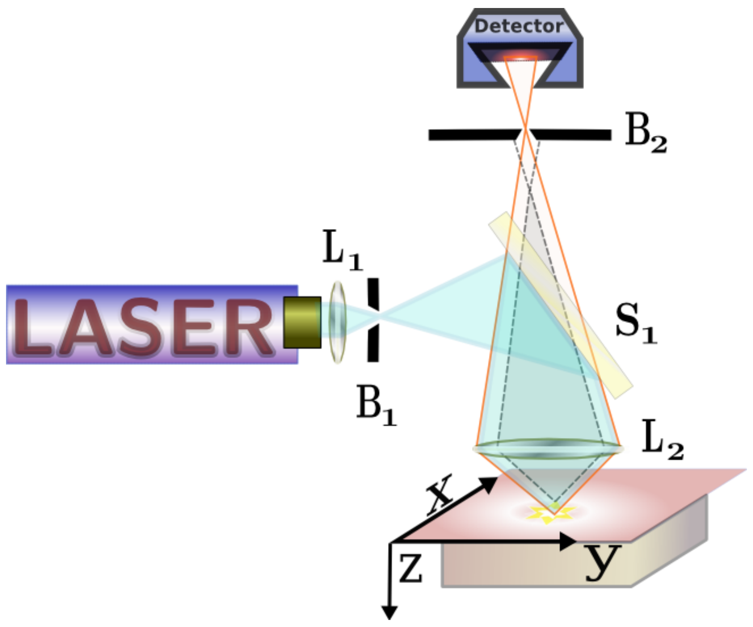


FIG. 1.4. Confocal microscopy: the first lens L_1 disperses the Laser light onto lens $L_2$with a high numeric aperture. The fluorescent light (red line) passes through the dichriotic mirror S_1 and blind B_2. Fluorescent light from outside the focal spot doesn't reach the photon detector (dotted lines).

time, knowledge and/or money) in the imaging process. Additionally, a maximum purification is needed to exclude excitation and fluorescence of anything else than the dye-molecule (in the single molecule context and in the following called fluorophore). If accomplished and the fluorohpore's precise position at the investigated molecule is known, one can infer from the recorded photon traces to the structure of the object under study. The explicit inference procedure to yield spatial quantities as e.g. the distance in nanometers heavily depends on the experimental setting which, in return, depends on the question one is asking about the molecule. As mentioned before, the signal obtained from confocal microscopes has to be transformed by means of computational reconstructions that account for the laboratory conditions and can implement noise filter and error corrections. The calculation of distance trajectories from the measured photon stream are usually carried out by numerical algorithms either implemented in the microscope or applied to the recorded photon traces afterwards on a computer as in the present case.

A common principle of all techniques is that the properties of the photo-active fluorophores (such as photon emission, absorption or polarization) are influenced by the molecule they are attached to and change as the molecule changes. The fluorophore (or dye) reports about the current processes in form of photon emissions. Together with a physical model of the conformational dynamics, the analysis of the photon stream can give direct conclusion about the molecule under study. For example, the chronological changes in confirmation, inhibitor binding etc. can be determined. To the current date, there are 22 types [15] of microscopes using the same principles for various purposes. In the case of the subunit rotation in F_0F_1-ATP synthase (for details on this molecular rotary engine see [16]), it was shown that the state changes due to fluorescence measurements enabled the authors to ultimately decide between competing hypotheses [17].

CHAPTER 2

Experimental Setup

2.1. The Single Molecule Microscopy

In the following chapter the experiment which was employed to obtain the time-series data will be described, and a short introduction to the sample under investigation and it's background in the biological context will be given. Photo-physical processes during the experiment and their treatment towards error correction are described thereafter and, finally, the connection between the experiment and the desired conformational analysis will be given.

2.1.1. Microscope and Sample. Through inversion of the light-beam course, it is possible to convert a classical microscope into a confocal one with only a few changes in positioning lenses and the object as shown in Chapter 1, Fig 1.4 .

The general scheme of the experiment providing the data for this thesis is depicted in Fig. 2.1, it was realized at the Institute of Biophysics, University of Ulm, Germany. For a detailed description of the procedure of sample preparation and a list of all

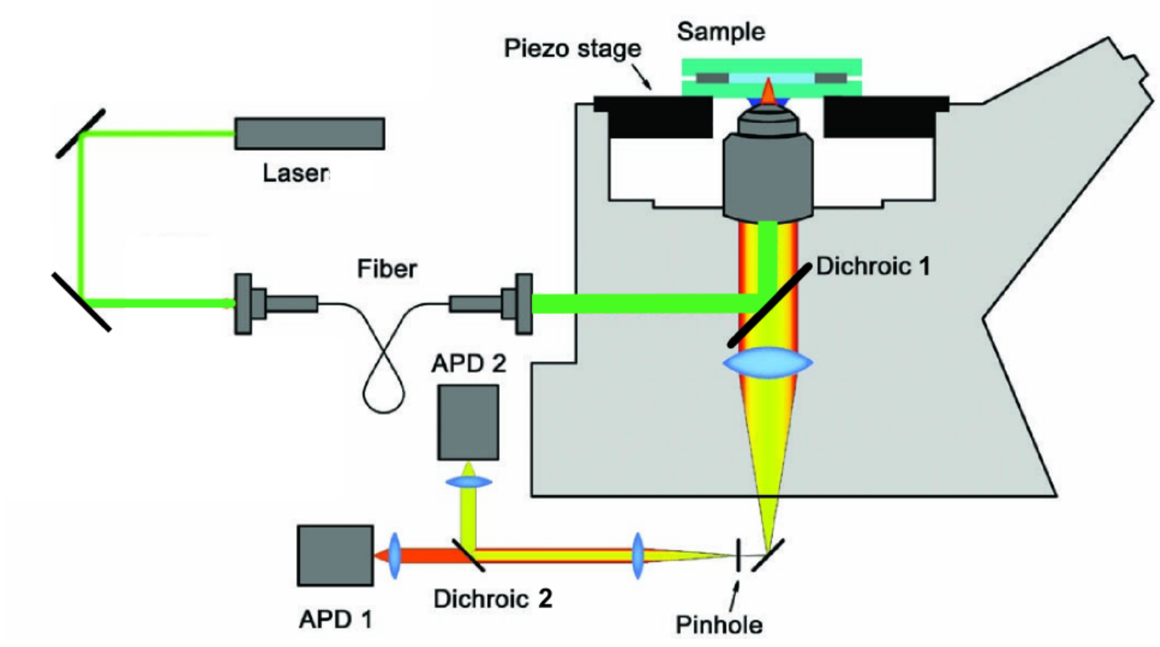


FIG. 2.1. Confocal microscope for signal acquisition of single molecule Förster resonance energy transfer in the sample. APD: avalanche photo diode. The green laser light illuminates the sample and fluorescent light is collected, directed through the filtering pinhole and passes the dichriotic beam splitter to be focused onto the two color detectors. Picture taken from [18].

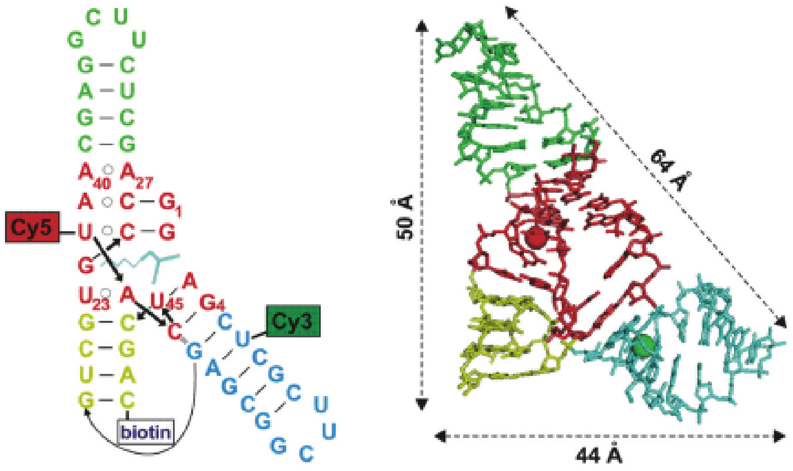


FIG. 2.2. Structure of the Diels Alderase sample: *left,* base pairs with marked positions of the donor (Cy3) and acceptor (Cy5), gray lines mark the biotin surface connection, *right,* tertiary structure displaying the λ-shape of the sample, fluorophores are symbolized as colored balls. Colors are arbitrary for better recognition. Picture taken from [19]

devices, their manufacturers used and calibrations, please refer to Adrei Yu. Kobitski et al [19].

2.1.2. Surface Immobilization of a Single Diels Alder Ribozyme. In contrast to the optical setup of confocal microscopy, the preparation of *single* molecules is a challenging task since it involves labeling and tracing the object heavily affected by Brownian motion. The huge advantage of surface immobilized spectroscopy for single molecule observations is that at a focal spot volume of a few femtolitres (only several diameters of the ribozyme), whereas a freely diffusing molecule would pass that volume within microseconds and would rarely be seen at all. In the immobilized case, the continuous recording times can reach minutes or even hours. The attachment of the molecule to the sample plate (usually a glass cover-slip) is accomplished through biotin-streptavidin inter-linkage: the biotinylated ribozyme (see Fig. 2.2) is flushed onto the cover-slip bearing a sparse distribution of streptavidin ends, which the modified RNA can bind to. The cover-slip thereby gets populated sparsely with immobilized ribozymes.

The single-molecule fluorescence emission from the immobilized molecule with attached reporter dye (Cy3) was stimulated using the green 514.5 nm line of an Argon/Krypton-ion laser. The acousto-optically tuned laser light illuminates the sample plate continuously within an area of 30×30 $\mu\mathrm{m}^2$ at a resolution of 128×128 pixels, with 5 ms dwell time for each pixel, until the emission from 500 to 1000 single molecules was collected. The diffraction-limited focal spot of the exciting laser is held stationary, whereas the sample plate is shifted using a piezoelectric stage scanner. For automatic detection of a labeled ribozyme moving into the focal spot, direct excitation of the acceptor dye (Cy5) at 633 nm was provided by the secondary He-Ne laser, though only used for calibration of the acceptor emission intensity beforehand. The sample plate is movable in the XY-plane (as depicted in Fig. 1.4) such that the focal spot can be centered at one molecule. For recording a continuous stream of the donor/acceptor pair, the focal spot is moved to the next molecule starting a new trajectory record until all immobilized molecules have been scanned. The emission was collected through a water-immersion objective into a two-channel single photon signal using a beam splitter at 640 nm. After passing through separate filters optimized for donor and acceptor emission, the arriving photons were recorded separately from two avalanche photo diodes, hence sorting all photons with wavelengths between 555 and 610 nm into the 'green channel', and between 650 and 750 nm into the 'red channel' (see Fig. 2.1). The same areas were first scanned for each pair with green then with red laser excitation, so that only those molecules that contained a functional pair of labels were included in the subsequent FRET efficiency analysis. Then, the fluorescence intensity was excited with continuous green light and recorded with a 1 ms time resolution until photo bleaching of donor and acceptor occurred. In the final analysis of state interchange rate coefficients, only those time traces were included that showed single-step bleaching. The immobilized ribozyme is continuously illuminated with the excitation laser source. Fluorophores exposed to an "over dose" of exciting photons is bleached temporarily or irreversibly (photo destruction), which is causing obscuring effects. Therefore, measurements should be done at lowest possible excitation intensities.

2.2. Exciting Processes

The process of a fluorescence measurement can coarsely be described as follows:
1. Excitation of the fluorophore through high energy laser photon (blue) absorption.
2. One of the following physical events may happen (see Fig. 2.3) :

- Immediate re-emitting of slightly lower energy photons, called green fluorescence (circled number 2 in Fig. 2.3).
- Inter-system crossing from S_1 to T_1, or from T_1 to the ground state with a much longer life-time due to Pauli's exclusion principle (phosphorescence, blue wave in Fig. 2.3). Often 80% of all transfers are inter-system crossings.
- Non-fluorescence transfer of the exciting energy towards near by molecules through non-radiative dipole-dipole resonance.
- Transfer of the exciting energy through electron transfer (quenching of fluorescence)
- Dissipation of the exciting energy into heat or infrared radiation (leakage)

Since the fluorophore pair is covalently attached to the investigated molecule, it changes properties as for example polarization and also the Förster radius R_0 (due to a change of extinctionε_a, orientational coefficient κ^2 and the refractive index n, see

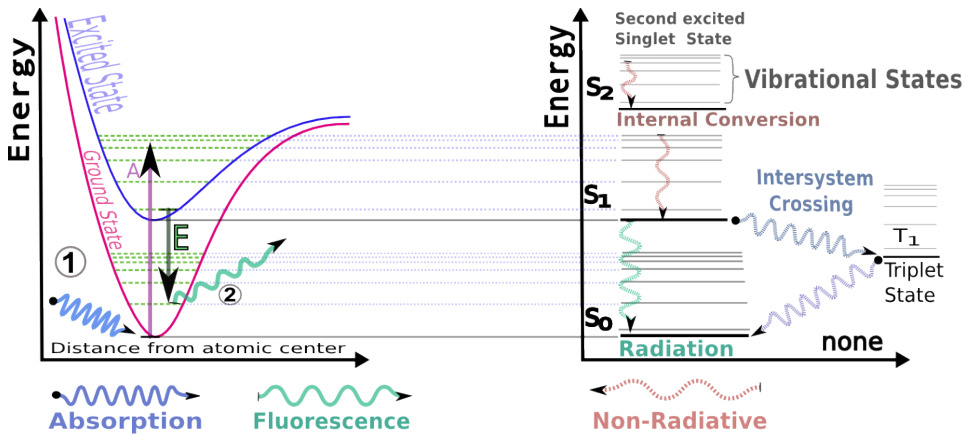


FIG. 2.3. Energy diagram for fluorescence measurements: *left,* Lennard-Jones Potential with two excitation states shown (ground state S_0 and excited state S_1). (1) Excitation (blue wave) elevates the electron in the ground state to a vibrational sub-level (green dashed lines) of the excited state. (2) Relaxation to the ground state with an exponential decay rate and fluorescent energy emission E. *right,* Typical Jablonski diagram with arbitrary x-axis and a second excited State S_2 and a triplet state T_1. Transfer of exciting energy between vibrational levels within one state are non-radiative. Inter-system crossing $\mathrm{T}_1 \rightarrow \mathrm{S}_0$ yields phosphorescence.

Eq. 38) as the molecule changes (e.g. its conformation). The basis for three major detection methods employed in fluorescence analysis and listed below are Förster resonance energy transfer (FRET) measurements which will be described in the next section :

(1) Time-correlated fluorescence microscopy (TFM) measures the relaxation time of the re-emitted photon. The polarization response can be measured additionally.
(2) FRET microscopy detects the spatial change of two or multiple fluorophores attached in a proximity radius of 10-100 to each other.
(3) Alternating Laser Excitation (ALEX)-FRET also excite the acceptor and ”check” which way the photon decided to take, hence filtering FRET fluorescence from direct excitation fluorescence.

All methods rely on the dye quality as their properties (or rather knowledge about their properties) determine the accuracy of measurement and therefore all analysis steps that follow. Those properties can be manipulated in various ways in order to reveal specific features of the molecule. A good fluorophore should have an emission spectra as depicted in Fig. 3.6 on page 37 later on so that FRET occurs efficiently (determined by the spectral overlap of the acceptors absorption with the donors emission spectrum) to finally emit an adequate amount of lower-energy acceptor photon emission (rad balls in Fig. 2.4).

The process can also be seen from the point of view of exciting state lifetimes : The intrinsic lifetime of an excited donor competes against the distance-dependent lifetime of the donor-acceptor complex: At close proximity the latter wins and the exciting energy is emitted via the acceptor, whereas in the distant case, the lifetime of the complex is much longer, therefore, the energy will be re-emitted from the donor within its decay characteristics.

2.3. Foerster Resonance Energy Transfer (FRET)

The energy transfer efficiency (sometimes also considered a transfer rate) is defined through the ratio of acceptor photons to acceptor *and* donor photons within a given discrete time window $\tau = t_{end} - t_0$:

$$E_\tau = \frac{I_A}{I_A + I_D} \tag{3}$$

where the intensities I_A and I_D are the sum of acceptor photons ($A \subset \mathbb{N}$) and donor photons ($D \subset \mathbb{N}$), respectively collected in the time-widow τ at each point in time:

$$I_A = \sum_{t_i=t_o}^{t_0+\tau} A(t_i) \quad , \quad I_D = \sum_{t_i=t_o}^{t_0+\tau} D(t_i) \tag{4}$$

The process of regaining the desired efficiency *trajectory* $E(t)$ from of the single photon traces is described in detail in Chapter 3, section 4.1.1 because it involves a considerable numerical and methodological effort.

Let's assume that there is an efficiency trajectory $E(t) = \frac{I_A(t)}{I_A(t)+I_D(t)}$ present that reflects a naive FRET situation. Theodor Förster published his book on luminescence in 1952 presenting a formula that relates the efficiency of a donor/acceptor pair to the distance between them [20]. At the distance where donor and acceptor are emitting equally intense, the efficiency is 0.5 defining the system-specific Förster radius R_0. The simplest form of the Förster law (for the full formula see Chapter 3, section 3.6.1 on page on page 36) can be written as an inverse law of sixth order in distance with the system-specific constant R_0 or as ratio of the lifetimes of a single donor (τ_D) and the donor-acceptor complex (τ_{DA}) at distance r :

$$E(r) = \frac{1}{1 + \left(\frac{r}{R_0}\right)^6} = 1 - \frac{\tau_{DA}(r)}{\tau_D} \tag{5}$$

2.4. Correction of Photon Trajectories

All measurements made with the described settings end with a special procedure in order to determine the environmental factors for the background noise (uncorrelated photons from the molecule or its surroundings) and crosstalk (donor photons that are mistakenly counted as lower-energy acceptor photons). After the active phase of the fully functional molecule (left panel in Fig. 2.5) the acceptor-dye is bleached

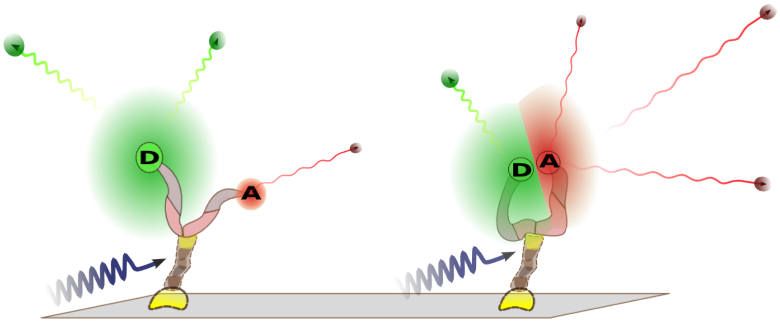


FIG. 2.4. Two exciting energy resonance scenarios: *Right*: high transfer efficiency for relative distances of donor and acceptor below 10 , the acceptor will emit most of the exciting energy (blue wave) as red photons (red balls with wave tail). *Left:* low transfer efficiency when further apart, the energy will mostly be converted to green photons (green balls) from the donor.

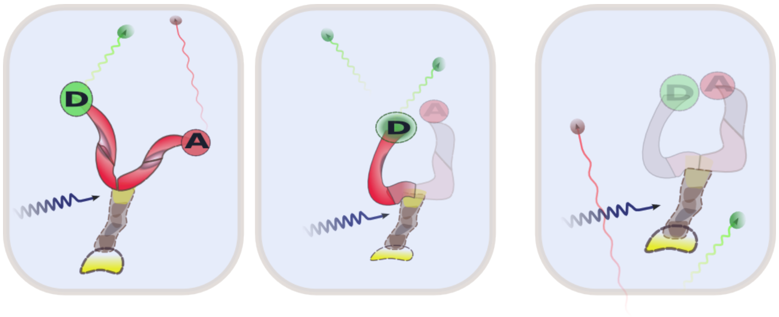


FIG. 2.5. Successive bleaching of the continuously excited (blue wave) RNA sample, *left:* Active phase with functional donor/acceptor pair. *Middle:* Acceptor (grayed out) is bleached leaving only donor emission (green balls with wave tail) as non-thermal dissipation path. *Right:* Both dyes are bleached, hence are non-fluorescent, all photons detected are due to background noise.

by the experimenter. This closes the channel for the donor to convert his excitement via FRET towards the acceptor (middle panel in Fig. 2.5). Therefore, an increase of donor intensity should follow (otherwise, the system isn't behaving accordingly and the recorded photon stream should be disregarded). The donor, finally, is bleached to record the pure background noise (right panel in Fig. 2.5). However, the signal may not always exhibit the expected phases (e.g. due to impurities in the sample solution), which is why it sometimes may not be possible to clearly distinguish the active phases from the two bleaching phases.

2.5. RNA Folding Kinetics - Conformational Heterogeneity

The ability of a molecule to remain in a particular spatial or functional state is often described as conformational heterogeneity. The conformations are seen as ”states” (a particular fold of the molecule as e.g in Fig. 2.6) that persist through a certain time-span and can be triggered or terminated through secondary processes such as inhibitor binding or changes in ion concentration in the environment. The conformation states however are not necessarily visible in the one-dimensional projection that the Förster law offers. The trajectory generated from the photon counts with the help of that equation and the procedure described before may provide an intuitive understandable measure, but to think that these trajectories are representative for the dynamics governing the system is by far misleading. The states are defined in the high dimensional space of conformational arrangements (folds) which can be described (and simulated) only with huge parameter sets, such as dihedral angles, mean forces between *each* base-pair, spatial coordinates of each base-pair, charge, polarization, and even quantum mechanical properties. The key to identify a state is it's dwell time compared to the overall fluctuation of the molecule. The fluctuation on the shortest timescale are due to the thermally stimulated random movement of each compartment of the molecule. In a homogeneous situation, the dynamics of the molecule would completely be described by those externally driven forces without distinguishable structures. However, measurements on the smallest constituents of living organisms (such as proteins, enzymes, nucleic acids etc) indicate that fundamental concepts of life are based on the opposite: only a few out of the enormous number of geometrically possible conformations are observed with non-random frequency, and their existence often complies with functional relations. These sets, through their 'almost' stable behavior are named *metastable states* (for a mathematical description in terms of eigenvalues see 3.3.2)

Every possible arrangement of the molecule posses a specific amount of free-energy from the mean Coulomb-forces that are acting between the atoms. If other configuration yield a lower free-energy situation, the molecule will change towards the lower-energy conformation if no energetic barriers are preventing the molecule from getting there. Most molecules possess conformational states that are characterized by metastable behavior originating from free energy minima. A computational approach as undertaken by Zhan et al. reveals the energy landscape of our enzymatic ribozyme during the catalysis of a Diels-Alder reaction (see Fig. 2.6). To infer this landscape given only one-dimensional efficiency trajectories is the hard inverse problem mentioned in the abstract. A method that can achieve this or is at least able to decide between competing model propositions is a tempting outlook we would like to approach here.

Conformational states are not only found in large proteins or protein complexes, in the case of the very small catalytic ribozyme (only 49 base-pairs in total, countable in Fig. 2.2) investigated in this thesis; folded, unfolded and intermediate states have been distinguished by Nienhaus et al. [19] with populations dependent on the concentration of divalent magnesium ions (Mg^{2+}). As will be shown in the analysis section, past and recent studies only gain little insight in the kinetic behavior, meaning

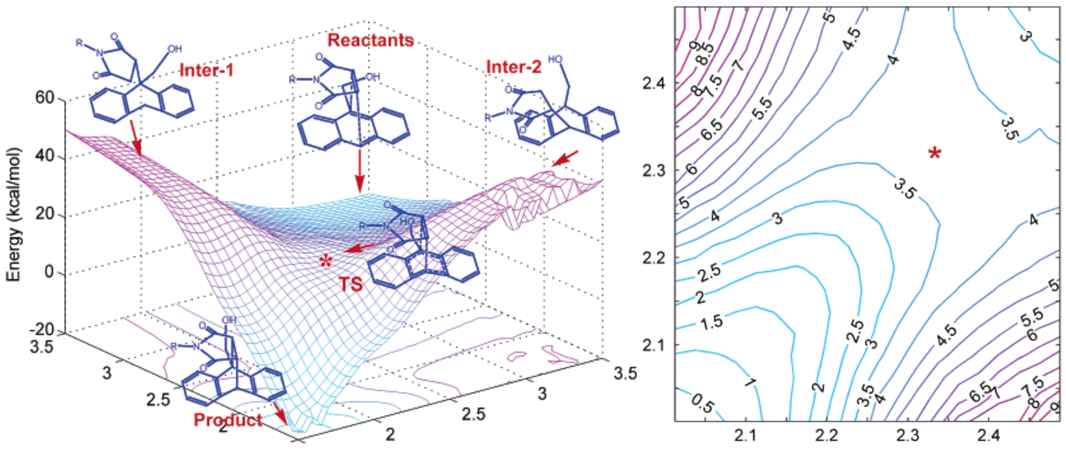


FIG. 2.6. The free-energy landscape of the Diels Alder ribozyme. *Left,* X and Y-axis are the positions in Å of two selected base pairs of the molecule. The ribozyme has to cross the saddle point for a catalytic reaction of the reactant towards the product. *Right,* the two dimensional projection with the energy color coded. Picture taken from [21]

to assign states through their dynamic behavior (fluctuation, transitions to other states or repeating sequential patterns of state changes) rather than assigning states by static properties as for example mean distance or positions. The understanding of states in a kinetic sense is the central viewpoint of this thesis and will be described further in Chapter 4.

CHAPTER 3

Theoretical Framework

Since this work is based on numerical analysis of probabilistic processes, the theoretical background will be described in the following. The aim is to deduce a description of the investigated molecular system from the gathered time-series, which has to be treated by the Inverse Problem Theory presented hereafter. A suitable tool for the description of processes governed by stochastic behaviour are *stochastic differential equations* which will be described in conjunction with the Markov Model afterwards. A more detailed description of the resonance energy transfer will be given through consideration of dipole-dipole interaction, and the numerical methods employed in the analysis will be described later on.

3.1. Inverse Problems and Bayes Ansatz

Many problems in physics or scientific descriptions of measurable phenomena in general consist of a collection of parameters called the model $\mathcal{M}$. For example, many statistical observations can be described by Gaussian distributions fully determined through the paramter tupel *mean* and *variance,* μ and σ respectively. The choice of the parameter values determines a specific realisation of the model suitable for constraints and boundary conditions of the current system under investigation. The model does not yet give a full description of the system or its observations since the parameters can be very abstract and do not need to be accessible for measurements. $\mathcal{M}$ rather describes the overall characteristics and properties of the system, but to reproduce the outcome of a time-evolving process, one needs the help of a function or operator introduced in the next section.

3.1.1. The Inverse Problem. We will assume that there is a a function or operator $\boldsymbol{G}$ (e.g a set of differential equations or polynomials) that sufficiently incorporates the fundamental physical relations of the process under study as functions of space $\mathrm{x} = \{x_1, x_2, ..., x_n\}$ (not necessary Euclidean) and time t. $\boldsymbol{G}(\mathcal{M}(\mathrm{x}, \mathrm{t}))$ should then be able to predict a wide variety of outcomes if all necessary initial conditions and boundary parameters (which can be real, complex or vector values) are known. Thus, $\boldsymbol{G}$ relates the paramters given through the model $\mathcal{M}$ and the experimental output (or measurement) $\mathcal{D}$:

$$\boldsymbol{G}(\mathcal{M}) = \mathcal{D}$$

When $\mathcal{M}$ and $\mathcal{D}$ are vectors, one refers to $\boldsymbol{G}$ as a function, whereas $\boldsymbol{G}$ is called operator in the case of $\mathcal{M}$ and $\mathcal{D}$ being described through functions. Values of

$\mathcal{D}$ can range from a single, real value within the interval $[0, 1]$, up to Terabytes of coordinates from time-series in a high-dimensional, complex space. If the calculated data from a certain model show only low accordance to experimentally obtained data, this can either be due to unconsidered processes which need to be incorporated in the model, or it can be induced by inappropriately chosen parameters. A good model should be able reproduce the observation of the investigated system and predict the outcome of new experiments on the system.

If there are only a few parameters inherent to the model and a small set of equations, one could achieve good correspondence to the experiment by trying out different values, until a trend of the outcome data $\mathcal{D}$ is obvious where an optimum can be conjectured. Unfortunately, this involves personal preferences and routines that may be hard or impossible to be formalized in a rigorous way. However, the huge number of wide-ranging parameters in molecular interactions demand for mathematical and algorithmic treatments, whereby the model parameters become the variables of the optimization function.

A particular property found in any real-life experiment at highest resolution is noise. If one does not incorporate the noisiness[1] of the output, it will produce a measurable, but hopefully small discrepancy η between the calculated outcome and the experimental measurements and the true model $\mathcal{M}_{true}$ that would be observed in the absence of noise:

$$\mathcal{D}= \boldsymbol{G}(\mathcal{M}_{\text{true}}) + \eta$$

The *mathematical* description of the process of finding the optimal parameter set for a given model is solving the *inverse problem*. Generating $\mathcal{D}$ from the model is termed the*foreward problem* and might involve solving ordinary or partial differential equations. For basic problems, it sometimes is possible to discriminate and compute the set of parameters that fit the observation data for a given model and it may even be possible to show that there is one and no second set of parameters that equally well or better fit the data. However, in most cases, the opposite is the case. Generally, it is not possible to decide whether an optimal fitting set can be found or not (*Existence*) and, if found, it is generally not possible to show that there is no second equal or better fitting set (*Uniqueness*). Let us assume a local optimum has proven good correspondence with a starting data set $\mathcal{D}_0$ and a slighty differing outcome $\mathcal{D}_1$ with very small differences ε to the starting set, then there arises the third crucial issue in Inverse Theory by observing the changes in the model resulting from deconvoluting $\mathcal{D}_0 \pm \varepsilon$ *(Stability, Instability):* if these changes for the models $\mathcal{M}_0$and $\mathcal{M}_1$ are dramatically high, the inverse problem is suffering from unstable behaviour and is called *ill-posed* in the case of continuous systems, or *ill-conditioned* for discrete linear systems.

[1] Noise is advised for the foundation of chaos theory, which is precisely the unpredictable huge change of outcome with even infinitesimally small intrinsic changes after finite time.

3.1.2. The Bayesian Approach. A fundamental difference towards the treatment of measurements that involve uncertainties is that, other than in the foreward problem known from classical mechanics with exact solutions, here, the model is described by *random* variables and yields a probability *distribution* f as solution for the model parameters. Thus, a solution will never exactly describe the outcome of one realization of the experiment, but, moreover, will assign a real number from the intervall $[0, 1]$, the *probability*, to every possible outcome with the boundary condition for the overall probability

$$\int_{-\infty}^{\infty} \mathrm{P(x)dx} = 1\,. \tag{6}$$

An agreement of experiment and theory can be obtained only by repeating the stochastic process often enough and seek similartities between the distribution of experimental results and a simulated outcome of the assumed model. Well-known probability distributions as the Gaussian or Chi-squared distribution have analytical expressions containing the parameters and can, therefore, be compared easily. However, a variety of problems exists that doesn't obey Gaussian or even symmetric laws. In this work, we focus on Gaussian (*normal*) distributed processes or processes that are goverend by *stochastic differential equations.*

The conditional probability is the probability of event A, given the occurrence of event B. It is written $P(\mathrm{A}|\mathrm{B})$ and reads "the probability of A, given B". Joint probabilities are the probability of two or more events in conjunction, that is, it is the probability of both (all) incident events. The *joint probability* of A *and* B is written $P(\mathrm{A} \cap \mathrm{B})$ and is used for the definition of the conditional probability.

DEFINITION 3.1. ***Conditional Probability***

The conditional probability P(A|B) of an event A assuming that B has occurred equals

$$P(A|B)) = \frac{P(A \cap B}{P(B)} \tag{7}$$

For statistically independent events A and B, that is $P(A \cap B) = P(A) \cdot P(B)$ the conditional probability is

$$P(A|B) = \frac{P(A)P(B)}{P(B)} = P(A). \tag{8}$$

The inverse problem in stochastic systems is characterized by finding the parameters (for distinction from the general case, in the following, $\mathcal{M}$ will be called λ and referred to as paramters) introduced by the known probability distribution functions $\mathrm{P(x)}$ identified now as the *prior* general model functions $\boldsymbol{G_{prior}}(\lambda)$, which reproduces the observed data $\mathcal{D}$ from the model. The inverse problem in probabilistic problems can, hence, be seen as modelling the inverse conditional probability (see definition 3.1) of the data $\mathcal{D}$ beeing produced by the assumed model ($\mathrm{P}(\lambda)$), also known as Bayesian inference.

Thomas Bayes theorem, formulated in the 18th-century, states the following conditional probabilities:

DEFINITION 3.2. ***Bayes theorem*** [a]

For the parameter set $\{\lambda_n\}$ with $n \in \mathbb{N}$ and the given observation $\mathcal{D}$, we have:

$$P(\lambda_n|\mathcal{D}) = \frac{P(\lambda_n)P(\mathcal{D}|\lambda_n)}{P(\mathcal{D})}. \tag{9}$$

with $P(\mathcal{D}) = \sum\limits_{n} P(\lambda_n)P(\mathcal{D}|\lambda_n) > 0$ for the discrete case or

$P(\mathcal{D}) = \int\limits_{allModels} P(\mathcal{D}|\lambda_n)P(\lambda)d\lambda > 0$ for the continuous case.

[a] In his words "The probability of any event is the ratio between the value at which an expectation depending on the happening of the event ought to be computed, and the value of the thing expected upon its happening".

It therefore introduces a second fundamental difference to classical deterministic problems by allowing us to incorporate prior information about the solution that may come from other data or prior experiences, the so-called *prior distribution* for λ (sometimes ”a priori distribution” or short ”prior”) $\mathrm{P}(\lambda)$. If there is no information available, the *principle of indifference* allows us to assign a uniform prior distribution assuming that every possible state is equally probable as initial realization. In the mathematical language, this distribution is called independent and identically distributed (**iid**). The collected data combined with the prior distribution generates a so-called *posterior distribution*

$$\mathrm{P}(\lambda|\mathcal{D}) \propto \mathrm{P}(\lambda)\mathrm{P}(\mathcal{D}|\lambda). \tag{10}$$

Bayes’ theorem relates the prior distribution $\mathrm{P}(\mathcal{D}|\lambda)$ which is computable before performing the experiment and the posterior distribution $\mathrm{P}(\lambda|\mathcal{D})$ via the given model $\mathrm{P}(\lambda)$ allowing the computation of the posterior distribution. $\mathrm{P}(\lambda|\mathcal{D})$ in turn can be computed from the outcome of the experiment and will hopefully be helpfull in understanding the system but should carefully not be mistaken as *the* answer to the problem but should rather be seen as *one possible* and more or less likely answer. Intuitively, one would suspect the model giving the highest posteriori value as the ”best” model (maximum a posteriori (MAP) model) as it states that it is highly probable to observe the data $\mathcal{D}$ from the system described by the model $\mathcal{M} = P(\lambda)$. Then again under consideration of the mentioned *uniqueness* difficulties it isn’t safe to say the there is any physical correspondance to model. Alternatively, the mean and variance of the posteriori distribution can provide helpful information in the process of finding the optimal model.

3.2. Parameter Estimation

3.2.1. The Likelihood Function. Now we consider a *second* statistical $P(B)$ process with a known probability distribution that describes the occurrence of an observable event B given the outcome of the *first* process $P(A)$. Thus we have two sets of events $\{\omega\}$ and are interested only in the overlapping region of $\omega_i \in A \cap B$. The joint probability distribution can then be calculated if $P(A)$ and $P(B)$ are statistically independent (see definition 3.1.) by multiplying both probabilities

$$P(A \cap B) = P(A) \cdot P(B) \tag{11}$$

This is, for example, often the case in medical treatment calculations given the infection probability of a certain desease (first process, A denotes the two possibilies infected or healthy) and the detection probability of the diagnosis (second process B is outcome of the diagnosis: positive or negative). The definition of the conditional probability then, for example, can compute the chance of the disease being diagnosed (and treated) while being perfectly healthy, the so-called false positives. We can easily generalize the joint probability for an arbitrary number of succeding processes $A_1, A_2, A_3, ..., A_n$

$$\mathrm{P_{total}(A_1 \subset A_2 \subset A_3 \subset ... \subset A_n) = \prod_{i=1}^{m} P(A_i)} \tag{12}$$

If the events can be partitioned into disjoint sets governing all events as in the case of being or not being ill and equipped with the prior distribution $\mathrm{P}(\mathcal{D}|\lambda)$ and the corresponding formula to compute a conditional probability, one can calculate the probability of an arbitrarily drawn sample to belong to one of these sets or states. The method which updates the prior probability with every newly collected data (drawn samples) to achieve better correspondence is called *Bayesian inference*.

In order to employ Bayesian inference we have just discussed, their translation into functional and programmable tools has to follow. This is carried out by the maximum likelihood analysis. $\mathcal{L}(\lambda \mid \mathcal{D})$ is the **likelihood function**, a conditional, possibly multi-variant (for the case of $\mathcal{D}$ consisting of vector valued observations $X = (x_1, x_2, ..., x_n)$) probability density function with the first argument held fixed, thus, it is a function of its second argument, the parameters. It is identical to the posterior distribution in Eq. 10 in the case of **iid** samples:

$$\mathcal{L}(\lambda \mid \mathcal{D}) = \prod_{i=1}^{N} \mathrm{P_i}(X_\mathrm{i}|\lambda)\ . \tag{13}$$

3.2.2. Baysian Networks. Now we will consider several probabilities P_n, which are connected in a way that the outcome of one realisation will be conditioning the

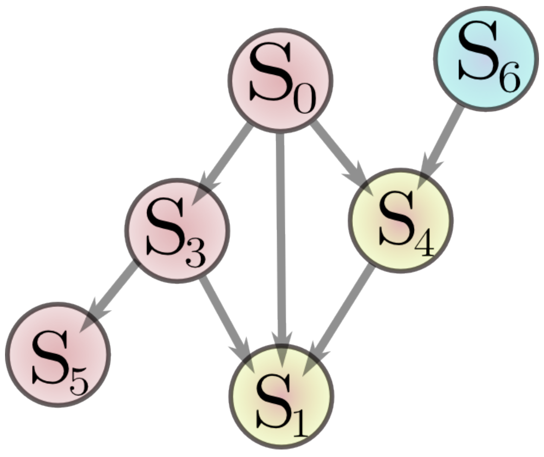


FIG. 3.1. Bayesian network displayed as directed acyclic graph. Every state from $\mathcal{S} = \{s_1, ..., s_n\}$ is associated with the probability P_n describing the output of s_n. Probabilities of red filled circles are dependend on state s_0 and independent from s_6. Mixed colored circled are dependend on both.

probability of the next realisation. This is called a Bayesian network. A visual representation can be given through a directed acyclic graph as schown in Fig. 3.1. It allows the events to have multiple pathways that they can choose to travel along but does not allow a circular sequence at any time. This restriction is particularly usefull as it allows the description of a joint probability distribution over all variables as a product of conditional probabilities

$$\mathbf{P}(\mathcal{S}|\lambda) = \prod_{i=1}^{n} P(s_i|\mathbf{pa}(s_i), \lambda) \tag{14}$$

where $\mathbf{pa}(s_i)$ denotes the parents of s_i (in Fig. 3.1 $\mathbf{pa}(s_5) = \{s_3, s_5\}$)

Two variables s_i, s_j are called *conditionally independent* if they are independent of a third variable's state s_k so that they can be expressed as a product of conditional probabilities from that third variable

$$\mathbf{P}(s_i, s_j|s_k) = \mathrm{P}(s_i|s_k)\mathrm{P}(s_j|s_k) \tag{15}$$

as it is for example is the case for s_3 and s_6. The conditional independence allows further simplification for the calculation of the overall probability $\mathbf{P}(\mathcal{S}|\lambda)$ in Eq. 14.

3.2.3. Maximum Likelihood Estimator. Since every probability function is bounded to both sides, there always exists a maximum and minimum value (which are equal for the constant case). Therefore, we can select the parameter set λ^* that produces the most likely outcome, hence the highest value of the likelihood function $\mathcal{L}$. A maximization of $\mathcal{L}$ with respect to the parameters (now regarded as latent variables) leads to an optimal parameter set λ^* for the given data. This is known as the *maximum likelihood principle.*

DEFINITION 3.3. ***Maximum Likelihood Principle***

$$\lambda^* = \arg\max_{\lambda} \mathcal{L}(\lambda|\mathcal{D}). \tag{16}$$

As already mentioned, it is generally not possible to find the global maximum (the maximization then only produces a suboptimal set of parameters) nor is there a guaranty that an analytical expression of the maximizing function can be found. If there exists an equation that maximizes the likelihood, it is called the maximum likelihood estimate, and it can be used to evaluate a family of probability distributions by optimizing λ^*. This, however, can lead to the multiplication of hundredthousands of values smaller than one and therefore to numeric difficulties caused through the finite range number representation in every finite binary system. Since the extreme values will remain unaffected of monotone transformation, the logarithm of expression 13 yields a numerically comfortable expression involving only sums, called the log-likelihood $\tilde{\mathcal{L}}(\lambda)$.

$$\tilde{\mathcal{L}}(\lambda) = \ln\left(\prod_{i=1}^{N} \mathrm{P}(\mathcal{D}_{\mathrm{i}}|\lambda)\right) = \sum_{\mathbf{i=1}}^{\mathbf{n}} \log\left(\mathcal{L}(\lambda)\right) \tag{17}$$

The maximum log-likelihood $\tilde{\mathcal{L}}(\lambda)$ as presented above only yields a point estimate without confidential intervals. If the variables are also modeled as a probability distributions the maximization leads to the *marginal likelihood* or *integrated likelihood*. This way, one can treat naturally uncertainties as well as additional variables which are not observed in the distribution function and avoid overfitting the data. The joint probability vector $\mathbf{P}$ =$(P_1, P_2, ..., P_n)$ of the data, and unobserved data $X = (x_1, x_2, ..., x_n)$, $\mathbf{P}(X, \lambda, \mathcal{D})$, describing the process can then be expressed with the prior distribution $\mathbf{P}(\mathcal{D}|\lambda)$ estimated by the maximum likelihood principle towards

$$\mathbf{P}(\mathcal{D},\lambda,X) = \mathbf{P}(X|\lambda)\mathbf{P}(\mathcal{D}|\lambda)\mathbf{P}(\lambda)\,. \tag{18}$$

This is known as ”learning from the data” in recognition aplications. The functions P_i know treat the observation probabilities as well as unobserved processes inherent to the model. Applying the sum rule of integrals to Eq. 3. we can rewrite this equation to the posterior probability

$$\mathbf{P}(\mathrm{X}, \mathcal{D}) = \int \mathrm{P}(\mathcal{D}, \lambda, X)d\lambda \tag{19}$$

which is able to produce a predictive distribution for X if substituting Eq. 18 for the integrand and applying the product rule for probabilities on the left-hand side $\mathrm{P}(\mathrm{x}, \mathcal{D}) = \mathrm{P}(\mathrm{x}|\mathcal{D})\mathrm{P}(\mathcal{D})$, then divided by $\mathrm{P}(\mathcal{D})$, we get

$$\mathbf{P}(X \mid \mathcal{D}) = \int \mathrm{P}(X|\lambda)\mathbf{P}(\lambda|\mathcal{D})\mathrm{P}(\lambda)d\lambda. \tag{20}$$

It was shown that it is possible to reformulate the original (deterministic) inverse problem into terms of the Bayesian statistical inference theory being capable of treating uncertainties and, additionally, accounts for poor or incomplete measurements. A tool to evaluate certain models was presented, the maximum likelihood estimator that will be applied to the models that are introduced in the following. A wel-known maximum likelihood estimator is a Gaussian fit, which computes the tupel $\lambda = (\mu, \sigma)$ for an one-dimensional distribution of x. It is an appropiate model for statistically normal distributed processes. The model would then be a stochastic normal process with the Gaussian distribution over the spatial coordinate x:

$$P_{Gauss}(x|\sigma, \mu) = e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \tag{21}$$

and the maximum likelihood function $\mathcal{L}(\sigma, \mu|x)$ given through Eq. 13. would yield with maximization from definition 3.3. the optimal variance and mean estimation λ^* from the data:

$$(\sigma, \mu)^* = \arg\max \mathcal{L}_{\text{Gauss}}(\sigma, \mu|x). \tag{22}$$

The maximum likelihood principle has been successfully applied throuthough many different problems in diverse areas of biophysical research. Time-series analysis as described in [22], is applicable to the method as well as structure prediction or gene expression and regulation.

3.3. Hidden Markov Models

3.3.1. Markov property. If observed on a sufficiently large time-scale, many processes appear "memory-less", meaning that at any time in the sequence, there is no influence provable of any past event affecting the upcoming event. This property was first described in a mathematically elaborate form by Andrei Andreyevich Markov at the end of the 19th century. The definition of a Markov process reads

DEFINITION 3.4. ***Markov process and Markov property***

Let $X(t)$ with $t > 0$ be a non-negative stochastic process defined on the state space $S = s_1, ..., s_m, S \subset \mathbb{N}^{\mathrm{m}}$. This process is a Markov process if it satisfies for all $n > 0, n \in \mathbb{N}$ time points $0 \geq t_0 < t_1 < t_2 < \cdots < t_n < t_{n+1}$, the following Markov property*:*

$$\begin{aligned} P(X(t_{n+1}) &= s_{n+1} \,|\, X(t_n) = s_n,\ X(t_{n-1}) = s_{n-1},\ \cdots,\ X(t_0) = s_0 \\ &= P(X(t_{n+1}) = x_{n+1} \,|\, X(t_n) = x_n, \end{aligned}$$

There are appliances in weather modeling, speech and pattern recognition, financial market development and more, from which suitable examples of Markovian systems can be constructed. A general demand for Markovianity resides in the exponential growth of the factors in the joint probability distribution function when adding past states to the memory kernel of the time evolution model. Of course, difficulties may arise in displaying and judging the outcome probabilities when estimating a system with too many parameters and dependencies. In fact, a Markov process is the simplest stochastic process after the Poisson process, which is even independent from its current state (e.g. the throw of a dice). These systems are described through the positive definite probabilities P_i collected in the probability vector $\overrightarrow{\mathbf{P}}$ obeying the normalization $\sum_{i=1}^{n} P_i = 1$ equivalent to the statement that the probability of observing any of the possible events is one, hence, guaranteed.

3.3.2. Markov Chains and Metastability. A sequence of observations starting from the initial state s_i with observation X_0 and the distribution vector $\vec{\pi} = \tilde{\mathbf{P}}(X_0 = s_i)$ fulfilling the Markov condition above at every single event is called Markov chain. Even in relatively small molecules, the number of atom bonds involved, combined with their degrees of freedom, results in an extremely high-dimensional state space $\mathbb{R}^{\mathrm{full}}$ of all possible configurations. Nevertheless, research in conformational analysis of bio-molecules seems to indicate that only a fairly small number n of dominant states persists for a longer time creating a subset $\mathbb{R}^{\mathrm{n}} \subset \mathbb{R}^{\mathrm{full}}$. These conformations are often identified in a biophysical context with metastable sets described by a continuous or discrete probability distribution P. The state sequence is the sequential trajectorie of those few states. Observations on protein systems have confirmed that the overall dynamics of even complex interacting protein species, can be understood and described by assigning each metastable set a specific function or sensitivity to signals from other proteins. The sequential change of those states in an equilibrium environment is then appropriately described by Markov chains. A Markov chain

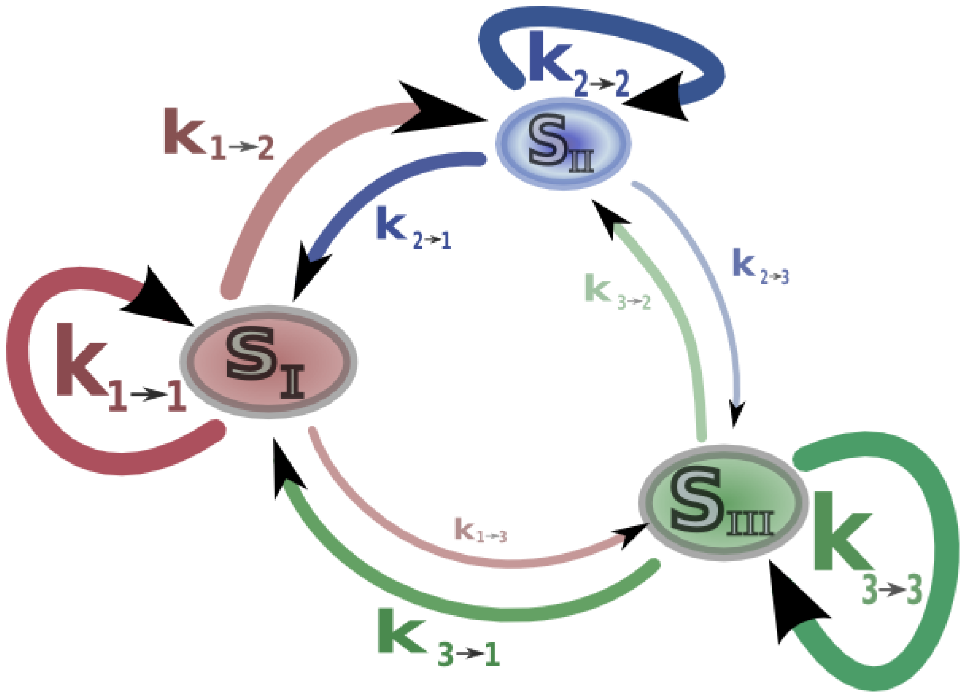


FIG. 3.2. A Three-state Markov model (states s_I,s_{II},s_{III}) with interchange rates $k_{i\to j}$ between the states $(i, j = 1, 2, 3, i \neq j)$. The circular curves with rates $k_{i\to i}$ denote the rates of reantrance (staying in the same state)

is called *time-homogeneous* if $\mathbf{P}(\mathrm{X}(\mathrm{k}+\mathrm{t}) = \mathrm{s_i} \mid \mathrm{X}(\mathrm{k}) = \mathrm{s_j})$ is independent of the choice of s with $\mathrm{s}, \mathrm{t} > 0$. The probabilites correspond to the transition rates $k_{i\to j}$ from state i to state j can be visualized as in the diagram shown in Fig. 3.2. Given the sequence of a time homogeneous Markov chain, one can construct the rate matrix $\mathbf{T} \subset \mathbb{R}^{\mathrm{n}\times\mathrm{n}}$ (23) simply by writing the transitions observed from every state $\mathrm{s_i}$ to every other state $\mathrm{s_i}$ in the corresponding entry

$$\mathbf{T}^{\mathrm{ij}} = \begin{pmatrix} k_{0,0} & k_{0,1} & \cdots & k_{0,j} & \cdots \\ k_{1,0} & k_{1,1} & \cdots & k_{1,j} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \cdots \\ k_{i,0} & k_{i,1} & \cdots & k_{i,j} & \cdots \\ \vdots & \vdots & \cdots & \vdots & \ddots \end{pmatrix}. \tag{23}$$

A transition matrix for stochastic processes always satisfies the condition $\sum_j k_{ij} = 1$ (normalization of the sum of rows, see previous section 3.3.1). A metastable system is defined for our propose through a very high probability of the system to maintain itscurrent state and therefore can be suitable described through the transition matrix T^{ij} in Eq. 23 if the *diagonal* elements (since they represent the probability of maintaining the current state) are very close to 1, therefore all off-diagonal elements have to be close to zero. This can also be seen in the eigenvalue spectrum of a metastabel transition matrix through eigenvalues close to 1.

3.3.3. The hidden layer: discrete hidden state, discrete output. Until now we only considered the case that, at every time in the sequence, the state the system occupies is known (e.g. measurable). This would mean for our present case of molecular conformations that, for every atom, the molecule complex, its present kinetic properties (e.g. rotational or vibrational state), and maybe even the electron orbitals would have to be accessible for measurements. This is by far unrealistic. In fact, we are provided with a vanishing small fraction of the just mentioned properties. What actually can be observed under realistic circumstances is, for example, the relative distance of two distinct regions of the molecule encoded in the FRET signal reaching us from a donor and an acceptor placed in that regions. To account for these massive restrictions, a second stochastic layer is introduced bearing a probability model for the *observables* emitted from the *unobservable,* hence hidden *state* $\mathcal{S}$. The number of these states has to be provided, although there are attempts of automatic determination of feasible states [23], one usually can find consistent assumptions that restrict the number of states.

It is assumed that the n states (or conformations) $\mathrm{s}_1$to s_n the system is able to occupy are known from preliminary analysis. Furthermore, let there be probability distributions $\mathrm{P}(\mathrm{s}_\mathrm{i})$ known to us that describe all possible observations d_i that can occure if the system is in the current state s_i. The system is called *ergodic* if there exists a point in time t_m at which *all* states have been emitted. This is a necessity for the following analysis, which would, otherwise, assign a zero probability to the states not being observed, thus it is impossible to produce any useful parameter estimates for that state. These observations can be continuous or discrete, and may even be as few as the three letters V,X, and Z as for example in Fig. 3.3, defining the elements of the data set $\mathcal{D}= \{\mathrm{d}_1, \mathrm{d}_2, \mathrm{d}_3\}$emitted from the two state system $\mathcal{S}$=$\{\mathrm{s}_1, \mathrm{s}_2\}$.

If considering only systems that are observable at time intervals that are much shorter than the average time of the system to change a state, the system will emit a characteristic string with statistical occurrences of the four letters. The string is determined by the probabilities $\mathrm{P}(X_i|\mathrm{s}_\mathrm{j})$ connected to the rate constants $\mathrm{k}^{\mathrm{s}_\mathrm{i}}_{\mathrm{X}_\mathrm{i}}$ (as depicted in in Fig. 3.3) and the state interchanging probability $\mathrm{P}(\mathrm{s}_\mathrm{i}|\mathrm{s}_\mathrm{j})$ with the correspinding interchanging rates $k_{i\to j}$ for the transitions between every state. This assumption isn't necessary (see [24]), but it helps the understanding of the hidden Markov process in the current case. Inferring the state of the system at a chosen, single point in the string (e.g. by just knowing *one* letter) is per se impossible, as long as there is a probability greater than one for every other letter to be emitted from any state. To solve this dilemma, we have to be patient and listen to the system for a while. After a time-span $\Delta\mathrm{t}$, we have collected a sufficiently long trajectory (lets say half a million letters) for a meaningful statistics. We can now try to identify the *state sequence* $\{\mathrm{s}_{\mathrm{t}_1}, \mathrm{s}_{\mathrm{t}_2}, \ldots, \mathrm{s}_{\mathrm{t}_\mathrm{n}}\}$, from the *letter sequence.* Thus, Hidden Markov Model offers a way to combine the model of the measurable data with the unobservable

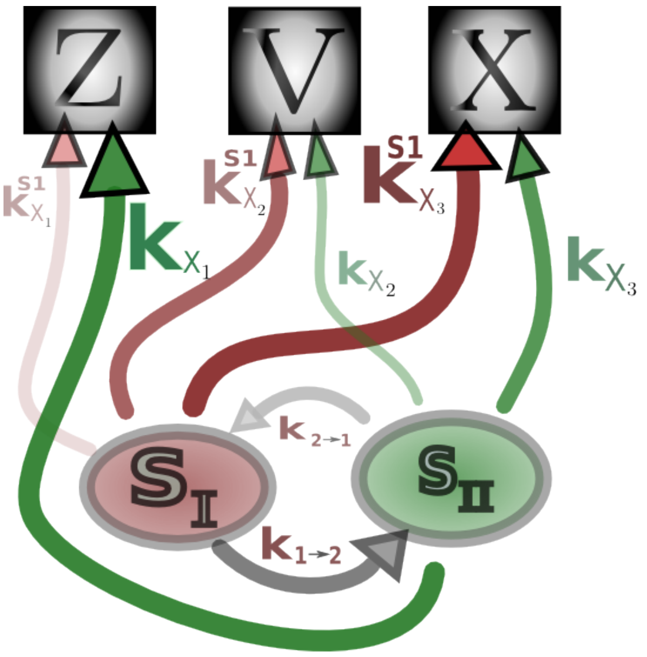


FIG. 3.3. Simple hidden Markov model with two states (s_I, s_{II}) and the three observables $X_i = \{X, V, Z\}$. $k_{X_i}^{s_i}$ are the rates of observation emissions, $k_{i\to j}$ are the state interchange rates .

processes underlying and generating them. By supplementing the incomplete data set $\mathcal{D}_O = \{X_1, \ldots, X_{n_n}\}$ with the output of the hidden model $\mathcal{D}_H = \{\mathcal{X}_1, \ldots, \mathcal{X}_{n_n}\}$ we can build a "complete" data set $\mathcal{D}_{Comp} = \left\{ \begin{array}{c} \mathcal{D}_H \\ \mathcal{D}_O \end{array} \right\}$. The joint density function (or distribution in the discrete case) $\mathbf{P}(\mathcal{D}_{Complete} \mid \lambda)$ can be written as the product of probabilities see Eq. 12 and 20

$$\mathbf{P}(\mathcal{D}_{Complete} \mid \lambda) = \mathbf{P}(\mathcal{D}_H \mid \mathcal{D}_O, \lambda) \cdot \mathbf{P}(\mathcal{D}_O \mid \lambda) \tag{24}$$

A ful characterization of the system bearing hidden states and dynamics describalble through a transition matrix T is given in the tupel $\mathcal{M} = (\mathcal{S}, \mathcal{D}, \mathbf{T}, \mathbf{P})$ of a Hidden Markov Model with the vector of probability distribution $\mathbf{P}$ or, in the continuous case, the probability density function ϱ .

There are three tasks associated with constructing a suitable Hidden Markov Model:

- What is the likelihood function $\mathcal{L}(\lambda \mid \mathcal{D})$ of our data when given only an initial model.
- For a given model characterized by the estimated parameter λ, what is the resulting hidden trajectory.
- Find an optimization method for the parameters λ to compute λ* from Eq. 16.

3.3.4. Hidden Markov Models with Gaussian Output.

A natural and often reasonable assumption for the distribution within a state is a Gaussian describtion

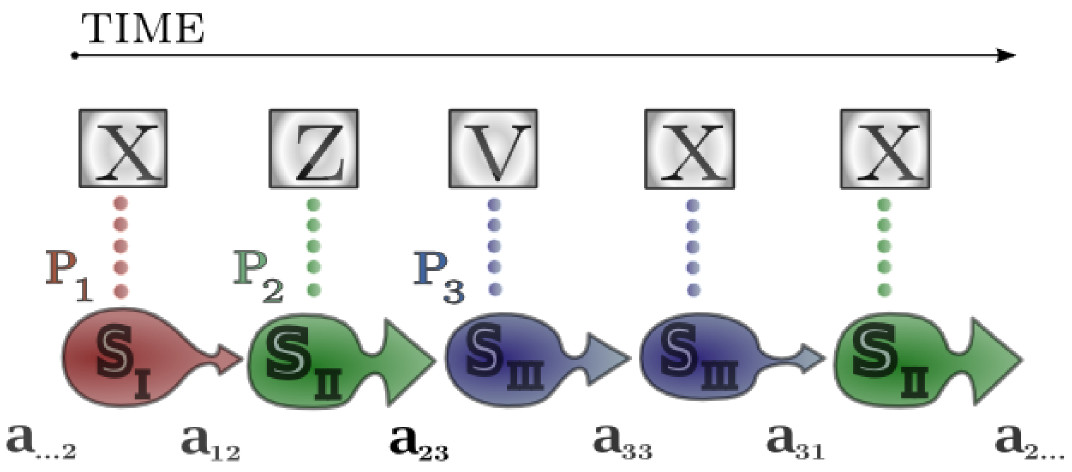


FIG. 3.4. The three states $S_I - S_{III}$ are emitting observation output X,Z,V, described by the corresponding probabilites $P_1 - P_3$. The chain of state-changes a_{ij} produces a characteristic output sequence.

given by Eq. 22. This represents the situation, in which the mean is the most probable observation outcome with a 98% chance to find observations within the double standard deviation. The Markov switching process immediately assigns the according mean and variance to the distributions $\mathrm{P}_1, \mathrm{P}_2, \mathrm{P}_3, ...\mathrm{P}_\mathrm{n}$ at every transition point as shown in Fig. 3.4. This still is somewhat unrealistic, since there is no ”time to travel” from one state to the other within the system. Therefore a dynamical enhancement is given next.

3.4. Stochastic Differential Equations

3.4.1. Langevin Dynamics. We are interested in an equation of motion for the n-dimensional stochastic system. This was offered by Langevin through introducing an effective potential V\ranlge that is inherent to the stochastic system described by random variables. Furthermore, the damping constant γ together with the noise intensity σ need to be identifyed in order to satisfy the following definition.

DEFINITION 3.5. ***Langevin equation***

$$\dot{q} = p, \; \dot{p} = -\nabla V(q) - \gamma p + \sigma\xi(t) \tag{25}$$

or (26)

$$dq = p, \; dp = (-\nabla V(q) - \gamma p)dt + \sigma dW \tag{27}$$

$\xi(t) = \dot{W}(t, w)$ denotes an n-dimensional Wiener process characterized by a zero mean: $\langle W \rangle = 0$ and the correlation function $corr\,(W(t)W(t)) = \delta(t - t')$ or $corr\,(\xi(t)\xi(t')) = \delta(t - t')$ respectively.

The equation considers a particle at the coordinate q being objected to a potential V and undergoing Brownian motion. The random movement in one or three space coordinates can be seen as produced by interactions on a very fast time-scale (unable to be described in a deterministic fashion), whereas the effective *potential function* $V(q,t)$ models the slow dynamics that are characteristic on longer time-scales. As it is often the case in molecular dynamics, which takes place in environments of extremely low Reynold numbers (of the order of 10^{-4}representing the friction) no inertia is considered, meaning that any movement will stop instantaneous if all forces are omitted (which of cause isn't the case at any point due to the time-continuous noise term). Hence, we can set $\dot{\mathrm{p}} = 0$ and, thus, regard the simplified Langevin equation for the high friction limit, also known as *Smoluchowski equation:*

$$\nabla V(q) + \gamma p = \sigma \dot{W} \tag{28}$$

The stochastic process induced by this equation is the Ornstein-Uhlenbeck process. The task is now to formulate a potential function that reproduces the behavior of the system. This could be achieved by modelling a apropriate potential landscape (as e.g. a multi well potential as in), where the metastability arises through the system's rare change within the minima (wells), if constructed appropriately.

A different approach allowing us to separate the *state* dynamics from the *state-change* dynamics is given through combining Eq. 28 with a Markov jump process between the n hidden states. For every state we can assign a characteristic potential and noise intensity. In the present case we will choose V as a harmonic potentials of the form $\mathrm{V_i(q)} = \alpha_\mathrm{i} \cdot (\mathrm{q} - \mu)^2$, thus characterizing the whole system through the parameter-tupels $(\gamma_\mathrm{i}, \sigma_\mathrm{i}, \alpha_\mathrm{i})$ and the transition matrix $\mathbf{T}$ responsible for the changes of $\mathcal{S}_\mathrm{i}$. For every state, there is an associated Ornstein-Uhlenbeck process resulting from Eq. 28

$$\gamma \dot{q} = 2\alpha_i \left(\mu_i - \mathrm{q}\right) + \sigma \dot{W} \tag{29}$$

describing the motion of the system.

3.4.2. Hidden Markov Models with Stochastic Output. We adopt the stochastic formulation of the observables to extend the Hidden Markov model with a stochastic differential equations (SDE) formulation to produce a *continuous* output over the state space. The state changes are again described by Markov jumps with an SDE for each state $\mathcal{S} = \{\mathrm{s_1}, ..., \mathrm{s_m}\}$. This yields an equation of motion for the continuous observation $\mathrm{q(t)}$ in the following form:

$$\dot{q}(t) = -\nabla_q V_{s_i(t)}(q(t)) + \sigma_{s_i(t)} \dot{W} \ \ (i = 1...d) \tag{30}$$

where $s_i(t)$ is the trajectory of the state vector in the reduced state space $\mathbb{R}^\mathrm{d}$ describing the time-series of s_i (as shown before in Fig. 3.4.), called the *Viterbi path*, generated

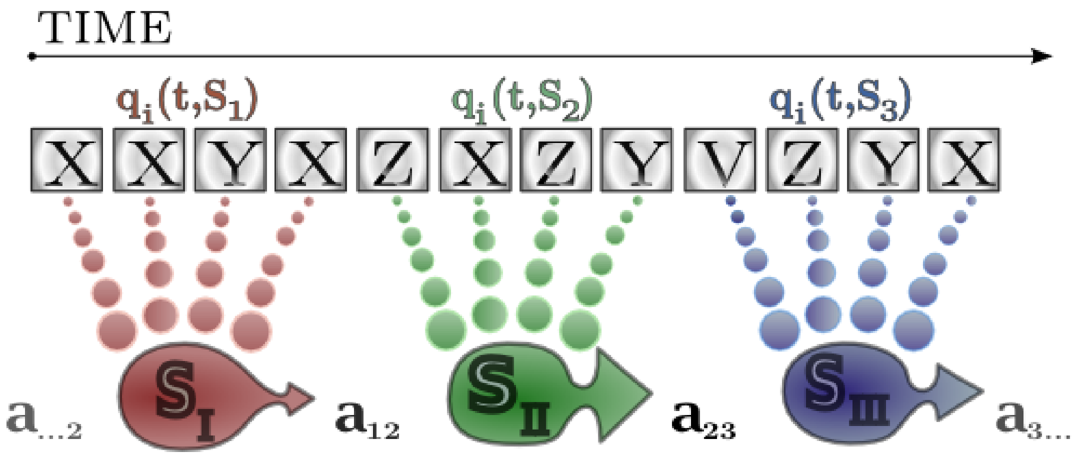


FIG. 3.5. The sequence of letter $q_i(t) \in \{X, Y, Z\}$ is generated from stochastic differential equations corresponding to reach state $s_I - s_{III}$. a_{ij} is a state Markov chain as in Fig. 3.4

through a Markov jump process. It follows from the assumption of harmonic potential functions $\mathrm{V_i(q)}$ that the non-linear SDE can locally be linearized towards

$$\dot{q}(t) = F^{\mathrm{s}(t)}(q - \mu^{s(t)}) - \sigma^{\mathrm{s}(t)}\dot{W}(t) \tag{31}$$

This is already formulated with upper indices to indicate its applicability for a multivariant description [2], where $F^{\mathrm{i}(t)}$is a set of $(\mathrm{d} \times \mathrm{d})$ force matrices.

An extensive description of the method used here is described in Schütte et. al. *Sequential Change Point Detection in Molecular Trajectories* 25. It utilizes a hidden Markov model jump process, where the dynamics of the molecule within the states are described by stochastic differential equations as the Smoluchowski Eq. 28 above.

3.4.3. Multivariant Estimation. For parameter estimation of a d-dimensional linear SDE based upon observations at equidistant time-points, an approximate likelihood function has to be found. It is known that for fixed initial conditions, the solution of a linear SDE is a Markov process and a time discretization of the solution is a multivariant normal distribution. Hence, given an observation $\mathrm{q_t}$, the conditional probability density of $\mathrm{q_{t+1}}$is a multivariant Gaussian with the vector valued mean $\mu^i \in \mathbb{R}^{\mathrm{n}}$ and covariance tensor $\Sigma \in \mathbb{R}^{\mathrm{n \times n}}$

$$\mathrm{f}_\lambda(q_{t+1} \mid q_t) = \frac{1}{\sqrt{\mid 2\pi R(\tau) \mid}} \exp\left(-\frac{1}{2}(q_{t+1} - \mu_t)^{\mathrm{T}} \mathrm{R}(\tau)^{-1} (q_{t+1} - \mu_t)\right) \tag{32}$$

$$\mu_t := \mu + e^{\tau F}(q_t - \mu), \;\; R(\tau) = \int_0^\tau e^{sF} \Sigma \Sigma' e^{sF'} ds$$

[2] In fact, it can also incorporate a higher order memory kernel if its parameter estimation method does not rely on a Hidden Markov assumption.

As with the *Smoluchowski equation* (28) we find for λ the three-tupel consisting of the system characteristic parameters (μ, F, Σ) mean, force towards the mean and noise intensity respectively, allowing us, in principle, the construction of a likelihood function $\mathcal{L}(\lambda \mid \mathbf{q}) = \prod_{\mathrm{t}=1}^{\mathrm{T}-1} \mathrm{f}_\lambda(\mathrm{q}_{\mathrm{t}+1} \mid \mathrm{q}_\mathrm{t})$.

With an analytic solution, the maximization process would simply be solving its derivative set to zero, unfortunately, there is no known solution for $\mathcal{L}$ so we follow the notation and idea of Meerbach et al. 26 and rewrite Eq. (32) towards

$$q_{t+1} = \mathcal{N}(\mu + e^{\tau F}(q_t - \mu), R) = (\mathbf{I} - e^{\tau F}\mu) + e^{\tau F} q_t + \mathcal{N}(0, R) \tag{33}$$

with the *multivariant normal* distributed random variable $\mathcal{N}(q, R)$ and the identity matrix $\mathbf{I}$ of matching size. If we consider an observation of length n generating the observation vector $\mathbf{q} = q_1, q_2, ..., q_n$ Eq. (33) suggests an autoregessive procedure of the order one, which can be written in the compact version

$$Y = \phi X + \varepsilon \tag{34}$$

with the following definitions

$$\begin{aligned} X &:= \begin{pmatrix} 1\,, 1, ...\; 1 \\ q_1, q_2, ..., q_{T-1} \end{pmatrix} \\ Y &:= (q_2, q_3, ..., q_T) \\ \phi &:= \left((\mathbf{I} - e^{\tau F})\mu e^{\tau F}\right) \\ \varepsilon &:= (\mathcal{N}(0, R), ..., \mathcal{N}(0, R)) \end{aligned} \tag{35}$$

the transformed parameter set $\widetilde{\lambda} = (\phi, R)$ is now able to be estimated through the likelihood function

$$\mathcal{L}(\tilde{\lambda} \mid \mathbf{q}) = \left(\frac{1}{\sqrt{|\, 2\pi R(\tau)\,|}}\right)^{(T-1)} e^{\left(-\frac{1}{2} tr(Y - \phi X)(Y - \phi X)' R^{-1}\right)} \tag{36}$$

with the merit of the analytic estimators

$$\hat{\phi} = Y X'(X X')^{-1} \; \hat{R} = \frac{\left(Y - \hat{X\phi}\right)\left(Y - \hat{\phi X}\right)'}{T - 1}$$

which are exact, but can become critical at the matrix inversion of XX'under some conditions. A more robust version can be used for parameter computation, when the moment matrix

$$M\left(\mathbf{q}\right) := \sum_{\mathrm{i}=1}^{T} (1, q_\mathrm{i}, ..., q_{\mathrm{i}+\mathrm{p}}) \begin{pmatrix} 1 \\ q_\mathrm{i} \\ \vdots \\ q_{i+p} \end{pmatrix} = \begin{pmatrix} XX' & XY' \\ YX' & YY' \end{pmatrix} =: \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}$$

is employed which carries all statistically relevant information about the time series. The maximum likelihood estimator in terms of the moment matrix is then able to produce Bayes factors that can be used to identify a change of states through the parameter behavior[3].

Methods exist for dealing with those parts of the system that aren't behaving Markovian (or only do so on a larger time-scale), as discussed in [27] and are object to current research.

3.5. Information-theoretical approach

Another way to look at single photon trajectories is to analyze the amount of information they are carrying. A single-molecule experiment is characterized by the following parts:

- a *molecule* in focus
- a reporter *dye,* emitting the fluorescent light
- the local *environmental* conditions

At the point where environmental side effects are of the same or even higher magnitude as the desired quantity a natural question would be to ask for a "limit" at which no more statements can be made about the measurement, as it had been the case in the example stochastic model, when too few observations (letters) were at hand. However, even with an arbitrarily long sequence, it could well be that the noise has governed the signal in a way that it is not carrying any information about the molecule at all.

Mutual Information. The mutual information (Mi) is a scalar quantity for measuring the mutual dependency of two variables. It can be described as the relative entropy between the probability $\mathrm{P(x,y)}$ of the two variables $\mathrm{x,y}$ and the product of their unconditionalized probability $\mathrm{P(x)P(y)}$:

$$\mathbf{Mi}(x;y) = P(x,y) \cdot \log_2 \frac{\mathrm{P(x,y)}}{\mathrm{P(x)P(y)}} \tag{37}$$

The calculations in [28] section D determine a minimum number of photons necessary for model inference. No data processing can increase the amount of information inherent to the measured system, therefore the number of photons at a fixed time discretization yields a maximum amount of information that can be extracted from the system. Any noise (equal to no information) sources reduces the information in the signal. In the extreme case, the information carried by two photons isn't able to explain anything at the molecular level. For a typical visible range of 400 -700 nm and an experimental duration of 100 seconds, it would take 57 bits per photon to record all the information present in the uncertainty-principle limited stream of a two-fluorophore system. Following this calculation, depending

[3] Since the results of the algorithm on data from our experimental data turned out less significant, further elaboration is omitted.

on the value of the background noise and crosstalk-factors, a minimum of about 1000 photons has to be acquired for inferring one transition rate.

It it also possible, as shown in [29], to extract dynamic information from the photon stream by means of Fischer-Information Theory. However this aproach is still very unexplored in molecular analytics and shall not be further elaborated here.

3.6. Förster Resonance Energy Transfer Efficiency

3.6.1. Dipole-Dipole Resonance. As described in the introduction, Förster resonance energy transfer (FRET) is a non-radiative (therefore sometimes called ”vibrational”) transfer of exciting energy from the externally excited and fluorescent donor towards the acceptor molecule dye in itsground state. This, of course, is a lossy process that is accompanied by other (e.g. radiative) transfer processes, and is disturbed e.g. by electron transfer and triplet absorption that both cause a quenching of the fluorescence light from either donor, acceptor or both. A detailed treatment of these influences and error sources demands a deeper understanding of the physico-chemical process during molecular interactions with photons and among each other. The rate k_T at which energy is transferred from the donor molecule towards the acceptor is modeled at best if both fluorophores are considered as two weakly coupled dipoles separated by the distance r. The full equation for the distance-dependent transfer is given in 30 chapter 13 by

$$k_T(r) = \frac{Q_D \kappa^2}{\tau_D r^6} \left(\frac{9000(ln10)}{128\pi^5 \mathsf{N} n^4} \right) \int\limits_0^\infty F_D(\lambda)\, \varepsilon_a(\lambda)\, \lambda^4 d\lambda, \tag{38}$$

where Q_D is the quantum yield of the donor in absence of the acceptor; n denotes the refractive index of the medium (for bio-molecules in aqueous solution often assumed to be 1.4); N is Avogadro's number; τ_D is the lifetime of the donor in acceptor absence; $F_D(\lambda)$ is the normalized[4], dimensionless spectral radiant fluorescence intensity of the donor; ε_a represents the wavelength-dependent molar extinction coefficient[5] of the acceptor; κ^2 is the orientation factor. The integral expression matches the spectral overlap between the donor emission and acceptor absorption spectra, which is measured in ${}^{nm^6}\!/_{mol}$ and, sometimes, can be expressed analytically or at least be measured at high precision.

$$J(\lambda) = \int\limits_0^\infty F_D(\lambda)\, \varepsilon_a(\lambda)\, \lambda^4 d\lambda \tag{39}$$

As with the spectral overlap, the quantum yield and the refractive index can be calculated or measured and are insensitive for distance changes in the Förster range

[4] the area under the curve is normalized to unity

[5] the extinction value for the RNA molecule under study here at the exciting wavelength of 514.5 nm was theoretically determined to be 453600${}^{1}\!/_{mol\cdot cm}$

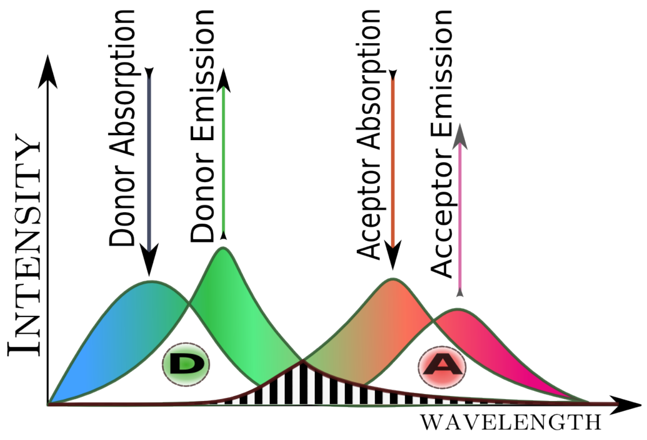


FIG. 3.6. Spectra of donor and acceptor fluorophores. Excitation takes place at the donor emission peak, fluorescence emission is characterized through the absorption spectrum. The overlap area is striped and determines the efficiency of non-radiative energy transfer.

of 10-100 as the physical constants are. Then Eq. (38) can be arranged to reveal the Förster distance

$$R_0^6 = \frac{9000(ln10)Q_D}{128\pi^5 \mathrm{N} n^4} \int_0^\infty F_D(\lambda)\, \varepsilon_a(\lambda)\, \lambda^4 d\lambda \tag{40}$$

at which the transfer rate k_T is equal to the decay rate of the donor in absence of the acceptor $k_\tau = \frac{1}{\tau_D}$ and, therefore, exactly half the energy is converted into FRET with the other half being dissipated into heat or other radiative or non-radiative processes. If the wavelength is expressed in nanometers, R_0 is $0.211(\kappa^2 n^{-4} Q_D \cdot J(\lambda))^{1/6}$. Equipped with the value of R_0, one has the convenient form of the Förster law

$$k_T = \frac{1}{\tau_D}\left(\frac{R_0}{r}\right)^6 \tag{41}$$

enabling us to determine, whether the FRET transfer is faster (in terms of $\frac{1}{k_T}$), in which case the major proportion is transferred via FRET, or slower than the decay rate of the donor, which would result in little transfer. The efficiency, accordingly, is then expressed as

$$E = \frac{k_T}{\tau_D^{-1} + k_T} = \frac{R_0^6}{R_0^6 + r^6} = 1 - \frac{\tau_{DA}}{\tau_D} = 1 - \frac{F_{DA}}{F_D} \tag{42}$$

where in the second fraction k_T was replaced with Eq. (41), the expression after the third equality sign uses the lifetimes of the donor with acceptor present τ_{DA} and without τ_D, the fourth expression uses their fluorescence F_D and F_{DA}, respectively. Regarding the graph of the Förster law, one quickly realizes that values less than 0.2 R_0 or above 2 R_0 result in huge uncertainties when dealing with noise (since changes at efficiencies of 0.99 or 0.01 are untraceable at experimental conditions). The optimal range of measurements should be around efficiencies of 0.5 equivalently at R_0.

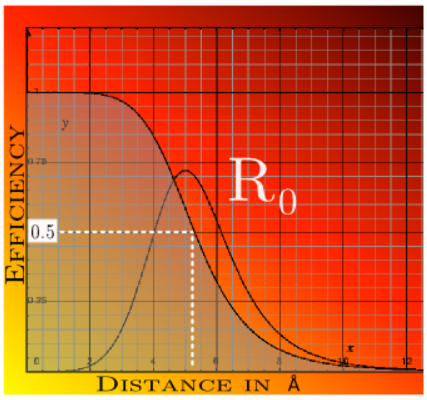


FIG. 3.7. The curve with shaded integral area represents Förster law (see Eq. 76), second curve (with peak at R_0) describes the the efficiency distribution with a half width of approximately $\frac{1}{3}R_0$.

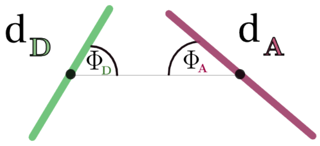


FIG. 3.8. Orientation of donor (violet) and acceptor (green) dipole moment $d_{D/A}$ with angles $\phi_{D/A}$ measured from the line connecting the centers (black dots)

A critical issue is the orientation factor κ, which depends on the mutual orientation of both dye dipole moments. its value is determined by the electromagnetic dipole-dipole coupling law.

3.6.2. Dipole Orientation. When dealing with dipole characteristics, one can't reduce the interactions to point sources of charge q, since the transition dipole moment (in the following termed moment) $\tilde{p} = q \cdot \tilde{d}$ is the intrinsic source of the electromagnetic dipole field which will react with any external field $\vec{E}$ with the torque $\vec{T} = \vec{p} \times \vec{E}$.

The coupling strength of two dipoles, arranged as shown in Fig. 3.8 is given through the multi-pole expansion containing the spherical harmonics, which are a good approximation if the dipole-dipole interactions are dominant (hence neglecting higher orders of the expansion). It is given through

$$\nu_{dipole} = \frac{1}{4\pi\varepsilon_0}\frac{\kappa^2\phi_D\phi_D}{r^3} \tag{43}$$

with the distance between the center of the dipoles r, the free space dielectric function ε_0 and the orientation factor κ. The latter is a measure that describes the alignment of the dipoles, which can yield a maximum value of 2 for the co-linear and parallel cases and 1 for only parallel dipoles. A change of κ^2 in Eq. (38) over this range results in a 26% change in r, whereas an anti-parallel alignment with $\kappa^2 = 0$ seriously affects

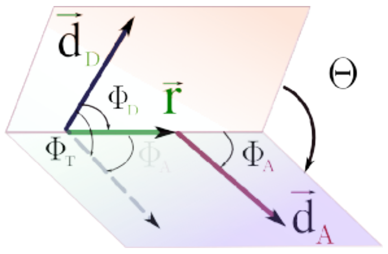


FIG. 3.9. Orientation of the donor and acceptor dipole in three dimension. Two planes intersecting at an angle Φ_T are constructed as shown. The dipole moment d_D projected from the $d_D \times r$ plane (orange) onto the plane wherein d_A lays is used for calculating the oriental factor κ (Eq. 44) .

the distance calculation. The most common assumption of isotropic distribution is characterized by a orientation factor of $\frac{2}{3}$, deviations of κ^2 can alter the distance r within a maximal range of 35%. If constraints are known so that one can determine the orientation factor through the orientations as in Fig. 3.9, it is computable through

$$\kappa^2 = (\cos\Phi_T - 3\cos\Phi_D\cos\Phi_D)^2 = (\sin\Phi_D\sin\Phi_A\cos\Theta - 2\cos\Phi_D\cos\Phi_D) \tag{44}$$

Given the dipole coupling ν_{dipole} for the system, the energy transfer given in Eq. (41) can be formulated as

$$k_T = \frac{2\pi}{\hbar}|\nu_{dipole}|^2 \int_0^\infty F_D(\lambda)\,\varepsilon_a(\lambda)\,\lambda^4 d\lambda \tag{45}$$

The decay rate of the excited donor is inverse proportional to its lifetime τ_D. In a FRET setting, the rate is further modified through the inverse proportionality of the sixth order to the distance r of the acceptor.

3.6.3. Light Intensity versus Discrete Photons. Whereas the Förster law is formulated in continuous time and continuous intensities, here, it is applied to a discrete time trajectory (which is due to the sample time-window and binary storage) and discrete intensity values (the photon counts). Most of the time one can transform the integral equations into summation over discrete time and intensities values as follows.

3.6.3.1. *Window method.* The most intuitive and still predominant method in many laboratories is to define the averaging window length τ and write the sum of all photons within $\left\{t_i^A \mid t - \frac{\tau}{2} < t_i^A < t + \frac{\tau}{2}\right\} \subset \mathrm{A}$ with A being the acceptor counts, as intensity $\mathrm{I}_{win}^{A}(\mathrm{t_i})$, and all photons within $\left\{t_i^D \mid t - \frac{\tau}{2} < t_i^D < t + \frac{\tau}{2}\right\} \subset D$ as intensity $\mathrm{I}_{\mathrm{win}}^{\mathrm{D}}(\mathrm{t_i})$ at the time-stamp $\mathrm{t_i}$ of the new trajectory vector $I_{win}(\mathrm{t}) = \left\{\mathrm{I}_{\mathrm{win}}^{\mathrm{A}}(\mathrm{t_i}),\mathrm{I}_{\mathrm{win}}^{\mathrm{D}}(\mathrm{t_i})\right\}$. The window-based method then yields the intensities

$$\mathrm{I}^{A}_{win}(t_i) = \sum_{i-\frac{\tau}{2}}^{i+\frac{\tau}{2}} \delta\left(t_i^A\right) \tag{46}$$

$$\mathrm{I}^{\mathrm{D}}_{\mathrm{win}}(t_i) = \sum_{i-\frac{\tau}{2}}^{i+\frac{\tau}{2}} \delta\left(t_i^D\right) \tag{47}$$

and efficiency

$$E_\tau(t) = \frac{|\mathrm{I}^{\mathrm{A}}_{\mathrm{win}}(\mathrm{t_i})|}{|\mathrm{I}^{\mathrm{A}}_{\mathrm{win}}(\mathrm{t_i})|+|\mathrm{I}^{\mathrm{D}}_{\mathrm{win}}(\mathrm{t_i})|} = \frac{|\mathrm{I}^{\mathrm{A}}_{\mathrm{win}}(\mathrm{t_i})|}{|I_{win}(\mathrm{t})|} \tag{48}$$

which then is applicable to the Förster law.

$$E_\tau(t) = \frac{1}{1+\left(\frac{r(t)}{R_0}\right)^6} \tag{49}$$

This method relies on a sufficient number of photons in each time window, which otherwise produces significant uncertainties due to shot noise. It therefore implies a trade off between the error bounds and the time resolution (window length), which is chosen rather arbitrarily for the individual case. Furthermore, this method saliently assumes a uniform *a priori* distribution of the FRET intensities rather than for the distance, which is incommensurate with the strong non-linearity of their connection. The distance distribution $p(r)$ can be calculated through the derivative of Eq. (49) yielding the non-uniform distribution depicted in Fig. (3.7):

$$p(r) = \frac{\left(\frac{r}{R_0}\right)^5}{\left[1+\left(\frac{r}{R_0}\right)^6\right]^2}$$

with a half width of about $\frac{1}{3}R_0$ implying a preferred residence at distances close to R_0.

Hidden Markov Model for intensity values. As already mentioned it is advisable to treat all measurements and their resultant quantities as driven by stochastic processes. If done so, one can use, in contrast to the time window method, the history at a given point in the time-series and its future (if recorded) for inference. If we want to consider the efficiency outcome of the FRET process $\mathrm{E(t)}$ as a probabilistic variable, we can rewrite Eq. (48) towards the probability

$$E(t) = \frac{p_A(t)}{p_A(t)+\gamma p_D(t)}$$

of a certain efficiency value when observing the photon pattern with $p_A(t)$ and $p_D(t)$ as probability distributions at the time t for the chance of detecting an acceptor photon or donor photon, respectively. γ incorporates the ratio of donor/acceptor quantum yield $\frac{Q_a}{Q_D}$ as well as the differing detection efficiencies $\eta_{A/D}$ in the measuring instrument (avalanche photo diodes in our case) for donor and acceptor photon detection.

$$\gamma = \frac{\eta_A}{\eta_D}\frac{Q_A}{Q_D}$$

As mentioned in the previous subsection, a uniform distribution of distances is unjustified, therefore leaving the possibility to only treat efficiencies, which in turn isn't as informative for understanding the folding process as the distances are. A probabilistic approach towards estimating the *distance* trajectory from photon records is desired. To that aim we will follow the article [22] by considering a statistical ensemble of distance trajectories $\{r(t)\}$ and compute each conditional probability $\mathrm{P}\left[r(t)|\{t_i^A, t_i^D\}\right]$ that r is realized for the captured photon arrival times $\{t_i^A, t_i^D\}$. The probability is given by Bayesian statistics (see (3.1.2))

$$\mathrm{P}\left[r(t)|\{t_i^A, t_i^D\}\right] \propto \mathrm{P}\left[\mathrm{r(t)}\right]\mathrm{P}\left[\left\{t_i^D, t_i^A\right\}|r(t)\right] \tag{50}$$

The time discretization $\frac{\triangle T}{N}$ is chosen such that during an observation interval $\triangle T$ at most one photon is found in any of the N time bins $[\tau_{i-1}, \tau_i]$with, $i = 1, ..., N$ (for a continuous limit $N \rightarrow \infty$ can be considered). For the discrete distance trajectory $r_1, ..., r_N$ the conditional probability to observe the photon pattern $E_1, ..., E_N$ is

$$\mathrm{P}\left[E_1, ..., E_N|r_1, ..., r_N\right] = \tau^{n_A+n_D}\prod_{i=1}^{N} f_i \tag{51}$$

where f_i denote the probability densities according to which of the three events "donor photon is recorded" (D), "acceptor photon is recorded" (A), or "no photon is recorded" (0) occurs during $[\tau_{j-1}, \tau_j]$, calculated through

$$f_i = \begin{cases} I_D(r_j)\left[1-\tau I_A(r_j)\right] & f\text{or } D \\ I_A(r_j)\left[1-\tau I_D(r_j)\right] & \text{for } A \\ \left[1-\tau I_D(r_j)\right]\left[1-\tau I_A(r_j)\right] & \text{for } 0 \end{cases} . \tag{52}$$

The recorded total photon number $I_0 = I_A(t) + I_D(t) = \frac{1}{\triangle T}(n_A + n_D)$ is used in Eq. 51 to transform the probability densities into probabilities. Regarding the over-damped opening and closing of the molecule on the millisecond time-scale, the Langevin equation of movement transforms into a Fokker-Planck equation, where the *a priori* probability $\mathrm{P}\left[\mathrm{r(t)}\right] \propto \lim_{\mathrm{N}\rightarrow\infty}\mathrm{P}\left(\mathrm{r_1, ..., r_N}\right)$, $r(t)$ can be assumed to result from a one-dimensional diffusion process with effective diffusion coefficient D. We can now write for the discretized distance transition probabilities the random walk (free diffusion) probabilities

$$g_{i+1|j} \propto \frac{1}{\sqrt{4\pi D\tau}}e^{-\frac{\left(r_{j+1}-r_j\right)^2}{4D\tau}} . \tag{53}$$

If knowledge about the energy landscape of the molecule is available, it can be incorporated in g in a Smoluchowski-type generalization, otherwise this approach assigns equal a priori probabilities to every distance r_i, which is the model-free approach. Eq. (51) now reads

$$\mathrm{P}\left[r_1, ..., \mathrm{r_N} | \{t_i^A, t_i^D\}\right] \propto f_1 \prod_{j=2}^{N} g_{j|j-1} f_j \tag{54}$$

From here, we can calculate the probability distribution of one chosen distance r_k at times $\frac{\tau_{k-1}+\tau_k}{2}$ through integration over all other distances

$$\mathrm{P}\left[\mathrm{r_k} | \{t_i^A, t_i^D\}\right] \propto \int \dots \int P\left[r_1, \dots, r_N | \{t_i^A, t_i^D\}\right] dr_1 \dots dr_{k-1} dr_{k+1} \dots dr_N \tag{55}$$

Using Eq. (54), we can rearrange integrals to obtain

$$P\left[r_k | \{t_i^A, t_i^D\}\right] \propto L_k f_k R_k \tag{56}$$

with

$$\begin{aligned} L_k &= \int g_{k|k-1} f_{k-1} dr_{k-1} \int g_{k-1|k-2} f_{k-2} dr_{k-2} \dots \int g_{2|1} f_1 dr_1 \\ R_k &= \int g_{k|k+1} f_{k+1} dr_{k+1} \int g_{k+1|k+2} f_{k+2} dr_{k+2} \dots \int g_{N|N-1} f_N dr_N \end{aligned}$$

which also can be expressed as the following iteration

$$\begin{aligned} L_k &= \int g_{k|k-1} f_{k-1} dr_{k-1} L_{k-1} \\ R_k &= \int g_{k|k+1} f_{k+1} dr_{k+1} R_{k+1} \end{aligned} \tag{57}$$

which transform into forward/backward Schrödinger-type equations for the continuum limit $\tau \to 0$ reassembling the generalized diffusion equations for $L_k \to L(r,t)$ and $R_k \to R(r,t)$. With $f_k - 1 = \tau F(r,t)$ (to ensure convergence), the recursion Eq. (57) now reads as differential formulation, using the relation $\partial_\tau g_{k|k-1} = D\partial^2_{r_{k-1}} g_{k|k-1} = D\partial^2_{r_k} g_{k|k-1}$ from the diffusion formula (53) :

$$\begin{aligned} \partial_t R(r,t) &= \lim_{\tau \to 0} \left\{ \partial_r^2 \left[(1 + \tau F_\tau(r,t)) R(r,t) \right] + \left[F_\tau(r,t) + \tau \partial_\tau F_\tau(r,t) L(r,t) \right] \right\} \\ \partial_t L(r,t) &= \lim_{\tau \to 0} \left\{ \partial_r^2 \left[(1 + \tau F_\tau(r,t)) L(r,t) \right] + \left[F_\tau(r,t) + \tau \partial_\tau F_\tau(r,t) L(r,t) \right] \right\} \end{aligned}$$

Solving and normalizing the above equations yields the desired probability distribution to find the distance r at time t

$$\mathrm{P}\left(\mathrm{r,t} | \left\{\mathrm{t_i^A, t_i^D}\right\}\right) \propto \mathrm{L(r,t)} \left[1 + \tau \mathrm{F}_\tau(\mathrm{r,t})\right] \mathrm{R(r,t)}$$

Equation (52) expressed as a Gaussian limit representation for the δ-function $\delta(t-t') = \lim_{\tau \to 0} h(t-t')$ with the Gauss-function $h_\tau(t-t') = (2\pi\tau)^{-1/2} \exp\left(-\frac{(t-t')^2}{2\tau^2}\right)$ and neglecting higher orders of τ, one finds the probability density

$$F_\tau(r,t) = [I_D(r) - 1] \sum_{j=1}^{n_D} h_\tau(t - t_j^D) + [I_A(r) - 1] \sum_{j=1}^{n_A} h_\tau(t - t_j^A) - I_0 \tag{58}$$

which can be plugged into Eq. (57). For convenience, only the solution for the third event ”no photon recorded” shall be displayed (for $\partial_t L(r,t)$ in the full version see [22]) with solutions propagating in time for $t > t'$ and $t < t'$ respectively given by

$$L(r,t) = e^{-I_0(t'-t)} \int L(r',t') e^{-\frac{(r-r')^2}{4D(t-t')}} dr' \tag{59}$$

$$R(r,t) = e^{-I_0(t'-t)} \int R(r',t') e^{-\frac{(r-r')^2}{4D(t-t')}} dr' \tag{60}$$

Nprobe method. Given the photon trajectory from donor/acceptor of length N (time-steps in ms) as a two-dimensional vector $p = \{\mathrm{D}, \mathrm{A}\}$ with $\mathrm{D} = \left\{t_1^D, ..., t_N^D\right\}$ and $\mathrm{A} = \left\{t_1^A, ..., t_N^A\right\}$ with at most one photon arriving for all $\mathrm{t_i}$, we can treat the photon observation as random variable $q(t)$ with

$$g(t) = \begin{cases} 0 & t \in D \\ 1 & t \in A \end{cases}, \tag{61}$$

the efficiency as expectation value of q is then of the form:

$$E_\tau(t) = \frac{1}{|P(t)|} \sum_{i=1}^{|P(t)|} q(P(t)_i) = \mathcal{E}(q(t))_{t-\frac{\tau}{2}...t+\frac{\tau}{2}} \tag{62}$$

with the expected mean square difference for time τ

$$\mathcal{E}\left[(E(t) - E(t+\tau))^2\right] \approx \mathcal{E}\left[(q(t) - q(t+\tau))^2\right] \tag{63}$$

We assume that the true FRET efficiency is constant, $E(t) = E = const$, and we observe the system with an intensity I_0 at the time step Δt over the period T, the collecting time window (not to be confused with the averaging time window of method 3.6.3.1). Hence, we are given the efficiencies through $p_A(t) = E$ and $p_D(t) = 1 - E$.

The probability of observing a given window pattern $I_{win}(\mathrm{t})$ is:

$$\mathcal{P}(I_{win}(\mathrm{t})|E(t) = E) = \prod_{i=1}^{|A(t)|} E \prod_{i=1}^{|D(t)|} (1 - E) = E^{|A(t)|}(1 - E)^{|D(t)|}. \tag{64}$$

The expectation value of the observed window efficiency for the window length $w = \frac{\tau}{\Delta t}$ is:

$$\mathcal{E}(E_\tau(t)|E(t) = E) = E \tag{65}$$

The standard deviation of this expectation follows as

$$\begin{aligned}\sigma_E &= \frac{\sigma_q}{\sqrt{I_{win}(\mathrm{t})}}\\ = \frac{\sqrt{< q^2 >- < q >^2}}{\sqrt{I_0 w}} &= \frac{\sqrt{< E^2 > - < E >^2}}{\sqrt{I_0 w}}\\ = \frac{\sqrt{E - E^2}}{\sqrt{I_0 w}} &= \frac{\sqrt{E - E^2}}{\sqrt{I_0}} w^{-\frac{1}{2}}\end{aligned}$$

We, thus, expect a power law behavior with an exponent of -0.5 of the standard deviation versus window length with a pre-factor depending on E.

Let $E(t)$ be distributed normally, $\mathcal{N}(E, \mu, \sigma)$ with mean μ and standard deviation σ. Let's also assume that $E(t)$ is sufficiently smooth such that it allows a detection of at least 2 successive photon probes with the approximately constant E. We can probe $E(t)$ n times successively (and optimally with maximum possible or feasible collection time window) assuming that during the n probes $E(t)$ is constant. Each probe then yields either an acceptor or donor photon. The probability for an acceptor photon is given by:

$$\mathcal{P}(a, t) = \begin{pmatrix} n \\ a \end{pmatrix} E(t)^a (1 - E(t))^{n-a} \tag{66}$$

We now compute the mean number of acceptor photons for each probe through

$$\mathcal{E}(a) = \frac{1}{n} \sum_{a=1}^{n} a \begin{pmatrix} n \\ a \end{pmatrix} E(t)^a = \frac{1}{n} \sum_{a=1}^{n} n \frac{(n-1)!}{k!(n-k)!} E(t)^a (1 - E(t))^{n-a}) = nE \tag{67}$$

and thus the mean efficiency:

$$\mathcal{E}\left(\frac{a}{n}\right) = \frac{1}{n} \mathcal{E}(a) = E \tag{68}$$

and the mean square difference to $n\mu$:

$$\begin{aligned}\mathcal{E}\left((a - n\mu)^2\right) &= \sum_{a=0}^{n} (a - n\mu)^2 \mathcal{P}(E(t))\\ &= \sum_{a=0}^{n} a^2 \mathcal{P}(E(t)) - 2n\mu \sum_{a=0}^{n} a\mathcal{P}(E(t)) + n^2\mu^2 \sum_{a=0}^{n} \mathcal{P}(E(t))\\ &= \mathcal{E}\left(a^2\right) - 2n\mu\mathcal{E}(a) + n^2\mu^2\\ &= Var(a) + [\mathcal{E}\,(a)]^2 - n\mu\mathcal{E}\,(a) + n^2\mu^2\\ &= nE(t)(1 - E(t)) + n^2\,[E(t)]^2 - 2n^2\mu E(t) + n^2\mu^2\\ &= (n^2 - n)(E(t))^2 + (n - 2n^2)E(t) + n^2\mu^2\\ &= n^2\left[(E(t))^2 - 2\mu E(t) + \mu^2\right] + nE(t) - (E(t))^2\end{aligned} \tag{69}$$

For the mean square difference of the efficiency we find:

$$\mathcal{E}\left(\left(\frac{a}{n}-\mu\right)^2\right)=\frac{1}{n^2}\mathcal{E}((a-n\mu)^2)=\frac{(n-1)\,(E(t))^2+(1-2n\mu)\,E(t)+n\mu^2}{n}. \tag{70}$$

Now we are able to compute the standard deviation of the measurements for Gaussian behavior $\mathcal{N}\left(E,\sigma,\mu\right)$:

$$\sigma_o=\int\mathcal{N}\left(E,\sigma,\mu\right)\sum a\frac{n!}{k!(n-k)!}E^a(1-E)^{n-a}dE \tag{71}$$

leading us to the expectation of a:

$$\begin{aligned}\mathcal{E}(a) &= \int\mathcal{N}\left(E,\sigma,\mu\right)nE\,dE && (72)\\ &= n\mu && (73)\end{aligned}$$

and its *variance*:

$$\begin{aligned}\mathcal{E}\left[(a-\mathcal{E}(a))^2\right] &= \int\mathcal{N}\left(E,\sigma,\mu\right)\sum\left(\frac{a}{n}-\mu^2\right)\frac{n!}{k!(n-k)!}E^a(1-E)^{n-a}dE\\ &= \int\mathcal{N}\left(E,\sigma,\mu\right)\frac{(n-1)\,(E(t))^2+(1-2n\mu)\,E(t)+n\mu^2}{n}dE\\ &= \frac{n-1}{n}\int\mathcal{N}\left(E,\sigma,\mu\right)[E(t)]^2\,dE+\frac{1-2n\mu}{n}\int\mathcal{N}\left(E,\sigma,\mu\right)E(t)dE\\ &\quad+\mu^2\int\mathcal{N}\left(E,\sigma,\mu\right)dE\\ &= \frac{n-1}{n}\mathcal{E}\left[E(t)^2\right]dE+\frac{1-2n\mu}{n}\mathcal{E}\left(E(t)\right)+\mu^2\\ &= \frac{n-1}{n}\left[Var(E(t))+\mu^2\right]+\frac{1-2n\mu}{n}\mu+\mu^2\\ &= \frac{n-1}{n}\left(\sigma^2+\mu^2\right)+\frac{\mu-2n\mu^2}{n}+\mu^2\\ &= \frac{\mu-\mu^2}{n}+\frac{n-1}{n}\sigma^2.\end{aligned}$$

For the particular case of 2 successive photons ,we find

$$\mathcal{E}\left[(a-\mathcal{E}(a))^2\right]=\frac{\mu-\mu^2}{n}+\frac{1}{2}\sigma^2=d^2 \tag{74}$$

hence, the estimation of the *standard deviation* of the original distribution is given by:

$$\sigma_{NP}=\sqrt{2d^2-\mu+\mu^2}. \tag{75}$$

With increasing number of Nprobes, the trajectories generated by the NProbe method become shorter, since longer intervals within the original time-series will be averaged for the estimated efficiency value.

CHAPTER 4

Data Analysis

4.1. Trajectory Estimation

4.1.1. From Photons to Intensities. For the application of the Förster-law to the inter-dye distance r

$$E(r(t)) = \frac{I_A(t)}{I_A(t) + I_D(t)} = \frac{R_0^6}{R_0^6 + r(t)^6} \tag{76}$$

data conversion from the experimentally obtained two-channel photon stream $\{(t_i^D, t_i^A)\}$ of donor and acceptor into one-dimensional efficiency trajectories $\{E(t_i)\}$ needs to be carried out. Whereas at ensemble measurements, where large amounts of photoluminescent molecules are excited and emit intense photon-streams I_A and I_D (e.g. measurements at the large chlorophyll-antenna complex in the photosynthetic subunit of algae [31]), which then directly yield intensity values (I_d, I_a) for each channel, the situation is substantially different for single molecule spectroscopy. In the experiment considered here, the emitting source is a single molecular dye, which quickly undergoes photo destruction if excited heavily and over a long period, therefore, only a few photons are emitted within the sampling rate of 1 ms from the signaling dye. Fortunately, the confocal microscopy techniques gathers most of the dye-photons so that for longest possible observation durations, it is possible to reduce the excitation power of the high frequency laser to a minimum, hence, minimize the photo-reactivity of the signaling dyes, causing saturation, triplet quenching or reversible and irreversible bleaching. The resulting signal of each dye contains an average of one photon event (not necessarily only one single photon, in rare cases 5 to 7 photons hit the detector at once, that is within the sampling rate) every 50-150 μs, which can, in terms of intensity interpretation, be thought as a step function with intensity value 1 within the time window of photon detection. The efficiency-computation using the definition 4 to compute $\frac{I_A}{I_A + I_D}$, therefore, would provoke a division by zero in most of the cases and give values of zero most of the times and infinity or one in the few other cases if done for every time step Δt.

Thus, a summation over a fixed time-window length traditionally has been used to generate a mean value centered at the half length $\frac{\tau_\Sigma}{2}$ (thus summing at each point t_j over the last $\frac{\tau_\Sigma}{2}$ and coming $\frac{\tau_\Sigma}{2}$ time-steps as formulated in chapter 3, section 3.6.3.1) with the cost of information-loss about effects on the short end of the timescale, that is within $[t_j - \frac{\tau_\Sigma}{2}, t_j + \frac{\tau_\Sigma}{2}]$. The averaging frame generates a mean at every time step, thereby producing an intensity trajectory with its smoothness depending on

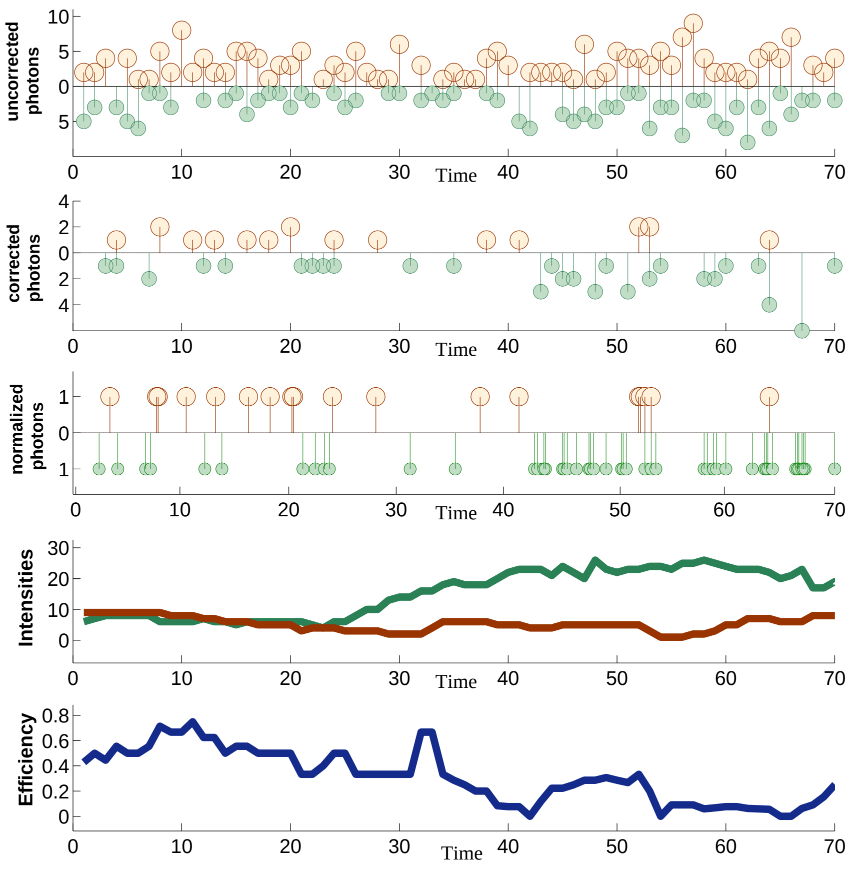


FIG. 4.1. Circles are photon counts, lines are intensities of acceptor (red) and donor (green), respectively. The bottom blue line is the efficiency of the intensities above.

the choice of the window length (10, 20, 50 and 100 time steps for comparison in the present case) depicted for an example trajectory in Fig. 4.2 clearly showing the loss of dynamic information. The intensities generated with the time-window method are applicable for efficiency computation via $E(t) = \frac{I_A(t)}{I_A(t)+I_D(t)}$, but further analysis will suffer from the mentioned loss of information on short timescales and is therefor not used for the variance computation. It is, however, essential for the visual impression and calibration, such as recognition of state changes and bleaching phases (see section 4.2.1).

4.1.2. Limitations of Histogram Interpretation. A traditional method for deriving quantities such as population distribution or occurrences of certain species is the histogram graph as shown in Fig. 4.3. It has been employed by biologists and bio-physicists, silently dropping all dynamic information: the histogram simply counts the number of occurrences of discretized y-values (efficiency or distance in our case) during the time-series along the x-axis into bins. Besides the inability of this method to analyze temporal dynamics within the time-series (caused through the reordering of the sequential information), it is furthermore dependent on the somewhat arbitrary choice of the histogram bin size which is the range of trajectory

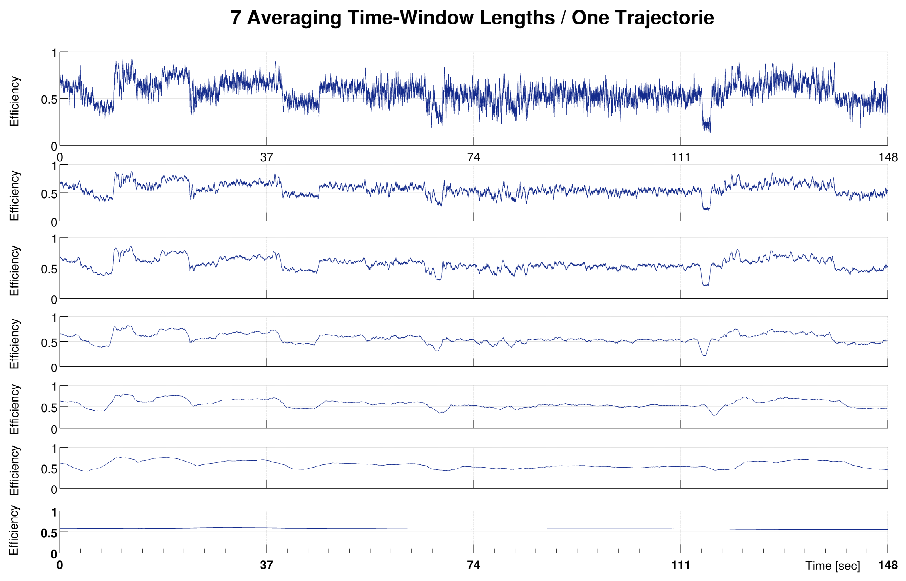


FIG. 4.2. The topmost efficiency trajectory is generated from a 20 ms averaging window τ. τ increases for the following trajectories until 10 sec for the lowest trajectory

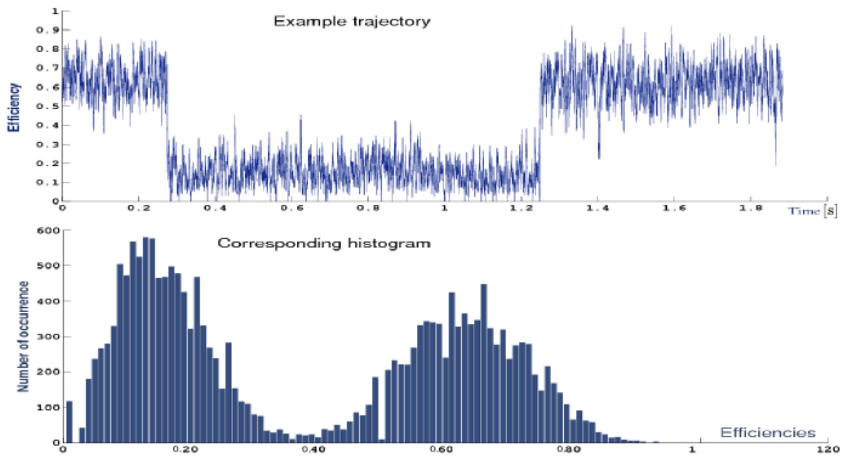


FIG. 4.3. Upper efficiency trajectory is generated from 20 ms binning. *Below* the corresponding histogram of the efficiency value occurrences.

z-values, equivalently histogram x-values $[x_j - \frac{bin}{2}, x_j + \frac{bin}{2}]$ counted as the same value x_j, or roughly speaking the width of the histogram bars.

As an example of the time-window length choice dependency Fig. 4.4 shows the various outcomes of histogram plots for five chosen τ_Σ on the entire collection of 173 efficiency trajectories. Note that the expected delta distribution would only arise for a single trajectory histogram, when τ_Σ reaches the total trajectory length T_j.

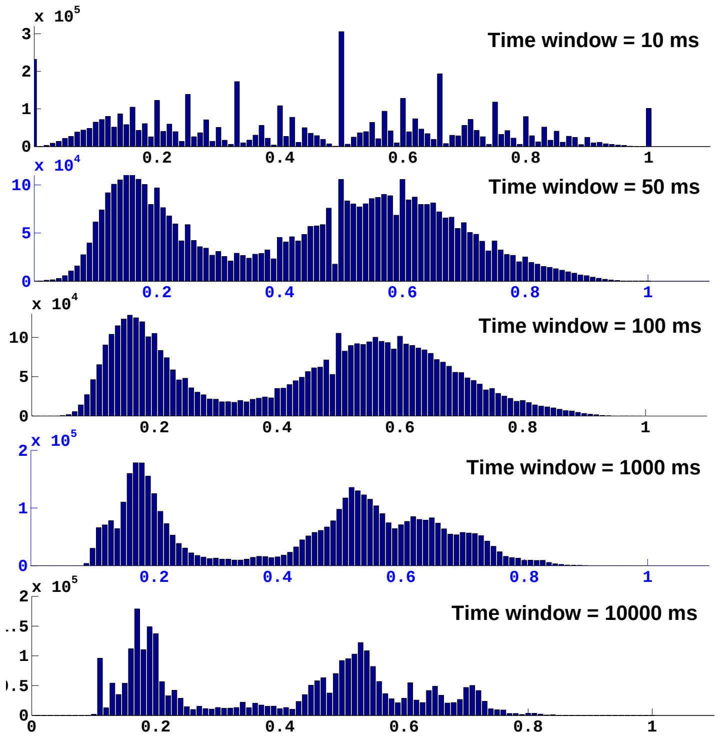


FIG. 4.4. Effects on the histogram during change of averaging time window. With increasing window length (downwards) the shape changes significantly.

4.2. Error Correction

Having developed an intensity trajectory $\{I_A(t), I_D(t)\}$, the foundation for efficiency analysis is laid out. A critical task in the adaption of the idealized model is to incorporate inevitable side effects and distortions: Nothing at room temperature is totally dark if measured at the single photon level. Background photons of matching color are unintentionally counted as either donor or acceptor signals, thus the true efficiency without background is given in the background correction supplemented version of Eq. 76:

$$I_A^{bg}(t) = I_A(t) - \alpha_A \tag{77}$$

$$I_D^{bg}(t) = I_D(t) - \alpha_D \tag{78}$$

which assumes that the *background noise intensities* $\alpha_{A/D}$ are time end efficiency independent.

Furthermore donor-emitted photons are sometimes of such a high wavelength that they are counted as acceptor emitted, the so-called **cross-talk**. The reverse direction is sometimes also considered, in the present case of largely separated emission spectra, there is almost no leaking of the acceptors emission into the donors absorption spectrum and the acceptor-donor crosstalk is set to zero. For the crosstalk corrected FRET efficiency Eq. 76 with cross-talk factor χ would therefore read:

$$I^{\chi} = I_A(t) - \chi I_D \tag{79}$$

$$I_D^{\chi} = I_D \tag{80}$$

Finally the simple expression for the efficiency $E(I_A(r), I_D(r))$ is assuming equal intensities for $I_A(r)$ and $I_D(r)$ if directly excited, implying an equally intense photon stream in both, donor and acceptor channels at the Förster radius R_0. This, for reasons explained hereafter, is generally not true in molecular photo-physics and is accounted for by introducing the **Gamma-factor** γ (sometimes also termed G-factor) defined either theoretically through the ratio of the quantum yield of donor and acceptor ϕ_A, ϕ_A respectively times the ratio of the detection efficiency for the donor channel at donor excitation $\eta_{A*}^{A}$and detection efficiency η_{D*}^{D} of the acceptor channel at acceptor emission:

$$\gamma = \frac{\phi_A \eta_{A*}^{A}}{\phi_d \eta_{D*}^{D}} \tag{81}$$

For the computation of the theoretical value one must know the following properties of the molecular and measuring system defined before or in section 3.6 of chapter 2:

- spectral Overlap J
- detections Efficiency η
- rotational Anisotropy κ
- reflective index n

The fully corrected version of formula 76 now reads:

$$E^{cor}(r) = \frac{I_A(t) - \chi\,(I_D(t) - \alpha_D) - \alpha_A}{I_A(t) - \alpha_A + \gamma\,(I_D(t) - \alpha_D)}\,. \tag{82}$$

The correction factors for background and crosstalk in our case can be determined quantitatively fairly easy from the Data as shown in the next section. However there are various approaches in estimating or calculating the Gamma factor (for reference see [32, 33]), ranging from intuitive reasoning a suitable range and distribution from empirical knowledge over analytical determination using the theory of quantum electrodynamics to sophisticated measurements (for example ALEXA-FRET measurements [34]) of direct excitation emission of both, donor and acceptor. The latter method was employed for cy5-cy3 fluorophore pair and found to have a mean value of $\gamma = 1.0$ with a rather broad distribution of ± 0.4.

4.2.1. Determination of the correction factors. The raw data are given in a two column vector with the length of the trajectory $i = 1, ..., T$ and $\delta\left(t_i^A, t_i^D\right)$ being the detector counts (number of photons) at each point in the time-series. Corrections are obtained through analyzing the mean intensities $\langle \mathrm{I_D(t)} \rangle$ and $\langle \mathrm{I_A(t)} \rangle$ generated from the time-window length method 3.6.3.1 during the two bleaching sequences following the productive phase of length $\mathrm{T_p}$ after time-stamp $\mathrm{t_p}$:

First Phase (*acceptor* is bleached for duration $\mathrm{T_{a*}}$ until $\mathrm{t_{a*}}$)

- $\widetilde{\mathrm{I}}_\mathrm{A} := \sum_{i=t_p}^{t_{a*}} \delta\left(t_i^A\right)$,
- $\widetilde{\mathrm{I}}_\mathrm{D} := \sum_{i=t_p}^{t_{a*}} \delta\left(t_i^D\right)$

Second Phase (*donor* is bleached for $\mathrm{T_{bg}}$ until the end of the trajectory $\mathrm{t_n}$)

- $\mathrm{I_A^{bg}} := \sum_{i=t_{a*}}^{t_n} \delta\left(t_i^A\right)$,
- $\mathrm{I_D^{bg}} := \sum_{i=t_{a*}}^{t_n} \delta\left(t_i^D\right)$,

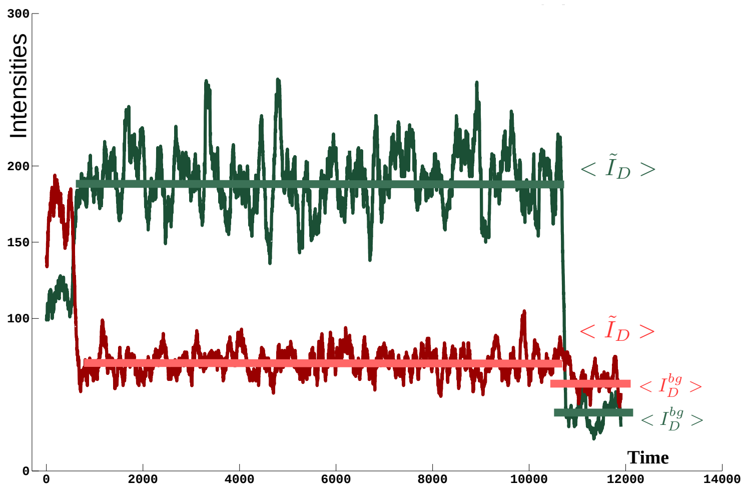


FIG. 4.5. Bleaching phases for determination of the error correction factors, mean intensities $\langle \widetilde{I}_A \rangle, \langle \widetilde{I}_D \rangle$ for the computing χ , $\langle I_A^{bg} \rangle, \langle I_D^{bg} \rangle$ for computation of α through Eq. 83 and 90

4.2.2. Background correction factor. From $\mathrm{I_A^{bg}}$ and $\mathrm{I_D^{bg}}$ we can compute the background correction factor by normalizing (thus computing the mean $\langle . \rangle$):

$$\alpha_A = \frac{\mathrm{I_A^{bg}}}{\mathrm{T_{bg}}} := \langle I_A^{bg} \rangle, \alpha_D = \frac{\mathrm{I_D^{bg}}}{T_{bg}} := \langle I_D^{bg} \rangle \tag{83}$$

and defining the background corrected intensities (generated from the time-window method) as

$$\hat{I}_D(t) : = I_D(t) - \alpha_D \tag{84}$$

$$\hat{I}_A(t) : = I_A(t) - \alpha_A \tag{85}$$

The collection of all individual background correction factors of the 56 trajectories selected for the single state analysis (those showing correct bleaching behavior) are spread around the mean values

$$\mu_A^{bg} = \frac{\sum_{i=1}^{56} \alpha_A^i}{56} = 0.14 \tag{86}$$

$$\mu_D^{bg} = \frac{\sum_{i=1}^{56} \alpha_D^i}{56} = 0.23 \tag{87}$$

with the standard deviations of

$$\sigma_A^{bg} = \frac{1}{55} \sum_{i=1}^{56} \left(\alpha_A^i - \mu_A^{bg} \right)^2 = 0.02 \tag{88}$$

$$\sigma_D^{bg} = \frac{1}{55} \sum_{i=1}^{56} \left(\alpha_D^i - \mu_D^{bg} \right)^2 = 0.04 \tag{89}$$

4.2.3. Crosstalk correction factor. We proceed with the background corrected values $\hat{I}_D(t)$ and $\hat{I}_A(t)$ and calculate via

$$\chi = \frac{\widetilde{I}_A(t)}{T_{a*}} - \alpha_A = \langle \widetilde{I}_A(t) \rangle - \langle I_A^{bg} \rangle \tag{90}$$

the background and one-way crosstalk corrected values

$$\begin{aligned} I_A^{cor}(t) &= \hat{I}_A(t) - \chi \\ I_D^{cor}(t) &= \hat{I}_D \end{aligned}$$

which yield a mean value and standard deviation of

$$\mu^{cross} = 0.11 \tag{91}$$

$$\sigma^{cross} = 0.13 \tag{92}$$

4.2.4. Gamma factor. The gamma factor, as mentioned before, can be determined theoretically or measured directly, however, for the latter measurements of acceptor excitation at the wavelength of the acceptors absorption-spectrum peak of 633nm would be necessary after each measurement which aren't available here. For a theoretical determination the known properties of the dye (as depicted in Fig. 4.7) knowledge of properties of the ribozyme-fluorophore complex would be required which isn't available either. Therefor a coarse estimation can be done by comparing the sum of donor and acceptor intensities before and after acceptor bleaching as shown in Fig. 4.5. The ratio of the two sum then gives the fraction of donor fluorescence that would cause the same quantum yield of donor and acceptor, hence

$$\gamma = \frac{\widetilde{\mathrm{I}}_\mathrm{A} + \widetilde{\mathrm{I}}_\mathrm{D}}{\mathrm{I}_\mathrm{A}^\Sigma + \mathrm{I}_\mathrm{D}^\Sigma} \frac{T_a}{T_p} \tag{93}$$

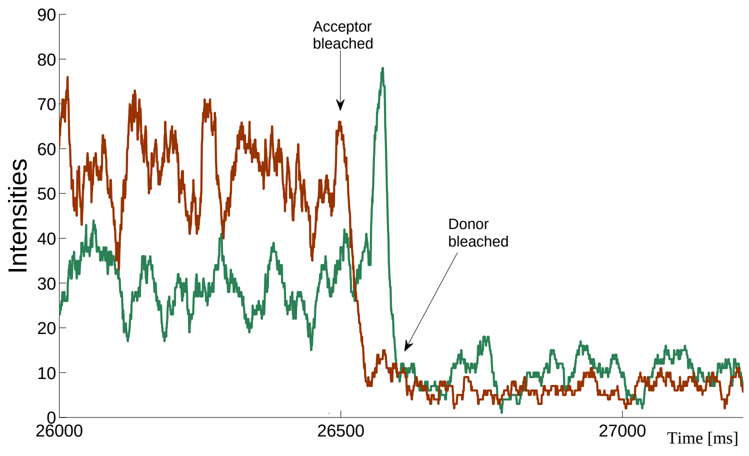


FIG. 4.6. Sample trajectory with short, therefore unsignificant, first bleaching phase as an example for high uncertainties in the correction factors due to a very short first bleaching phase (between the two arrows).

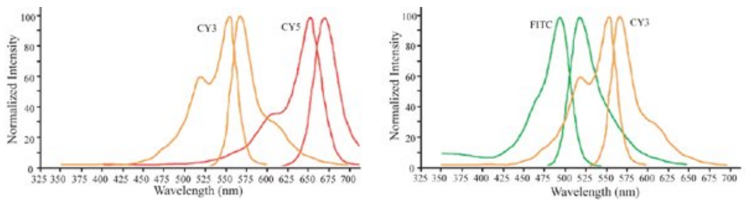


FIG. 4.7. *Left:* Emission spectrum of cy5-donor (red) and cy3-acceptor (yellow) *right:* their absorption spectrum in green and yellow, respectively.

where $\mathrm{I}_{\mathrm{A}}^{\Sigma}$ and $\mathrm{I}_{\mathrm{D}}^{\Sigma}$ are the sum of all acceptor and donor photons during the active phase, $\frac{T_a}{T_p}$ is the normalizing factor. The distribution of all calculated gamma factors is centered at the mean

$$\mu^{\gamma} = 0.84 \tag{94}$$

$$\sigma^{\gamma} = 0.25 \tag{95}$$

Unfortunately the quality of the extracted correction factors strongly depends on the duration of the two bleaching phases. If either of the two is too short the value will have a wide uncertainty range. For the first phase is determining the cross-talk, it over or under-estimates χ if as short as shown in the Fig. 4.5 below, or even worse, it can be missing at all. Under these restrictions about half the trajectories couldn't be included in further analysis which sadly reduced the overall quantity of individual trajectory, and hence, the significance of statistical methods. However, in recent publications as few as 20-50 trajectories are used for statistical inference such as the error correction but also for kinetic analysis yielding transition rates from cross-correlation analysis [35]. The calculation on 36 full trajectories were carried out with custom written Matlab scripts (see Appendix).

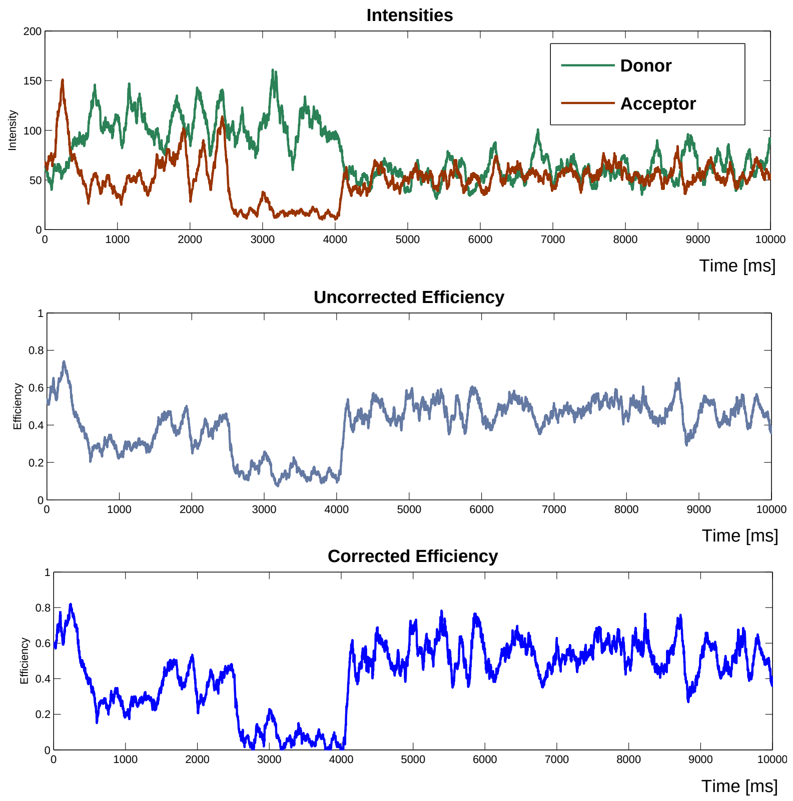


FIG. 4.8. Efficiencies generated from intensity values (middle) and photon counts (bottom). *Top,* the donor / acceptor intensities from time window averaging.

4.2.5. Variance versus Mean. Whereas in systems with well separated states the recognition process can be automated using threshold values this turned out to be inapplicable here since efficiency fluctuations of the states often overlapped. Nevertheless it is possible to choose states visually recognizable through looking at the efficiency behavior. Those manually selected states can be analyzed using the *NProbe* (see 3.6.3.1) generated mean value and variance for each state, representing the conformational flexibility rather rather than fluctuations originated through the measurement . The scattered plot depicted in 4.10 of all states then gives an overview about the system dynamic behavior.

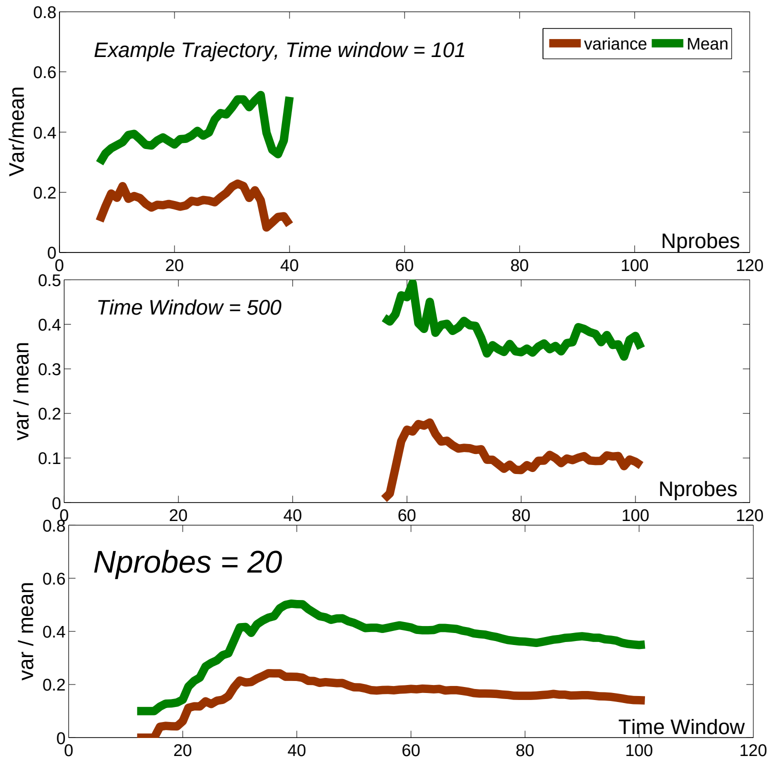


FIG. 4.9. Outcome of differing Nprobe parameter (see 3.6.3.1) and collection time-windows on an example trajectory. The variance (brown) and mean (green) show a plateau starting at 20 NProbes.

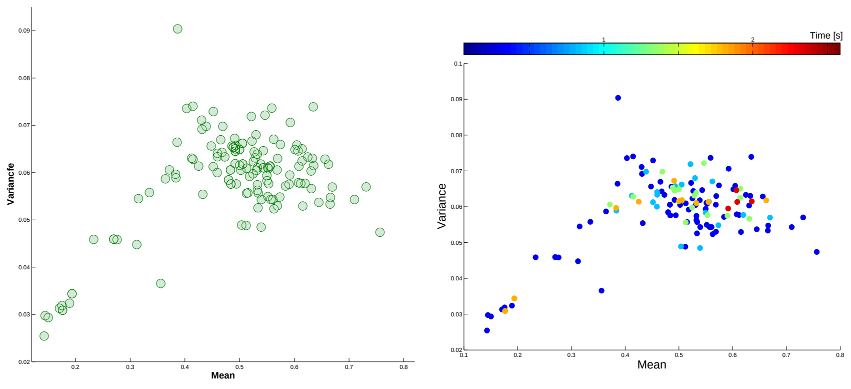


FIG. 4.10. Every circle represents a state. *Left:* Distribution of variance and mean calculated from method 3.6.3.1. *Right:* Coloring according to the duration of each state.

The method works fine for 2 to 40 Nprobes at small window sizes (10-100 counts) and produces a stable variance and mean within that range (see Fig. 4.9). A data representation for a collection of NProbe and time-window parameters from the stable region in Fig. 4.9. was calculated for all of the manually selected single states. The distribution in Fig. 4.10. showed a stable behavior (data not shown) within that range not degenerate for increasing collection time-windows as would occur with a naive variance / mean

The diagram 4.10 does not reveal clearly distinguishable states, but nevertheless can be interpreted in the way that there is a high occupancy of the middle region which could consist of many overlapping states, separated by their dwell time within that state (color in Fig. 4.10 b)) or their transition probability to other states. The amount of data does not allow to draw more direct conclusions from the plot so only coarse estimations are taken into the next analysis section. A possible clustering would be to look for dense aggregations of circles as done in Fig. 4.11below.

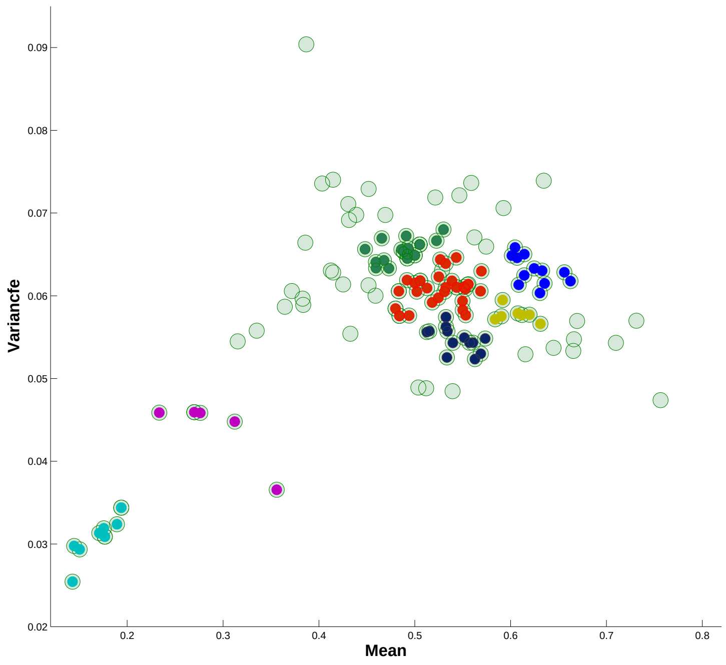


FIG. 4.11. Possible clustering of states according to visually recognizable accumulation of circles and used for initial state assignment for parameter estimation.

4.3. Distance Estimation

So far, we have regarded the trajectory in terms of efficiency values exclusively. There are two approaches that can produce the distance of the reporter dyes in nanometers that is of high importance for molecular biology. The first approach uses knowledge about the RNA structure, the second one uses the Förster law to infer the distance from the FRET efficiencies via Förster law (see Eq. 48).

4.3.1. Gaussian Chain of Nucleotides. From a geometric model that describes the molecule as an idealized chain of n stiff and interlinked rods (representing the 49 ribonucleic acids) building up a worm-like chain with $\mathsf{F} = 3\mathrm{n} - 2\mathrm{g}_\mathrm{N}$ degrees of freedom ($\mathrm{g}_\mathrm{N}$is the number of interlinks, thus g_N =48). One can calculate the probability distribution of possible arrangements of the 49 elements from the empirically determined contour length of a single nucleotide (6.3 Å) through the Gaussian chain model. The maximum (and very improbable) value is defined through the stretched out molecule contour length, 49×6.3 Å= 309 Å.

For comparison of the experimentally determined distance values a worm like chain with the measured persistence length $l_p = (10 - 21)$ Åas introduced for ribonucleic enzymes in [36] was employed by Nienhaus et al. yielding identical results. Therefore the Förster radius of 53 Ådetermined in this experiment can be regarded as the correct scaling factor for distance trajectory from the efficiency trajectory via Försters law.

4.4. Conformation Analysis

4.4.1. Identifying States. As discussed at the beginning of this chapter the traditional histogram method is not suitable for a kinetic investigation of the time-series. We therefore will employ a Hidden Markov Models for the unknown state trajectory to be revealed. States are no longer determined exclusively by their efficiency values but additionally taking their flexibility into account by calculating their variances.

This of cause can't be done naively using the definition of variance for the random variable X:

$$Var(X) := \langle\langle X - \mu\rangle^2\rangle = \langle X^2\rangle - \langle X\rangle^2. \tag{96}$$

Every measurement done at finite time resolution and discrete variables suffers from shot noise due to the Poissonian distribution of the observed events within the window length τ. We would like to separate the variance due to photonic shot noise (induced by the avalanche photon detectors) from the ”true” variance of the molecules current conformation. Therefore we will use the *NProbe* method introduced in Chapter 2 which is able to generate an efficiency trajectory free of shot noise by employing a slightly modified background correction: Instead of

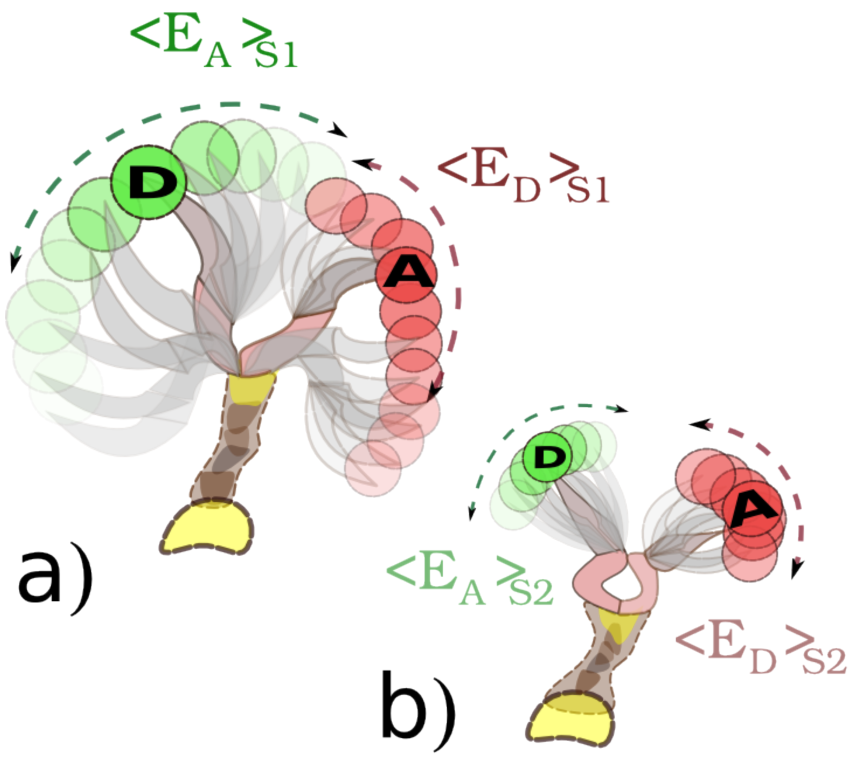


FIG. 4.12. Two distinct states: a) An unfolded situation where parts of the molecules (marked with the fluorophores D and A) can fluctuate widely. b) At the same mean distance the markers in the structurally different complex now have less mobility.

subtracting the background intensity from time-window based trajectories we delete a photon by chance at each point at the raw trajectory. The chances are determined by comparing the pseudo-random variable X generated by the computer and drawn from the uniform distributed interval $[0, 1]$ with the background values computed from 4.2.2 above:

$$P^A_{bg}(t^A_i \neq 0) = \begin{cases} t^A_i - 1 & for\ X < \alpha_A \\ t^A_i & else \end{cases}$$

$$P^D_{bg}(t^D_i \neq 0) = \begin{cases} t^D_i - 1 & for\ X < \alpha_D \\ t^D_i & else \end{cases}$$

and similarly the crosstalk and gamma factor can be treated in a probabilistic way by changing an acceptor photon into a donor photon by chance

$$P_{Xtalk}(\{t^D_i, t^A_i \neq 0\} = \begin{cases} \{t^D_i + 1, t^A_i - 1\} & for\ X < \alpha_A \\ \{t^D_i, t^A_i\} & else \end{cases} \tag{97}$$

$$P_\gamma(\{t^A_i \neq 0\} = \begin{cases} \{t^D_i + 1, t^A_i\} & for\ X > \gamma \\ \{t^D_i, t^A_i\} & else \end{cases} \tag{98}$$

The correction were applied to all available trajectories with proper bleaching phases.

4.4.2. State Estimation using a Hidden Markov Model. Now the reason we chose a Hidden Markov Model for the *state interchanging dynamics* becomes obvious when we would like to extract the optimal parameter of the currently assumed model from the experimental data bearing hidden states. The auto-regressive method introduced in chapter 2, section 3.4.2 yields an analytic expression for the likelihood function applicable to observational data bearing some hidden states. The method was implemented into the Java software library *Metamacs* and can be initialized by either providing a transition matrix or, as in the case employed here, through the Viterbi-path.

The Java-class HMMVAR (Hidden Markov models for multivariant processes) estimates the parameters for Gaussian behavior as well as dynamics described through stochastic differential equations. In fact the Gaussian estimation is a special case of general method. The parameters estimated are, as presented in the theory chapter, mean μ, variance σ^2 and the force $F = \nabla U$ resulting from the harmonic potential $U(x) = ax^2$, $a \in \mathbb{R}$ for every state.

4.4.3. Expectation maximization (EM). The idea of applying the EM algorithm is to maximize posterior probability (hence finding a maximum likelihood estimate) of our parameters λ responsible for the state changing behavior through alternation between *estimating the unknown variables* $\mathcal{D}_H$ that would have most likely come from the parameter distribution $\mathrm{P}(\lambda)$ and *maximizing the parameters* λ given the estimated, hidden data.

ALGORITHM 4.1 (**EM Algorithm**).

- ***Expectation-Step (E-Step):*** *Evaluate the expectation value* $Q(\lambda, \lambda_k) = E[\log\, P(\mathcal{D}_{Complete}|\lambda)|\mathcal{D}_{\mathcal{O}}, \lambda_k]$ *given the parameter estimates* λ_k.
- ***Maximization-Step (M-Step):*** *Define a new parameter set* λ_{k+1} *by maximizing the expectation:*

$$\lambda_{k+1} = \arg\max_{\lambda} Q(\lambda, \lambda_k). \tag{99}$$

The basic idea was first formulated by Hartley in 1958 and has become refined until a proof of convergence has been accomplished by Dempser, Laird an Rubin in 1977. In our case, the hidden sequence is the conformational state sequence. Whereas the EM method is a description of finding only the maximum likelihood of the parameters λ, we find that ,in the case of the hidden data being generated through Markov Jumps,one can use the EM algorithm for additionally estimating the posterior distribution. For a detailed description please refer to [37].

4.5. Results

A complete corrected efficiency trajectory was composed by appending the previously selected single state efficiency trajectories, generated form 500 ms averaged intensity values (46). The complete time series duration is 827000 time steps and is used for the following analysis methods.

4.5.1. Hidden Markov Estimation with Gaussian Output. At first appliance of the HMMVAR method the first iteration step of the EM algorithm already stalled and wrote "NaN" (Not a Number) to the console output. This can happen if the forward and backward probabilities are extremely close to zero, thereby running out of the range representable numbers defined through the 32 bit range. This can be caused through states defined in the Viterbi path that are not visited often enough. In fact, after excluding states below a length of 1 sec and dividing one trajectory of 43 sec duration (which wasn't exhibiting any visually detectable state changes) into 4 pieces the iterating started properly until the iterating limit or the precision threshold was reached.

Three State Model with Initial Mean Clustering. The 122 states were grouped into three cluster according to their mean generated from the NProbe method (with NP=5 and collecting window of 25, see 3.6.3.1) and a Viterbi path based on this clustering was generated.

$$\begin{aligned} s_1^\mu &= \{\mu_{NP}\} \subset \mu_{NP} \leq 0.5 \\ s_2^\mu &= \{\mu_{NP}\} \subset 0.5 < \mu_{NP} \leq 0.7 \\ s_3^\mu &= \{\mu_{NP}\} \subset 0.7 < \mu_{NP} \leq 1 \end{aligned} \tag{100}$$

After the EM algorithm finished the following (μ, σ) parameters emerged after calling the $getRegressionMatrices$ and $getCovariances$ method of HMMVAR:

$$\begin{aligned} \mu(s_1^\mu) = 0.3798 \quad &, \quad \sigma(s_1^\mu) = 0.0108 \\ \mu(s_2^\mu) = 0.6122 \quad &, \quad \sigma(s_2^\mu) = 0.0022 \\ \mu(s_3^\mu) = 0.5178 \quad &, \quad \sigma(s_3^\mu) = 0.0005 \end{aligned} \tag{101}$$

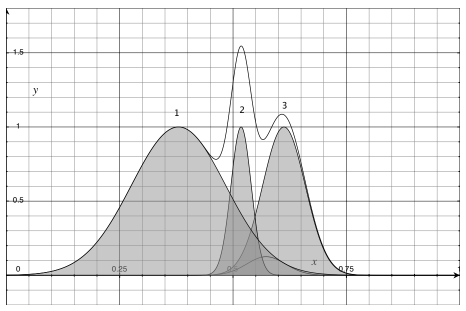


FIG. 4.13. HMM-Gauss estimated efficiency distribution of the three-state model with initial mean clustering (shaded curves) the unshaded curve is the unnormalized sum of the three Gaussians.

The estimated transition Matrix of the full efficiency trajectoreis after initialization with the Viterbi path from the mean clustering and 500 EM iteration steps yielded:

$$T_1^{ij} = \begin{array}{ccc} \mathbf{0.99858} & 0.00005 & 0.00137 \\ 0.00006 & \mathbf{0.99826} & 0.00168 \\ 0.00080 & 0.01308 & \mathbf{0.99789} \end{array} \tag{102}$$

The eigenvalues ε of this matrix are representative for the timescales governing the interchange rates between the states. $\varepsilon_1^1 = 1,\ \varepsilon_2^1 = 0,9984\ \varepsilon_3^1 = 0,9963$. Interpreting these values is only sensible if there are substantial reasons that the estimated system has physical relevance. The eigenvalues lead to the timescales τ_i via:

$$\tau_i = \log(\varepsilon_i)^{-1} \tag{103}$$

which can also be done for the diagonal transition matrix entries T^{ii} in order to get the lifetime of the metastable states. For T_1^{ij} the calculation of τ_i with $i = 1, 2, 3$ yields

$$\begin{aligned} \tau_1^1 &= 131\,sec \\ \tau_2^1 &= \infty\,sec \\ \tau_3^1 &= 622918\,sec \end{aligned} \tag{104}$$

This shows that one has to be very carefully with those estimations. Since the whole length of the input trajectory is shorter than the estimated timescale of the third state, this value is unlikely to have any relevance. A value of infinity also is senseless for a metastable state.

Three State Model with Initial Variance Clustering. The same procedure was done for a variance based clustering:

$$\begin{aligned} s_1^\sigma &= \{\sigma_{NP}\} \subset \sigma_{NP}^2 \leq 0.04 \\ s_2^\sigma &= \{\sigma_{NP}\} \subset 0.04 < \sigma_{NP}^2 \leq 0.08 \\ s_3^\sigma &= \{\sigma_{NP}\} \subset 0.08 < \sigma_{NP}^2 \leq 0.1 \end{aligned} \tag{105}$$

yielding the state parameters

$$\begin{aligned} \mu(s_1^\sigma) = 0.4557 \quad &, \quad \sigma(s_1^\sigma) = 0.0058 \\ \mu(s_2^\sigma) = 0.5190 \quad &, \quad \sigma(s_2^\sigma) = 0.0119 \\ \mu(s_3^\sigma) = 0.5267 \quad &, \quad \sigma(s_3^\sigma) = 0.0007 \end{aligned} \tag{106}$$

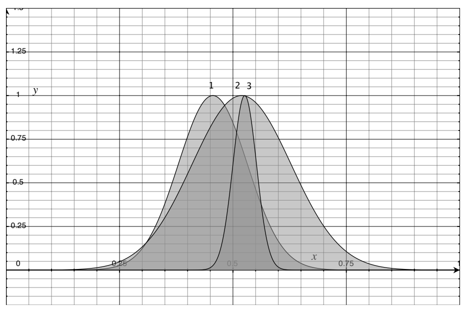


FIG. 4.14. HMM-Gauss estimated efficiency distribution of the three-state model with initial variance clustering

with the corresponding transition Matrix

$$T_2^{ij} = \begin{array}{ccc} \mathbf{0.99987} & 0.00013 & 0 \\ 0.00006 & \mathbf{0.99999} & 0.00168 \\ 0 & 0.00053 & \mathbf{0.997947} \end{array} \tag{107}$$

The matrix estimated here shows almost non-ergodic behavior because of the probability for the $S_2 \rightarrow S_2$ of value is extremely close to 1. This implies that as soon as the system enters state 2 it will stay there for a very long time until the $0.0001\ \%$ chance of leaving eventuates. Furthermore the zero probability for the $S_1 \rightleftharpoons S_3$ interchanges give rise to a possible sub-partitioning of the system. The eigenvalues $\varepsilon_1^1 = 1,\ \varepsilon_2^1 = 0,9999\ \varepsilon_3^1 = 0,9995$ yield millions of seconds and are therefore uninteresting and omitted.

Six State Model With Initial Mean Clustering. An advantage of HMM analysis is the possibility of concluding the number of states inherent to the investigated system though initial overestimation. The eigenvalues spectrum of the transition matrix states that are not metastable are separated through a spectral gap and indicate the number of states before the gab are sufficient for the metastable behavior whereas the remaining states can be disregarded.

For the initial clustering the following thresholds were used:

$$\begin{aligned}
\tilde{s}_1^{\mu} &= \{\mu_{NP}\} \subset \mu_{NP} \leq 0.3 && (108)\\
\tilde{s}_2^{\mu} &= \{\mu_{NP}\} \subset 0.3 < \mu_{NP} \leq 0.45 && (109)\\
\tilde{s}_3^{\mu} &= \{\mu_{NP}\} \subset 0.45 < \mu_{NP} \leq 0.55 && (110)\\
\tilde{s}_4^{\mu} &= \{\mu_{NP}\} \subset 0.55 < \mu_{NP} \leq 0.65 && (111)\\
\tilde{s}_5^{\mu} &= \{\mu_{NP}\} \subset 0.65 < \mu_{NP} \leq 0.7 && (112)\\
\tilde{s}_6^{\mu} &= \{\mu_{NP}\} \subset 0.7 < \mu_{NP} \leq 1 && (113)
\end{aligned}$$

resulting in the estimation

$$\begin{aligned}
\mu(\tilde{s}_1^{\mu}) = 0.4902 \quad &, \quad \sigma(\tilde{s}_1^{\mu}) = 0.0002\\
\mu(\tilde{s}_2^{\mu}) = 0.1976 \quad &, \quad \sigma(\tilde{s}_2^{\mu}) = 0.0012\\
\mu(\tilde{s}_3^{\mu}) = 0.5277 \quad &, \quad \sigma(\tilde{s}_3^{\mu}) = 0.0001\\
\mu(\tilde{s}_4^{\mu}) = 0.6389 \quad &, \quad \sigma(\tilde{s}_4^{\mu}) = 0.0019\\
\mu(\tilde{s}_5^{\mu}) = 0.4216 \quad &, \quad \sigma(\tilde{s}_5^{\mu}) = 0.0010\\
\mu(\tilde{s}_6^{\mu}) = 0.5652 \quad &, \quad \sigma(\tilde{s}_6^{\mu}) = 0.0002
\end{aligned} \tag{114}$$

The estimated transition Matrix after initialization and 100 EM iteration steps yielded:

$$T_3^{\mu\nu} = \begin{array}{cccccc}
\mathbf{0.99396} & 0,00001 & 0,00467 & 0,00003 & 0,00134 & 0\\
0,00008 & \mathbf{0.99962} & 0,00006 & 0,00021 & 0 & 0,0002\\
0,00548 & 0,00001 & \mathbf{0.99131} & 0,00318 & 0,00002 & 0\\
0,00002 & 0,00002 & 0,00001 & \mathbf{0.99502} & 0,00007 & 0,0048\\
0 & 0,00007 & 0,00151 & 0 & \mathbf{0.99841} & 0\\
0 & 0 & 0,00356 & 0,00177 & 0 & \mathbf{0.99467}
\end{array} \tag{115}$$

The corresponding eigenvalues ε^3 are

$$\varepsilon_1^3 = 0,9885,\ \varepsilon_2^3 = 0.9908\ \varepsilon_3^3 = 0,9957\ \varepsilon_4^3 = 0,9985\ \varepsilon_5^3 = 1.0\ \varepsilon_6^3 = 0,9996 \tag{116}$$

with the timescales computed from Eq. 103

$$\tau_1^3 = 8678\, sec\, \tau_2^3 = 10867\, sec\, \tau_3^3 = 23625\, sec\, \tau_4^3 = 57706\, sec\, \tau_5^3 = 4.5{\cdot}10^{18}\, sec\, \tau_6^3 = 2,5{\cdot}10^6 \tag{117}$$

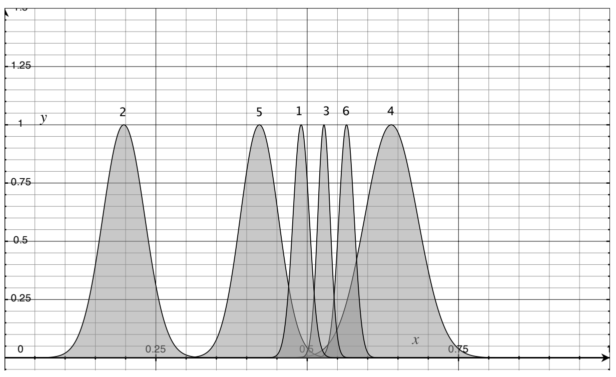


FIG. 4.15. HMM-Gauss estimated efficiency distribution (y-Axis) of the three-state model with initial mean clustering.

Six State Model with Initial Variance Clustering.

$$\begin{aligned}
\tilde{s}^{\sigma}_1 &= \{\sigma_{NP}\} \subset \sigma^2_{NP} \leq 0.04 \\
\tilde{s}^{\sigma}_2 &= \{\sigma_{NP}\} \subset 0.04 < \sigma^2_{NP} \leq 0.08 \\
\tilde{s}^{\sigma}_3 &= \{\sigma_{NP}\} \subset 0.08 < \sigma^2_{NP} \leq 0.1 \\
\tilde{s}^{\sigma}_4 &= \{\sigma_{NP}\} \subset \sigma^2_{NP} \leq 0.04 \\
\tilde{s}^{\sigma}_5 &= \{\sigma_{NP}\} \subset 0.04 < \sigma^2_{NP} \leq 0.08 \\
\tilde{s}^{\sigma}_6 &= \{\sigma_{NP}\} \subset 0.08 < \sigma^2_{NP} \leq 0.1
\end{aligned} \tag{118}$$

resulting in the mean and variance estimation

$$\begin{aligned}
\mu(\tilde{s}^{\sigma}_1) = 0.4334 \quad &, \quad \sigma(\tilde{s}^{\sigma}_1) = 0.0007 \\
\mu(\tilde{s}^{\sigma}_2) = 0.4954 \quad &, \quad \sigma(\tilde{s}^{\sigma}_2) = 0.0002 \\
\mu(\tilde{s}^{\sigma}_3) = 0.3662 \quad &, \quad \sigma(\tilde{s}^{\sigma}_3) = 0.0504 \\
\mu(\tilde{s}^{\sigma}_4) = 0.6256 \quad &, \quad \sigma(\tilde{s}^{\sigma}_4) = 0.0006 \\
\mu(\tilde{s}^{\sigma}_5) = 0.5647 \quad &, \quad \sigma(\tilde{s}^{\sigma}_5) = 0.0002 \\
\mu(\tilde{s}^{\sigma}_6) = 0.5297 \quad &, \quad \sigma(\tilde{s}^{\sigma}_6) = 0.0001
\end{aligned} \tag{119}$$

The estimated transition Matrix after initialization and 100 EM iteration steps yielded:

$$T^{\mu\nu}_4 = \begin{array}{cccccc}
\mathbf{0.99717} & 0,00238 & 0,00042 & 0,00002 & 0 & 0 \\
0,00216 & \mathbf{0.98716} & 0,000017 & 0,00003 & 0,00285 & 0,00778 \\
0,00073 & 0,00008 & \mathbf{0.99868} & 0,00047 & 0,00004 & 0 \\
0,00002 & 0,00002 & 0,000281 & \mathbf{0.99793} & 0,00175 & 0 \\
0 & 0,00829 & 0,00151 & 0,001836 & \mathbf{0.99501} & 0 \\
0 & 0 & 0 & 0 & 0 & \mathbf{0.99171}
\end{array} \tag{120}$$

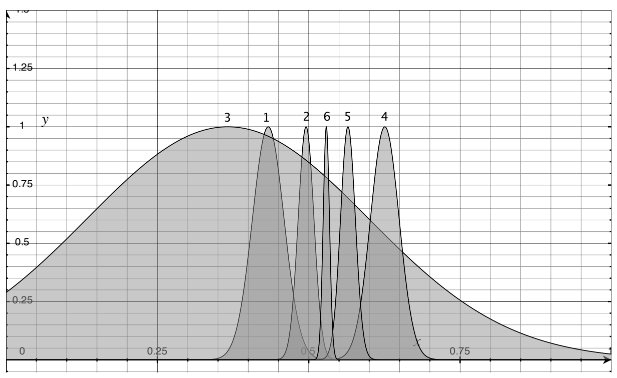


FIG. 4.16. HMM-SDE estimated distribution of the three-state model with initial mean clustering.

with the corresponding eigenvalues ε^4

$$\varepsilon_1^4 = 0,9805,\ \varepsilon_2^4 = 0.9938\ \varepsilon_3^4 = 0,9964\ \varepsilon_4^4 = 1,0\ \varepsilon_5^4 = 0.9983\ \varepsilon_6^4 = 0,9987$$

and their timescales from Eq. 103

$$\tau_1^3 = 5,1{\cdot}10^4\ sec\,\tau_2^3 = 1.6{\cdot}10^5\ sec\ \tau_3^3 = 2.7{\cdot}10^6\ sec\ \tau_4^3 = 2.2{\cdot}10^{18}\ sec\ \tau_5^3 = 6.2{\cdot}10^6\ sec\ \tau_6^3 = 7,4{\cdot}10 \tag{121}$$

4.5.2. Hidden Markov Estimation with Stochastic Differential Equations. The same procedure was done for the SDE parameter estimation (see 3.4.3) providing us with the additional mean force parameter F and a modified variance Σ. The variance clustering was changed since a minimum number of assignments of every state is necessary for the initialization of the metamacs HMMVAR algorithm.

Three State Model with Initial Mean Clustering. Using the same mean groups to assign initial states we find after applying the EM algorithm:

$$\begin{array}{lll} \mu(s_1^\mu) = 0.525256 & \Sigma(s_1^\mu) = 3,9 \cdot 10^{-6} & F(s_1^\mu) = -7,6348 \\ \mu(s_2^\mu) = 0,364889 & \Sigma(s_2^\mu) = 1,5 \cdot 10^{-6} & F(s_2^\mu) = -11,1737 \\ \mu(s_3^\mu) = 0,505443 & \Sigma(s_3^\mu) = 0,006517 & F(s_3^\mu) = -0,4474 \end{array} \tag{122}$$

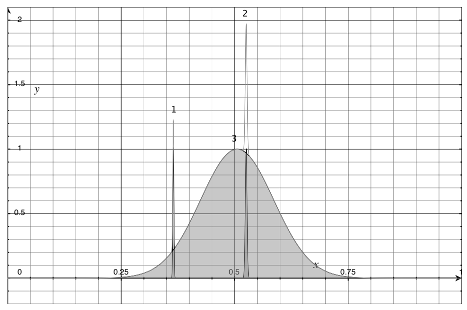


FIG. 4.17. HMM-SDE estimated distribution of the three-state model with initial mean clustering

we also get the transition matrix

$$T_5^{\mu\nu} = \begin{matrix} \mathbf{0,9919} & 0.0075 & 0,00060 \\ 0.0097 & \mathbf{0,9901} & 0,00018 \\ 0,8341 & 0.1659 & \mathbf{4 \cdot 10^{-57}} \end{matrix} \tag{123}$$

which clearly is degenerated in the last entry (3,3). This can mean that there are only two states. The eigenvalues ε^5 also represent this situation:

$$\varepsilon_1^5 = -0,0005,\ \varepsilon_2^5 = 1\ \varepsilon_3^5 = 0,9825 \tag{124}$$

$$\begin{aligned} \tau_1^5 &= 2.2 \cdot 10^{18}\, sec \\ \tau_2^5 &= 8,2 \cdot 10^{6}\, sec \\ \tau_3^5 &= 3,1 \cdot 10^{6}\, sec \end{aligned} \tag{125}$$

Three State Model With Initial Variance Clustering. As mentioned before a slightly modified variance grouping had to be employed in order to utilize the HMMVAR estimation:

$$\begin{aligned} s_1^\sigma &= \{\sigma_{NP}\} \subset \sigma_{NP}^2 \leq 0.04 \\ s_2^\sigma &= \{\sigma_{NP}\} \subset 0.04 < \sigma_{NP}^2 \leq 0.06 \\ s_3^\sigma &= \{\sigma_{NP}\} \subset 0.06 < \sigma_{NP}^2 \leq 0.1 \end{aligned} \tag{126}$$

giving us the state parameter

$$\begin{matrix} \mu(s_1^\sigma) = 0,506491 & \Sigma(S_1^\sigma) = 0,006441 & F(s_1^\sigma) = -0,4551 \\ \mu(s_2^\sigma) = 0,525336 & \Sigma(S_2^\sigma) = 3,8 \cdot 10^{-6} & F(s_2^\sigma) = -7,6212 \\ \mu(s_3^\sigma) = 0,358468 & \Sigma(S_3^\sigma) = 1,5 \cdot 10^{-6} & F(s_3^\sigma) = -11,2019 \end{matrix} \tag{127}$$

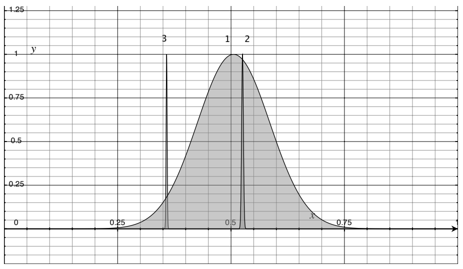


FIG. 4.18. HMM-SDE estimated distribution of the three-state model with initial variance clustering

and transition matrix

$$T_6^{\mu\nu} = \begin{array}{ccc} 2\cdot 10^{-56} & 0,8361 & 0,1639 \\ 0,0006 & \mathbf{0},9925 & 0,00168 \\ 0,0002 & 0,0089 & \mathbf{0},9909 \end{array} \tag{128}$$

with eigenvalues ε^6

$$\varepsilon_1^6 = -0,0005,\ \varepsilon_2^6 = 1\ \varepsilon_3^6 = 0,9841$$

and timescales from Eq. 103

$$\begin{aligned} \tau_1^6 &= 4.5\cdot 10^{18}\, sec \\ \tau_2^6 &= 7,0\cdot 10^{6}\, sec \\ \tau_3^6 &= 1,9\cdot 10^{6}\, sec \end{aligned} \tag{129}$$

Six State Model With Initial Mean Clustering.

$$\begin{array}{lll} \mu(s_1^\mu) = 0,527441 & \Sigma(s_1^\mu) = 6,7\cdot 10^{-6} & F(s_1^\mu) = -6,5851 \\ \mu(s_2^\mu) = 0,475339 & \Sigma(s_2^\mu) = 2,8\cdot 10^{-6} & F(s_2^\mu) = -7,3067 \\ \mu(s_3^\mu) = 0,496208 & \Sigma(s_3^\mu) = 0,8\cdot 10^{-6} & F(s_3^\mu) = -10,2744 + \pi i \\ \mu(s_4^\mu) = 0,527329 & \Sigma(s_4^\mu) = 1,7\cdot 10^{-6} & F(s_4^\mu) = -8,0501 \\ \mu(s_5^\mu) = 0,469561 & \Sigma(s_5^\mu) = 1,5\cdot 10^{-6} & F(s_5^\mu) = -4,4394 + \pi i \\ \mu(s_6^\mu) = 0,496797 & \Sigma(s_6^\mu) = 0,014688 & F(s_6^\mu) = -0,2087 \end{array} \tag{130}$$

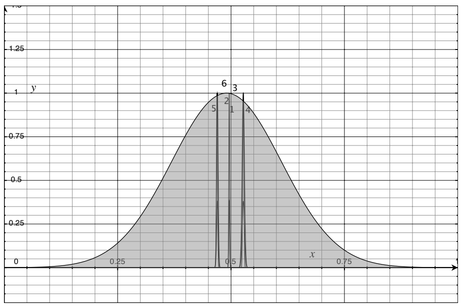


FIG. 4.19. HMM-Gauss estimated efficiency distribution of the six-state model with initial mean clustering

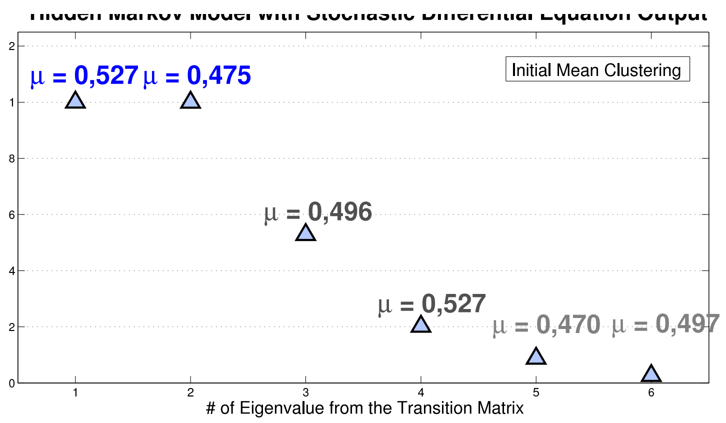


FIG. 4.20. Eigenvalues of the transition matrix $T_7^{\mu\nu}$

the complex values above are due to negative entries in the regression matrix from which the force F is computed via the logarithm. There is no physical interpretation of these complex values. The transition matrix for the six states above is

$$T_7^{\mu\nu} = \begin{array}{cccccc} \mathbf{0.51998} & 1\cdot 10^{-11} & 2\cdot 10^{-7} & 0,48002 & 9\cdot 10^{-12} & 0 \\ 6\cdot 10^{-11} & \mathbf{0,69823} & 0,30076 & 0,00063 & 0 & 0,00037 \\ 5\cdot 10^{-7} & 0,17971 & \mathbf{0.79822} & 0,00002 & 0,02205 & 0 \\ 0,31873 & 0,00008 & 2\cdot 10^{-6} & \mathbf{0.68073} & 0,00020 & 0,00026 \\ 0 & 7\cdot 10^{-6} & 0,88127 & 0 & \mathbf{0.11873} & 0 \\ 0 & 0 & 0,32595 & 0,64785 & 0 & \mathbf{0.02620} \end{array} \tag{131}$$

with corresponding eigenvalues ε^7in descending order

$$\varepsilon_1^7 = 1,0,\ \varepsilon_2^7 = 0.9994\ \varepsilon_3^7 = 0,5280\ \varepsilon_4^7 = 0,2014\ \varepsilon_5^7 = 0,0875\ \varepsilon_6^7 = 0.0258$$

yielding the timescales from Eq. 103

$$\tau_1^7 = 2,3\cdot 10^{18}\, sec\ \tau_2^7 = 1.86\cdot 10^6\, sec\ \tau_3^7 = 1.6\cdot 10^3 \tag{132}$$

$$sec\ \tau_4^7 = 624\, sec\ \tau_5^7 = 273\, sec\ \tau_6^7 = 410\, sec$$

Six State Model With Initial Variance Clustering. The SDE estimation for the six-state model initially grouped by the state mean value is

$$\begin{array}{lll} \mu(s_3^\sigma) = 0,480040 & \Sigma(s_3^\sigma) = 7,2\cdot 10^{-6} & F(s_3^\sigma) = -7,5464 \\ \mu(s_3^\sigma) = 0,542539 & \Sigma(s_3^\sigma) = 2,5\cdot 10^{-6} & F(s_3^\sigma) = -10,3157 \\ \mu(s_3^\sigma) = 0,541512 & \Sigma(s_3^\sigma) = 4,1\cdot 10^{-6} & \mu(s_3^\sigma) = -5,8526 \\ \mu(s_3^\sigma) = 0,482903 & \Sigma(s_3^\sigma) = 1,5\cdot 10^{-6} & \mu(s_3^\sigma) = -9,2420 \\ \mu(s_3^\sigma) = 0,285305 & \Sigma(s_3^\sigma) = 0,7\cdot 10^{-6} & \mu(s_3^\sigma) = -10,7970 \\ \mu(s_3^\sigma) = 0,485318 & \Sigma(s_3^\sigma) = 0,016191 & \mu(s_3^\sigma) = -0,3813 \end{array} \tag{133}$$

with the corresponding matrix

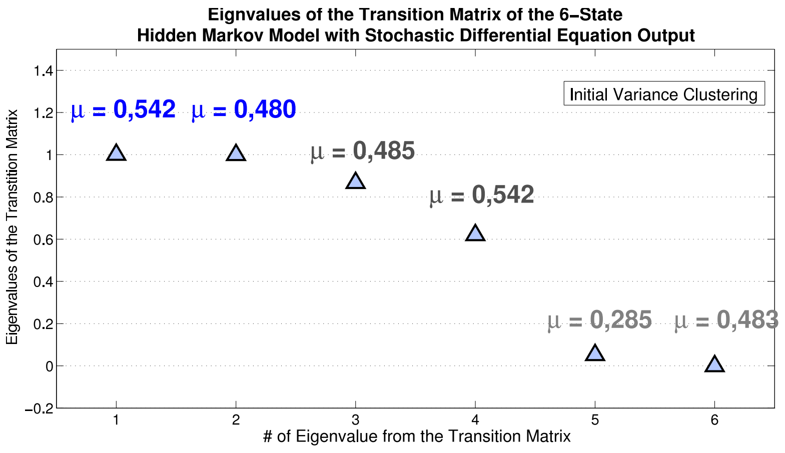


FIG. 4.22. Eigenvalues of the transition matrix $T_8^{\mu\nu}$

$$T_8^{\mu\nu} = \begin{array}{cccccc} \mathbf{0.3601} & 0,0014 & 0,0786 & 0,5599 & 0 & 0 \\ 0,0011 & \mathbf{0,8099} & 03 \cdot 10^{-6} & 6 \cdot 10^{-8} & 0,1882 & 0,00076 \\ 0,1118 & 2 \cdot 10^{-6} & \mathbf{0.8880} & 0,00016 & 2 \cdot 10^{-19} & 0 \\ 0,3291 & 9 \cdot 10^{-6} & 0,0014 & \mathbf{0.6695} & 7 \cdot 10^{-17} & 0 \\ 0 & 0,1918 & 2 \cdot 10^{-16} & 2 \cdot 10^{-16} & \mathbf{0.8082} & 0 \\ 0 & 1 & 0 & 0 & 0 & \mathbf{2 \cdot 10^{-21}} \end{array} \tag{134}$$

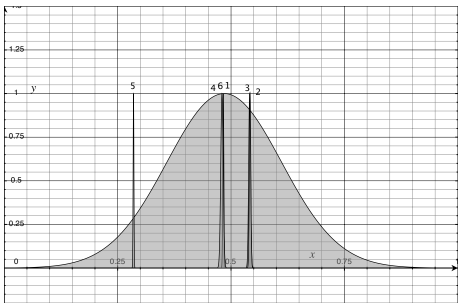


FIG. 4.21. HMM-SDE estimated efficiency distribution of the six-state model with initial variance clustering

giving us the eigenvalues ε^8

$$\varepsilon_1^8 = 1,0\; \varepsilon_2^8 = 0.9991\; \varepsilon_3^8 = 0,8664\; \varepsilon_4^8 = 0,61960\; \varepsilon_5^8 = 0,0516\; \varepsilon_6^8 = -0,0009$$

and timescales

$$\tau_1^8 \;=\; 337\,sec\;\tau_2^8 = 119\,sec\;\tau_3^8 = 6973\,sec \tag{135}$$

$$\tau_4^8 = 1.3 \cdot 10^{18}\,sec\;\tau_5^8 = 1.03\,sec \cdot 10^6\;\tau = 2089\,sec$$

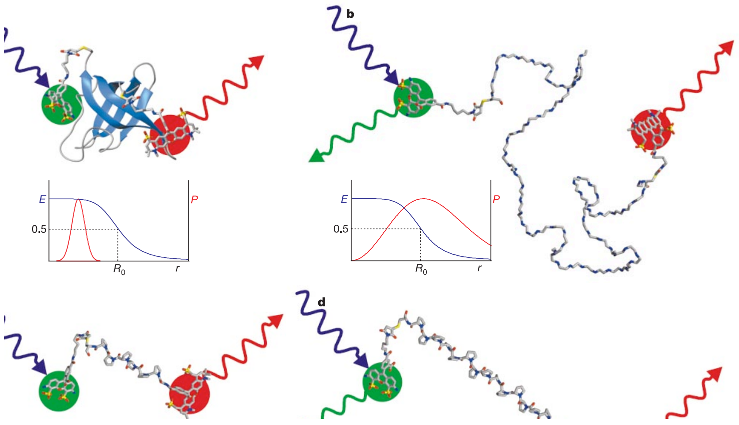


FIG. 4.23. *a)* The folded molecule bears the two fluorophores (colored balls) at a close distance, therefor a high FRET efficiency (peak of the red curve) is measured. *b)* at distant location the FRET efficiency (blue line in the insets) is low, additionally the structure of the molecule between donor (green) and acceptor (red) determines the variance as depicted in the inset. *c)* A situation similar to the molecule studied here with only few base pairs between the fluorophores *c)* the fully stretched out molecule has a low FRET value but a narrow distribution. Picture taken from [38]

4.5.3. Simulated Data from Transition Matrix. For a comparison of the model A simulated we can generate a sample trajectory from our parameter-set λ and transition matrix T^{ij} with $i, j = 1, .., n$ number of states. First we generate the hidden state trajectory by comparing the outcome of a uniform distributed random variable (normalized to unity) with the row entries of T^{ij} for the corresponding state, i.e. if the system is in state $s = 3$ at t_0, $\mathrm{T}^{3\mathrm{j}} = (\mathrm{T}^{31}, ..., \mathrm{T}^{3\mathrm{n}})$. The values of this row vector now determine the probability distribution for the system to be found in the next time-step. For meta-stable, stochastic matrices the diagonal entries are close to one, therefore the highest probability for $t_1$is to stay in state 3. In rare cases the pseudo-random variable gives a number that falls into the small fraction of values that cause a jump of the system into the next state.

The resulting trajectory then is either generated by drawing random variables from the normal distribution corresponding to the current state in case of a HMM or regression of the SDE with corresponding parameters (μ_n, F_n, Σ_n). A simulated efficiency trajectory from a Hidden Markov Model with SDE output and its estimated states is shown in Fig. 4.25. The state changes are as rapid as in the estimation. A more infromative result would be a state changing behaviour that meets the visual impression which could not be achieved here.

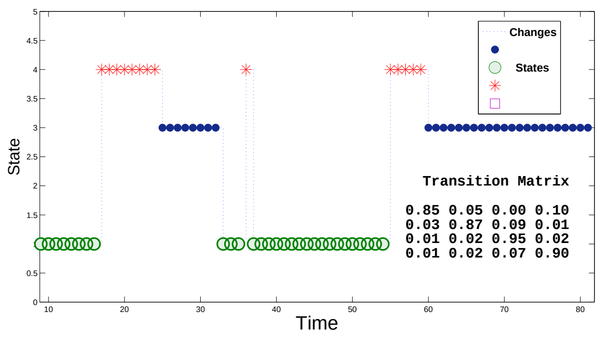


FIG. 4.24. Simulated Viterbi-path for a random 4 state-series generated from the Transition Matrix displayed as inset. Only three states have been visited during the very simulation of 80 time steps.

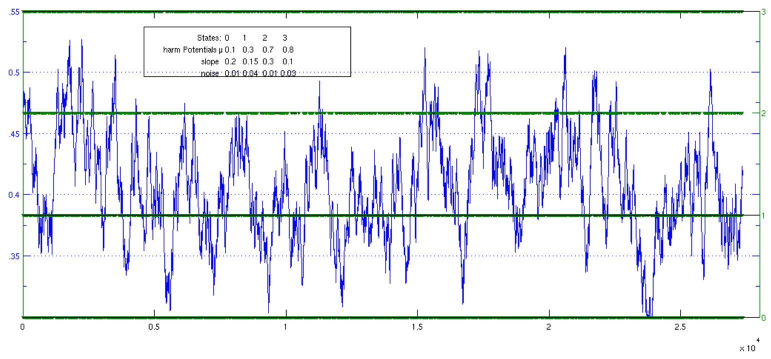


FIG. 4.25. Simulated efficiency trajectory from a SDE for a similar Viterbi-path as in Fig. 4.24 at much longer duration (ca 23000 time steps). The SDE parameter slope and noise for each state are shown in the inset.

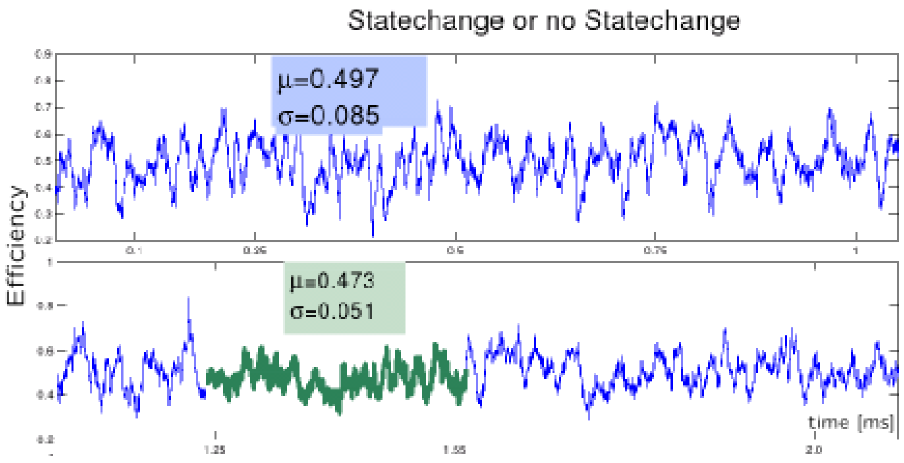
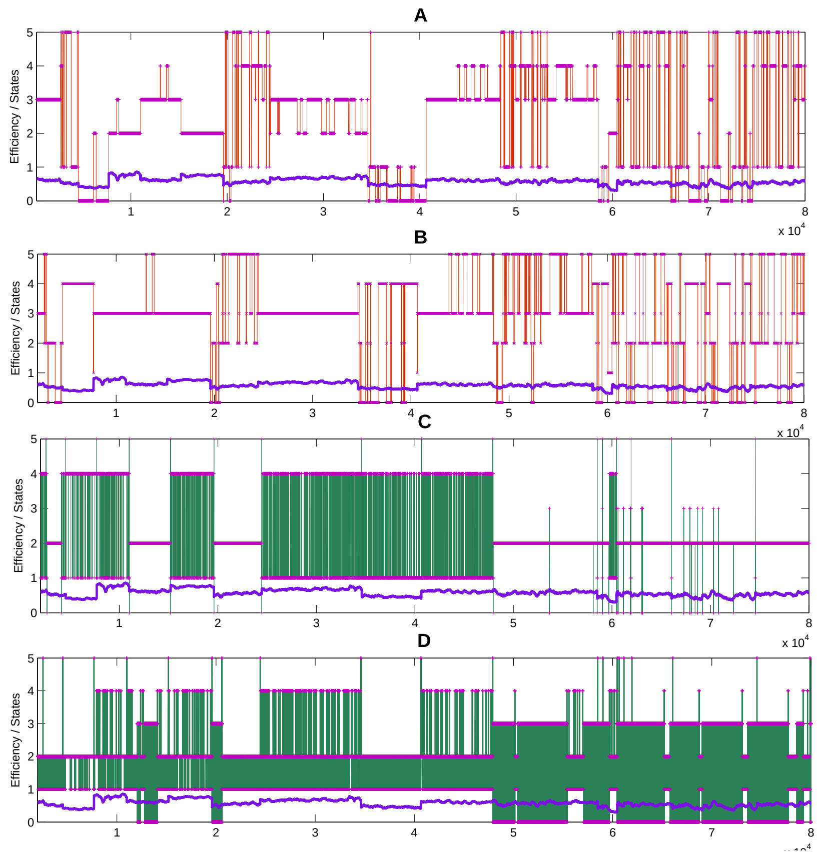


FIG. 4.26. Example efficiency trajectory with a possible state change with similar mean values suggesting a scenario as in Fig. 4.12.

If there are states characterized by a similar mean but differing in their flexibility, a problem arises in visual identification as Fig. 4.26 below suggests. This is the great advantage of the employed algorithm.



,

FIG. 4.27. A) Estimated Viterbi path (orange lines) of the 6-state model with initial mean clustering from the efficiency trajectory (blue trajectory). The states (purple dots) represent a state characterized by either only the mean efficiency value (A and B) from Eq. 119. or the SDE parameters (C and D) in Eq. 133.

4.5.4. Comparison: Simulation and Reality. The simulated trajectory in Fig. 4.25 visually meets the experimental outcome quite well. A quantitative comparison should rather look at the interchange rates that have been determined by other methods as well. The Viterbi path (Fig. 4.27) should reassemble the visual impression of state changes. However this is very hard to achieve and goes beyond the scope of this thesis.

If the estimated results are reliable, meaning reproducible and are relevant for understanding the dynamics of the molecular system, the parameter characterizing the states can give insight into the properties of the system (see Fig. 4.23). Unfortunately this could not be accomplished in the analysis shown here but would probably be possible with further work on data filtering and processing.

CHAPTER 5

Conclusion

5.1. A Novel Kinetic Intepretation

It has been shown that the folding behavior of the small Diels-Alder ribozyme exhibits metastable behavior and that the states corresponding to the metastable sets can be determined without the use of traditional methods as e.g. the correlation function used widely in the analysis of inter system conversion rates. This was facilitated through the use of Hidden Markov Models for the description of the conformational dynamics of the molecule. The model offers the novel avenue of computing representative parameters using the information of the complete trajectory. Though the power of those model has long been known and implemented in speech recognition or weather prediction, it still has little influence towards the interpretations of data from biological systems acquired by experimental physicists today. We have applied a modeling of the three state system proposed by Nienhaus et al for the catalytic ribozyme Diels Alderase to the efficiency trajectories of the individual molecule providing a different view on the system 1st in terms efficiency analysis $E(t)$ rather than analyzing the distance $r(t)$ and secondly by redefining the states as characterized by their conformational fluctuation. We furthermore constructed an extended model having six states and were able to affirm the first 3 states as well as discovering further states indicating a complex structure of the folding path. As an novel result we were able to identify the dwell time τ of a folded state at an efficiency value of $E = 0.19$ at τ =118 s. The criteria defining the states in our investigation were the mean Förster resonance energy transfer efficiency and its variance, which are especially applicable for the representation of bio-molecular processes as discussed in 2.3. As an additional advantage, the probabilistic approach in 4.1.1 enabled us to separate the signal noise from the molecular fluctuations preparing the ground for further analysis. The method enables unbiased detection of conformational states and is becoming more important since the number of states treated in the model can be much higher than three or six, therefore a Markovian modeling could potentially treat complex systems of interacting molecules.

5.1.1. Ambiguous Results. The interpretation is a hard task since a visual correspondence can't always be seen as the Fig. 4.27. confirms. Depending on the choice of averaging time window lengths, the outcome can be very ambiguous, demanding a solid initialization of the Markov Model. Further research may be able to construct a framework, which is able to treat experimental data in a robust way, making it accessible to experimentalist as a standard tool. Further investigations have to be undertaken in both, the physical system of interacting molecules, and the algorithmic procedure of analyzing the date.

Insight to the cause and course of molecular processes fundamental for the emergence of life have been enriched through the considerable growth of computational power and mathematical laws and principles that are able to describe complex systems and reveal further details [21] . Still, a detailed understanding of the catalytic activity observed in even the smallest ribo nucleic acids, able to accelerate the formation of a new carbon-carbon bound between two reactant molecules, is missing. This may originate from traditional methods lacking an account for the kinetic parameters inherent to the system. The parameters from the model used in this work reflect the conformational situation of the molecule and are adapted to the recorded data, therefore allowing us in principle to detect heterogeneity in many molecular system dynamics between single molecules. As proposed by [39] the presence of both, fast folding paths that are visiting a diversity of hundreds of states at a sub-microsecond scale as well as long persistent conformations are necessary for the successful catalyzation of the Diels-Alder reaction by a ribozyme. We can emphasize this view and provide a tool for kinetic analysis applicable for a wide range of systems governed by probabilistic processes, which are determined through conformational states inaccessible for direct measurement.

5.2. Outlook

So far we have shown that a good foundation for algorithmic treatment of experimental data can be achieved if the following preliminaries are fulfilled:

- Discrimination of adequate observational data conform to the model
- Correction (or model incorporation) of distorting artifacts in the measurement such as blinking
- Separation of detection noise and intrinsic fluctuations (i.e. shot-noise and conformational flexibility)

Essential for successful analysis of experimental data is the first point, which is already a growing subject in research, providing the ground for a close cooperation of experimentalists and theoretical scientists. The second point can, in principle, be incorporated in the estimation process rather than applied in beforehand. Mastering the third point will ultimately uncloud the view on conformation analysis in many life-science areas.

Acknowledgement

I would like to thank the following people for their support and guidance during the making of the thesis:

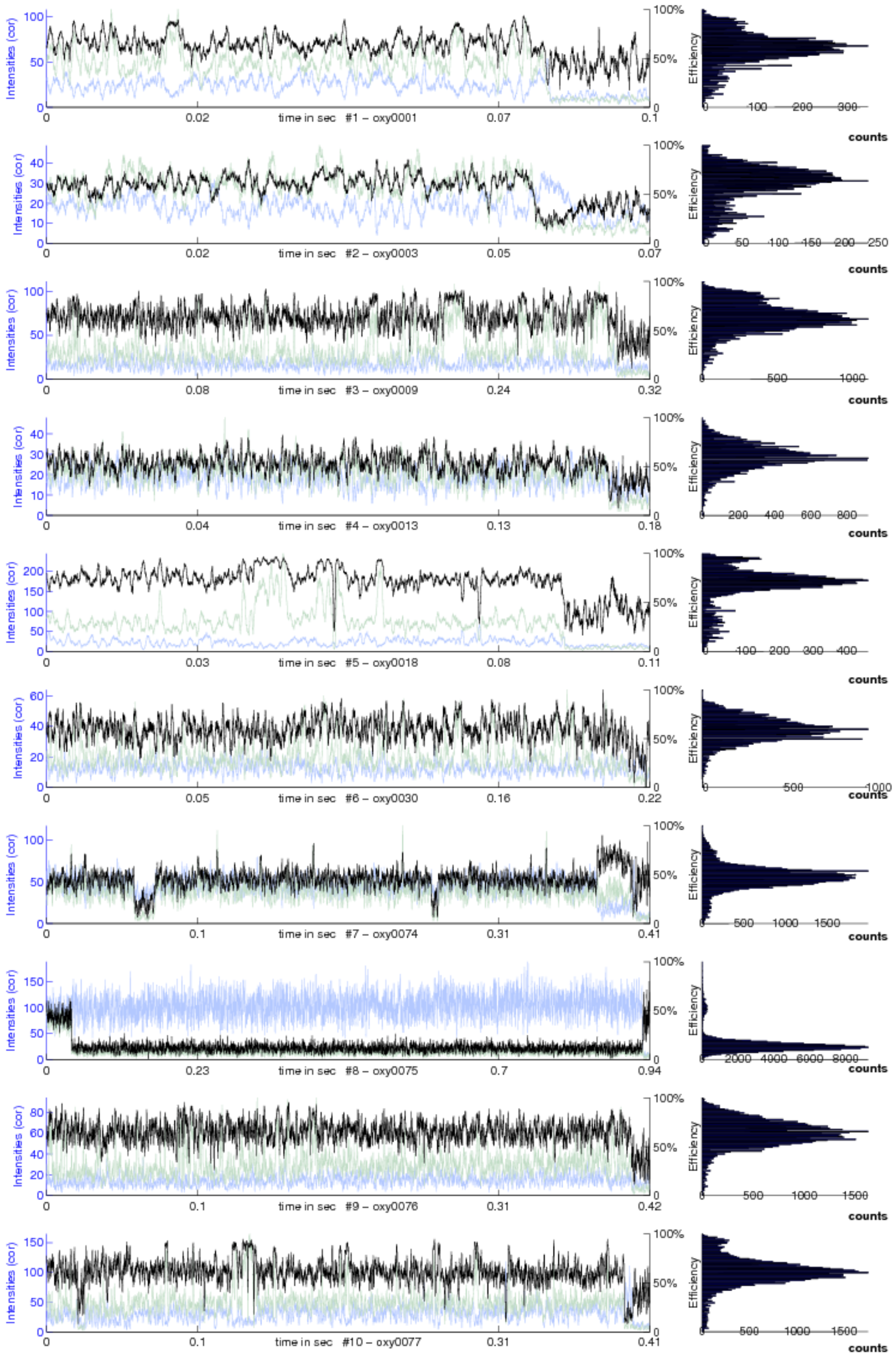
- **Prof. Eckehard Schöll,** director of the Institut für Theoretische Physik at the TU-Berlin who agreed to the proposal of a cooperation with the Bio computing group at the FU-Berlin.
- **Dr. Frank Noé**, who was an excellent supervisor and friend who greatly supported my understanding of computational science and provided me with insights into bio-molecular methods.
- **Prof. Christoph Schütte**, director of the Institute of Mathematics and Computer Science at the FU-Berlin, who exquisitely managed the task of juggling diverse areas of mathematics, informatics and biophysics.
- **Martin Fischbach**, for providing his Metamacs and general Java-knowledge every time I was producing 'NaN' or similar results.
- **Franz-Josef Schmitt** for his experimental expertise that helped me to understand the diverse fluorescent microscopy methods and problems and for pointing out errors and giving me hints for this thesis.
- **Martin Held** and **Jan Wigger** with whom I shared an office and who I could ask immediately and always received support if i encountered problems of scientific, numerical or computer specific matter.
- **Dr. Tim Conrad and Prof. Illia Horenko** for providing weekend's access to the institute building to me.
- **Arash Azhand,** together with whom I was starting the thesis work at the FU-Berlin, who became a close friend during that time, always open for critical discussions of any kind and an exemplary character of a modern physicist which I treasured each day.
- The room-mates of my shared flat: *Ekkehard Ehrenstein, Martin Delius, Thilo Schönneman, Dirk Beck, Andreas Herrmann, Lasse Kosiol, Jakob Löber* (all of them physicists), *Kai Jechorek* and short term guest, for their understanding and balance of my extensive absence during the finalizing state and a continuous coffee supply when not absent.
- **My parents**, who incessantly supported my studies and all of my choices during that time. The orthographic correctness in this thesis is largely due to the effort of my mother.
- **Dr. Kathy Lüdge** (my sister), who helped me avoiding the faux pas and following the principles of writing scientific documents.

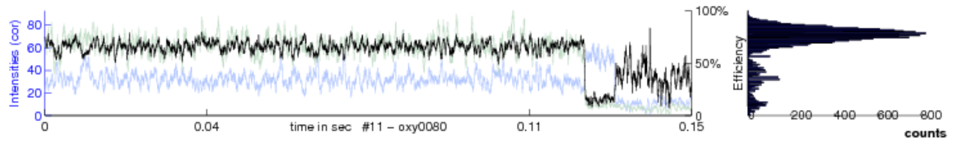
List of Figures

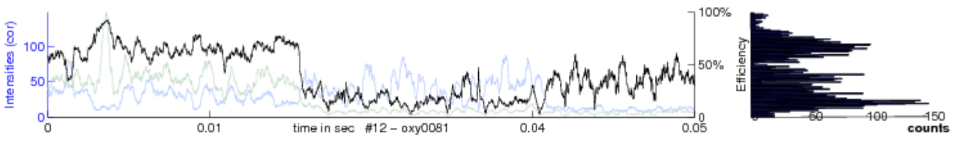
Apendix A - Data and Algorithms

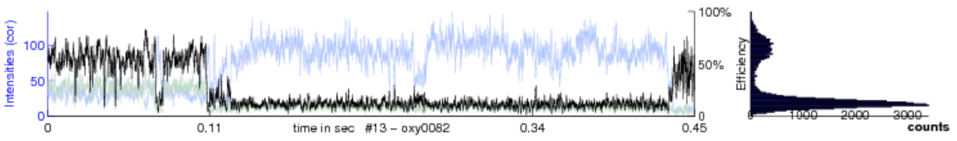
Data. The following diagrams list all trajectories that where used for analysis in this thesis. The histograms on the right represent the distribution of FRET values within that one Trajectory. The plots were generated with the *tenPlotsDAEnhanced2.m* Matlab script (see source code appendix).

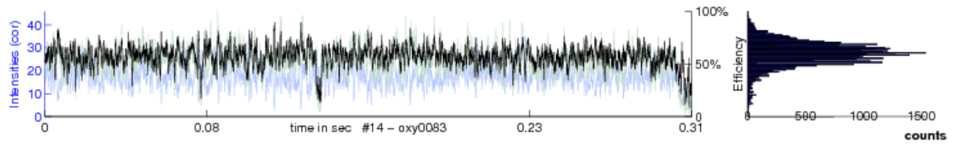
-

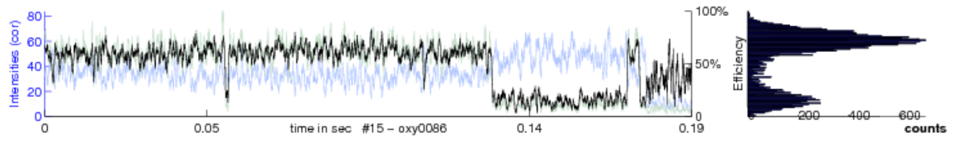

Intensities (cor)
Efficiency
counts
time in sec #1 – oxy0001
time in sec #2 – oxy0003
time in sec #3 – oxy0009
time in sec #4 – oxy0013
time in sec #5 – oxy0018
time in sec #6 – oxy0030
time in sec #7 – oxy0074
time in sec #8 – oxy0075
time in sec #9 – oxy0076
time in sec #10 – oxy0077
100%
50%

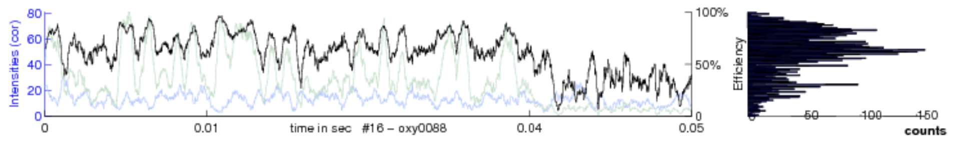

Intensities (cor)
time in sec #11 – oxy0080
Efficiency
counts

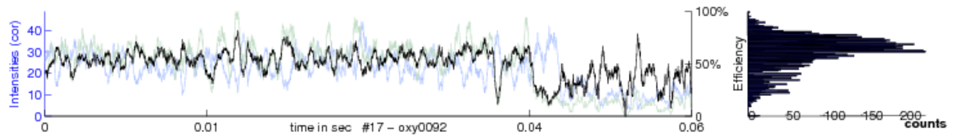

Intensities (cor)
time in sec #12 – oxy0081
Efficiency
counts

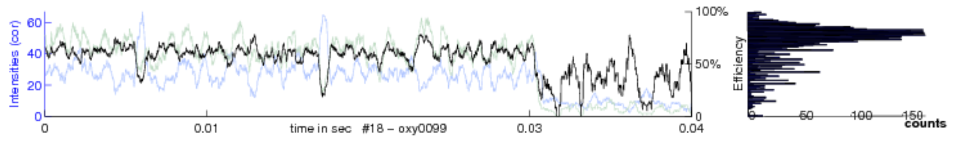

Intensities (cor)
time in sec #13 – oxy0082
Efficiency
counts

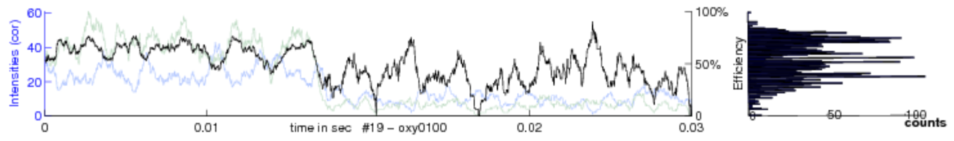

Intensities (cor)
time in sec #14 – oxy0083
Efficiency
counts

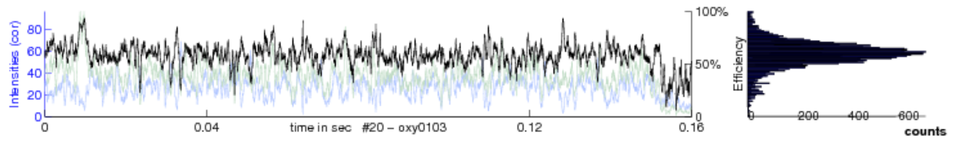

Intensities (cor)
time in sec #15 – oxy0086
Efficiency
counts

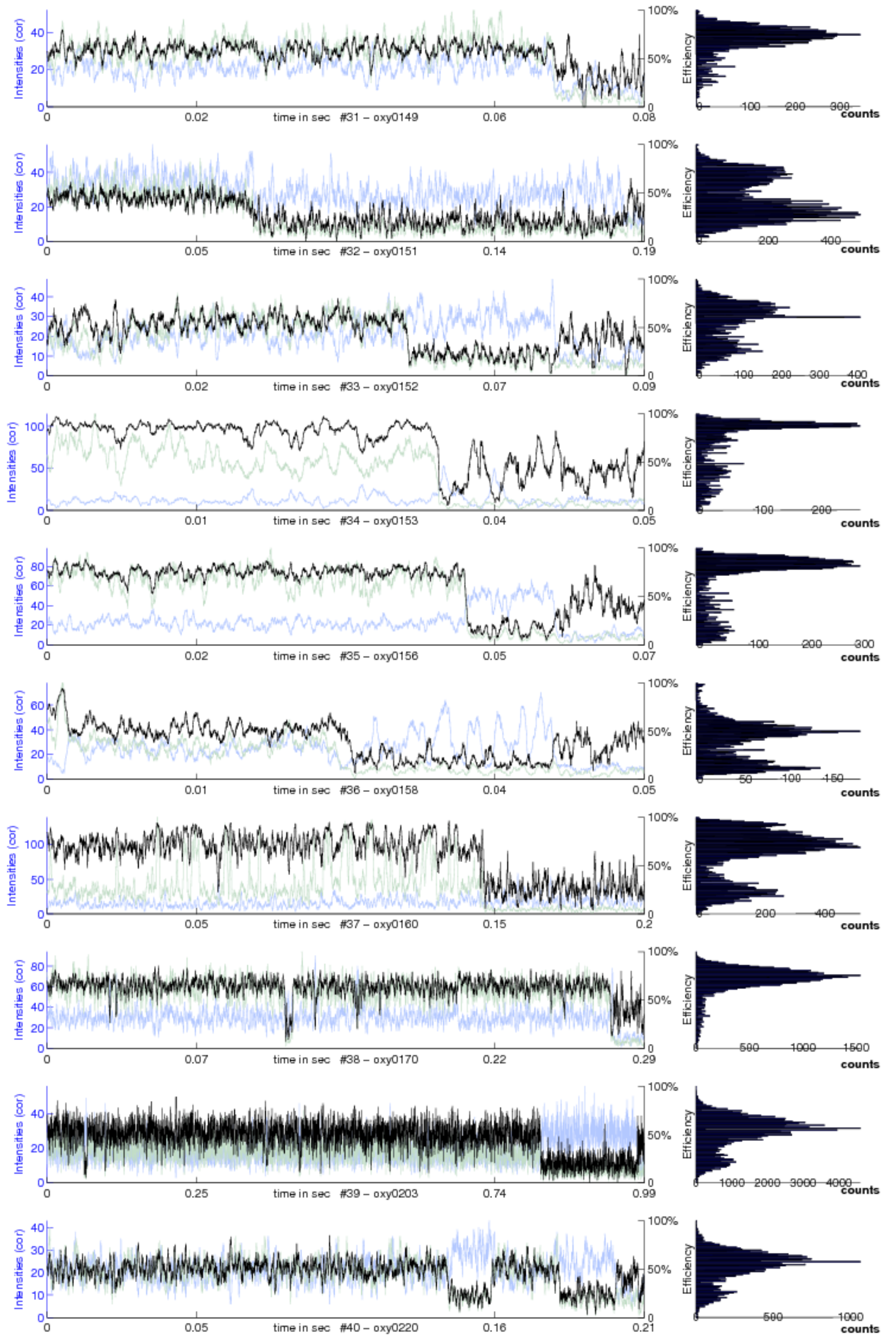

Intensities (cor)
time in sec #16 – oxy0088
Efficiency
counts

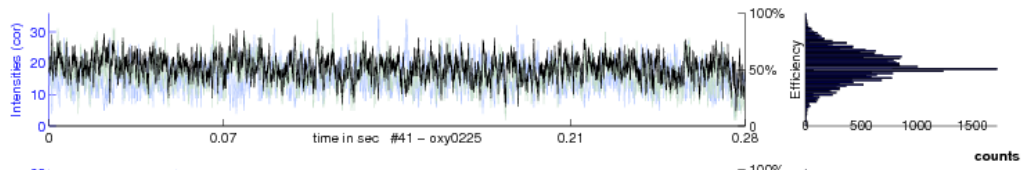

Intensities (cor)
time in sec #17 – oxy0092
Efficiency
counts

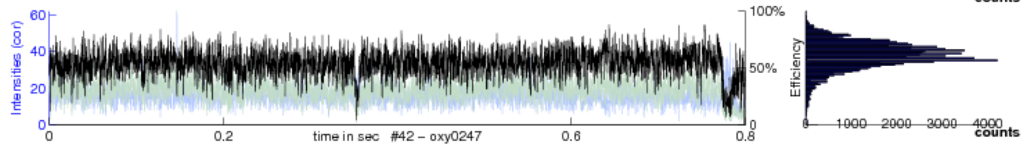

Intensities (cor)
time in sec #18 – oxy0099
Efficiency
counts

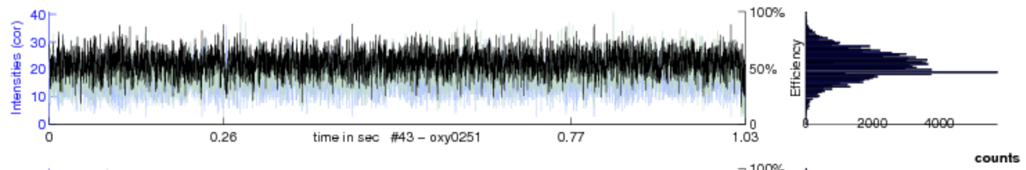

Intensities (cor)
time in sec #19 – oxy0100
Efficiency
counts

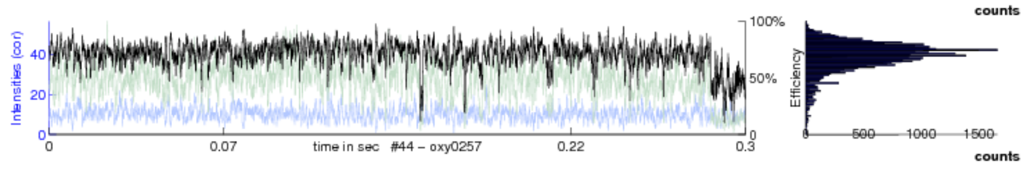

Intensities (cor)
time in sec #20 – oxy0103
Efficiency
counts

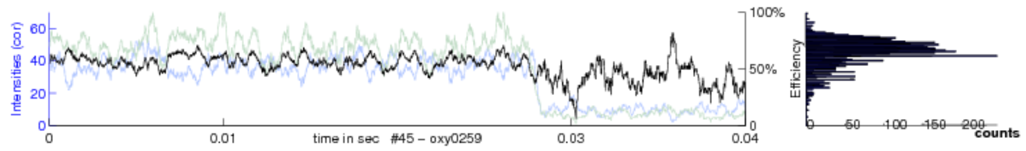

Intensities (cor)
Efficiency
time in sec #31 – oxy0149
time in sec #32 – oxy0151
time in sec #33 – oxy0152
time in sec #34 – oxy0153
time in sec #35 – oxy0156
time in sec #36 – oxy0158
time in sec #37 – oxy0160
time in sec #38 – oxy0170
time in sec #39 – oxy0203
time in sec #40 – oxy0220
counts
100%
50%

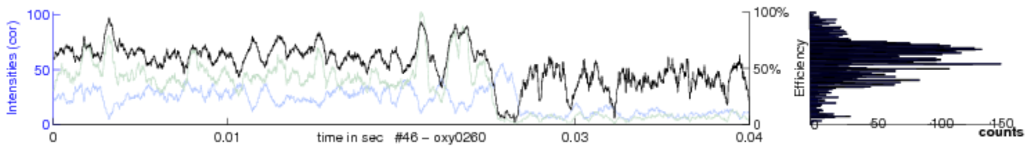

Intensities (cor)
time in sec #41 – oxy0225
Efficiency
counts

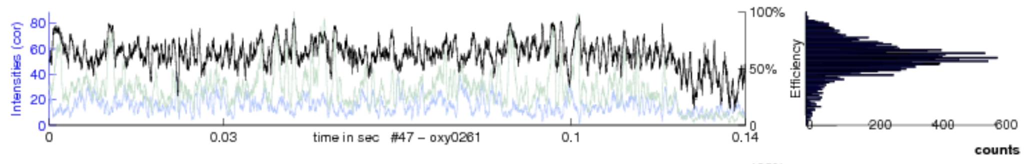

Intensities (cor)
time in sec #42 – oxy0247
Efficiency
counts

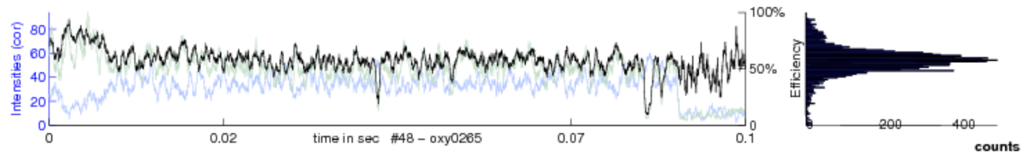

Intensities (cor)
time in sec #43 – oxy0251
Efficiency
counts

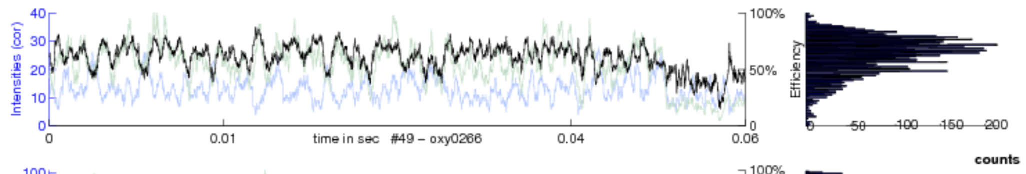

Intensities (cor)
time in sec #44 – oxy0257
Efficiency
counts

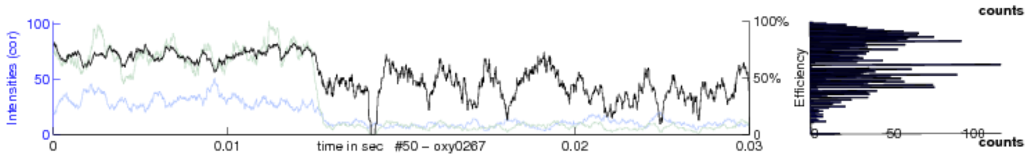

Intensities (cor)
time in sec #45 – oxy0259
Efficiency
counts

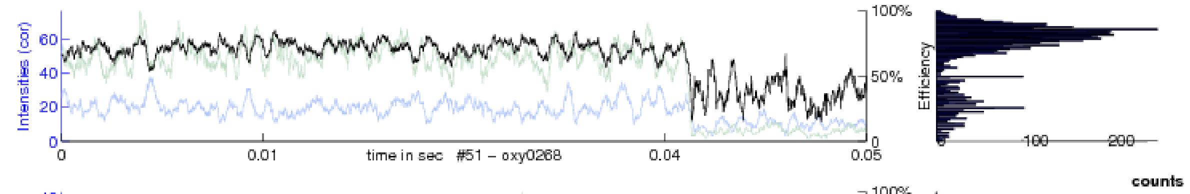

Intensities (cor)
time in sec #46 – oxy0260
Efficiency
counts

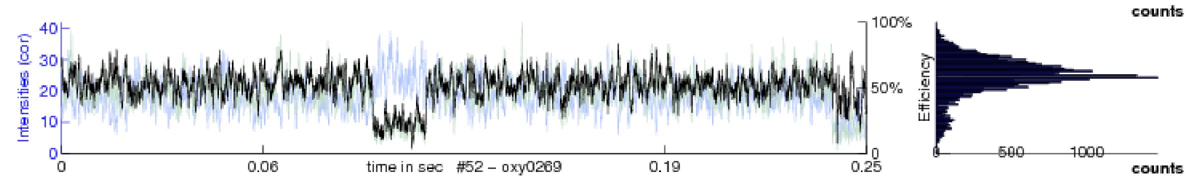

Intensities (cor)
time in sec #47 – oxy0261
Efficiency
counts

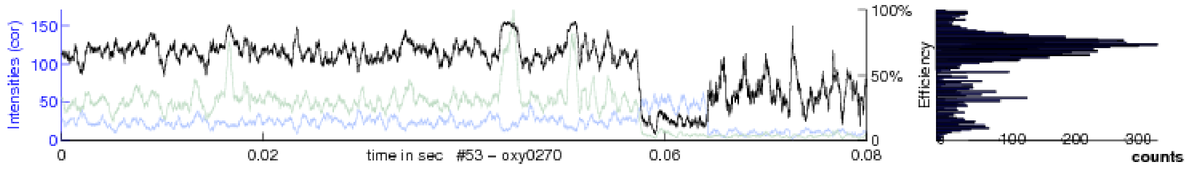

Intensities (cor)
time in sec #48 – oxy0265
Efficiency
counts

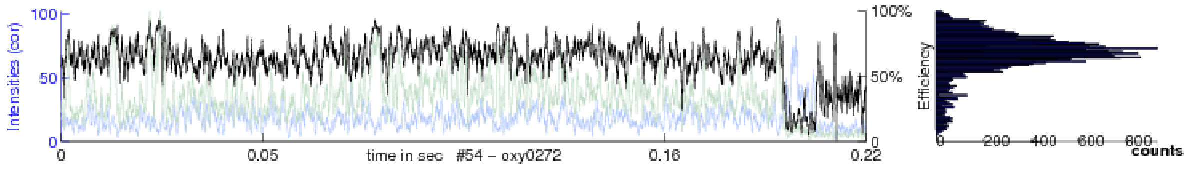

Intensities (cor)
time in sec #49 – oxy0266
Efficiency
counts

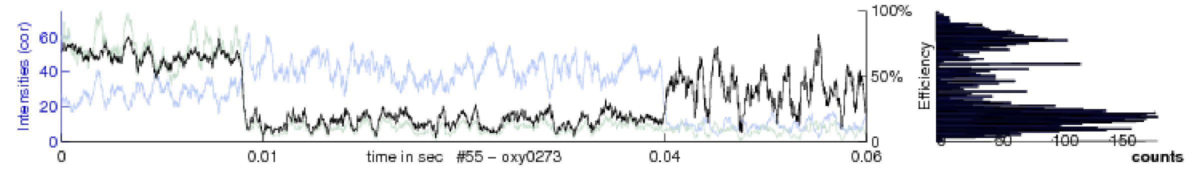

Intensities (cor)
time in sec #50 – oxy0267
Efficiency
counts

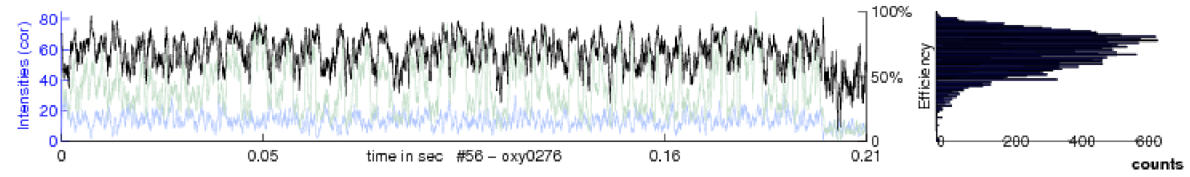

Intensities (cor)
0
20
40
60
0
0.01
time in sec #51 – oxy0268
0.04
0.05
100%
50%
0
Efficiency
0
100
200
counts

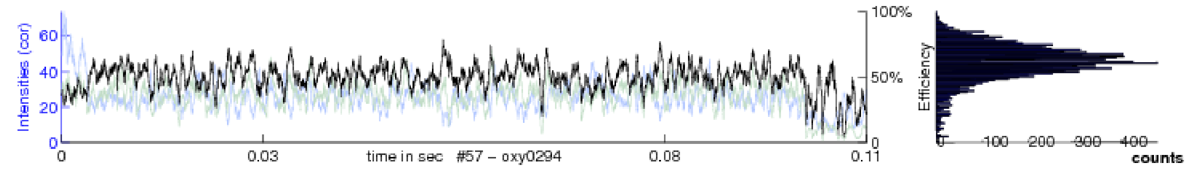

Intensities (cor)
0
10
20
30
40
0
0.06
time in sec #52 – oxy0269
0.19
0.25
100%
50%
0
Efficiency
0
500
1000
counts

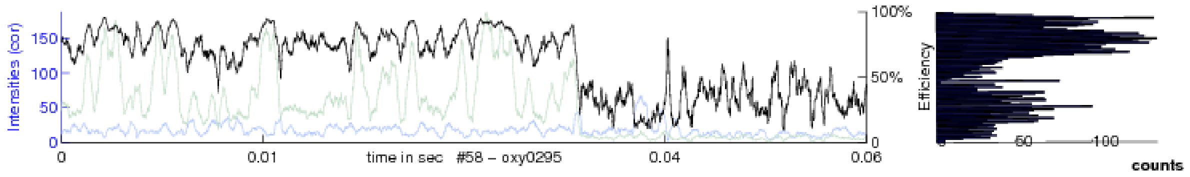

Intensities (cor)
0
50
100
150
0
0.02
time in sec #53 – oxy0270
0.06
0.08
100%
50%
0
Efficiency
0
100
200
300
counts

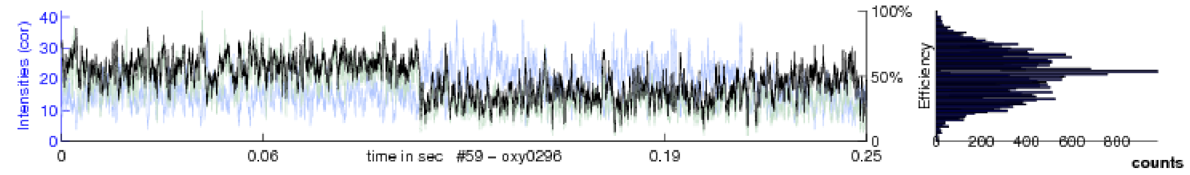

Intensities (cor)
0
50
100
0
0.05
time in sec #54 – oxy0272
0.16
0.22
100%
50%
0
Efficiency
0
200
400
600
800
counts

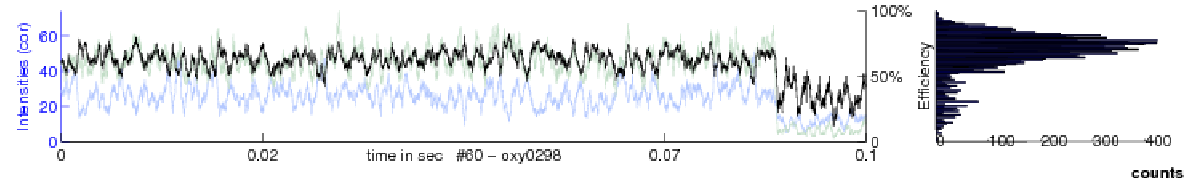

Intensities (cor)
0
20
40
60
0
0.01
time in sec #55 – oxy0273
0.04
0.06
100%
50%
0
Efficiency
0
50
100
150
counts

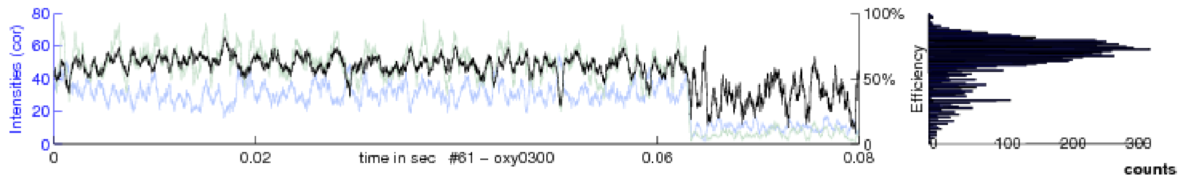

Intensities (cor)
0
20
40
60
80
0
0.05
time in sec #56 – oxy0276
0.16
0.21
100%
50%
0
Efficiency
0
200
400
600
counts

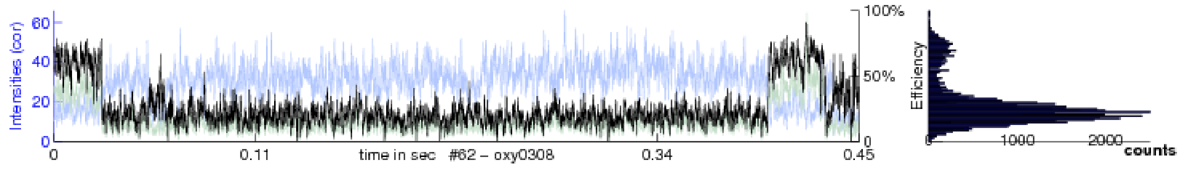

Intensities (cor)
0
20
40
60
0
0.03
time in sec #57 – oxy0294
0.08
0.11
100%
50%
0
Efficiency
0
100
200
300
400
counts

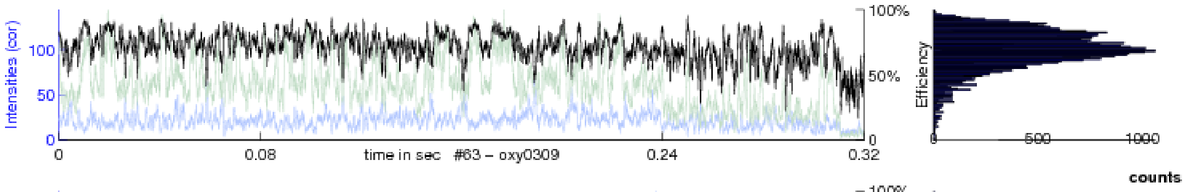

Intensities (cor)
0
50
100
150
0
0.01
time in sec #58 – oxy0295
0.04
0.06
100%
50%
0
Efficiency
0
50
100
counts

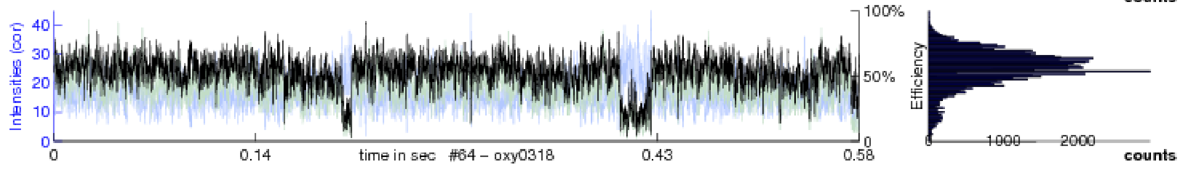

Intensities (cor)
0
10
20
30
40
0
0.06
time in sec #59 – oxy0296
0.19
0.25
100%
50%
0
Efficiency
0
200
400
600
800
counts

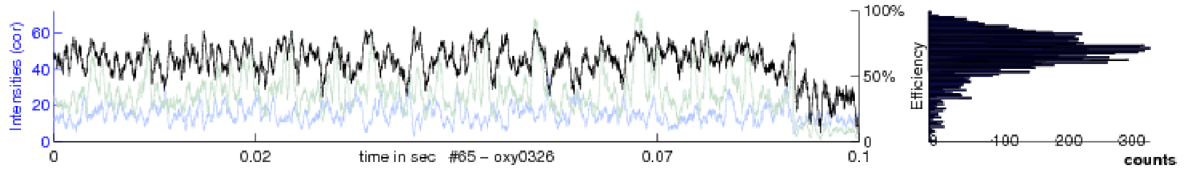

Intensities (cor)
0
20
40
60
0
0.02
time in sec #60 – oxy0298
0.07
0.1
100%
50%
0
Efficiency
0
100
200
300
400
counts

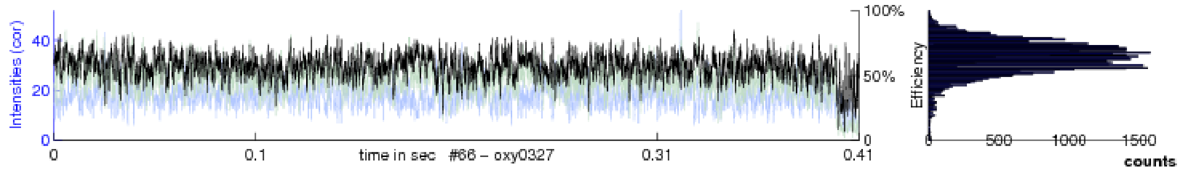

Intensities (cor)
time in sec #61 – oxy0300
Efficiency
counts

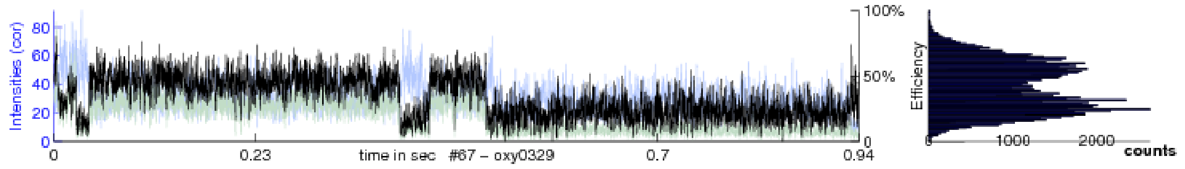

Intensities (cor)
time in sec #62 – oxy0308
Efficiency
counts

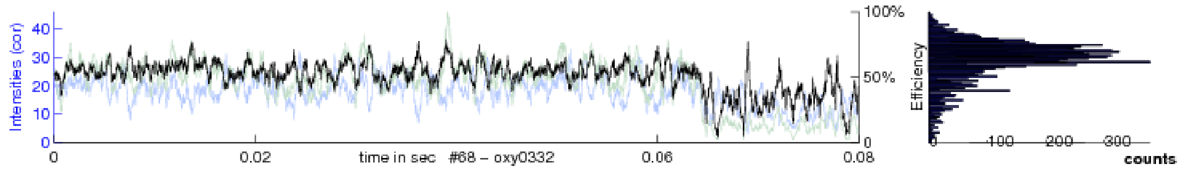

Intensities (cor)
time in sec #63 – oxy0309
Efficiency
counts

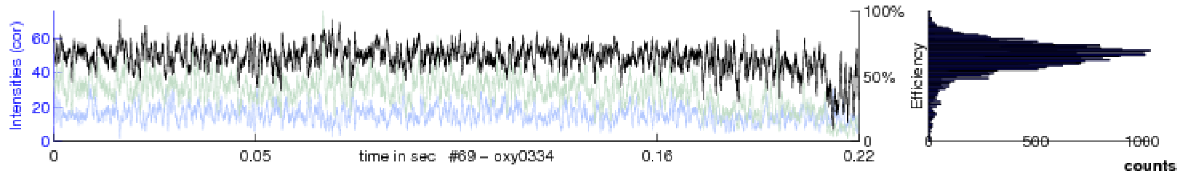

Intensities (cor)
time in sec #64 – oxy0318
Efficiency
counts

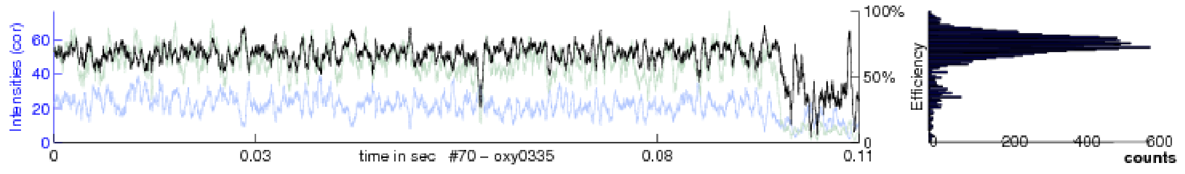

Intensities (cor)
time in sec #65 – oxy0326
Efficiency
counts

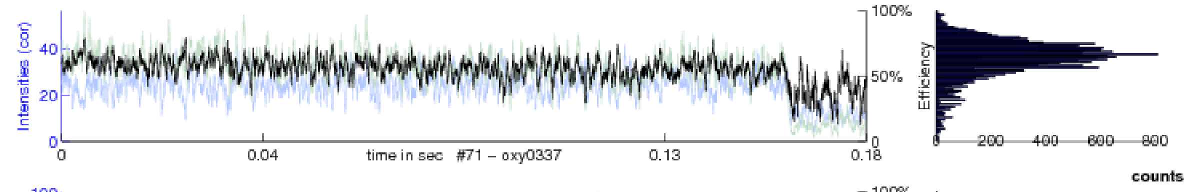

Intensities (cor)
time in sec #66 – oxy0327
Efficiency
counts

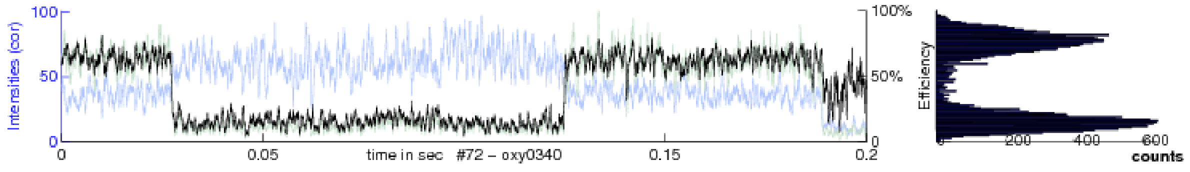

Intensities (cor)
time in sec #67 – oxy0329
Efficiency
counts

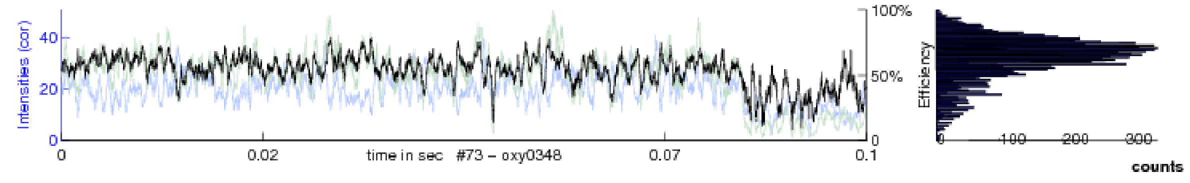

Intensities (cor)
time in sec #68 – oxy0332
Efficiency
counts

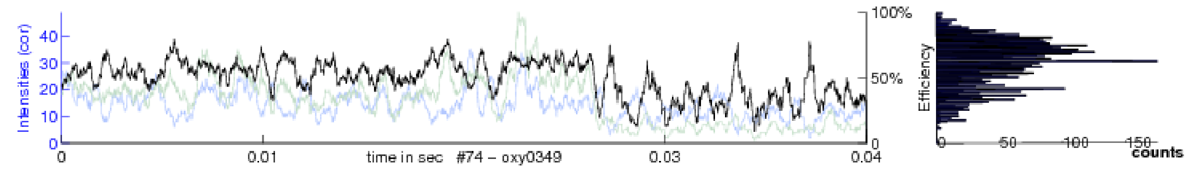

Intensities (cor)
time in sec #69 – oxy0334
Efficiency
counts

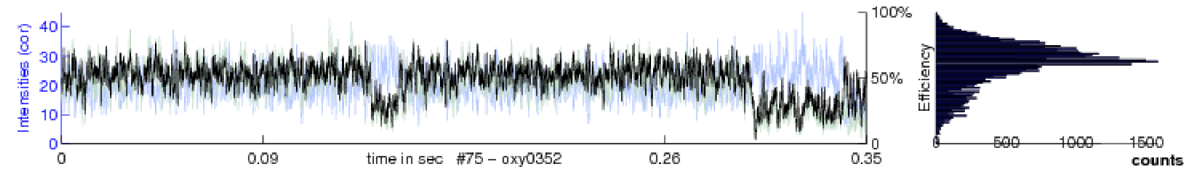

Intensities (cor)
time in sec #70 – oxy0335
Efficiency
counts

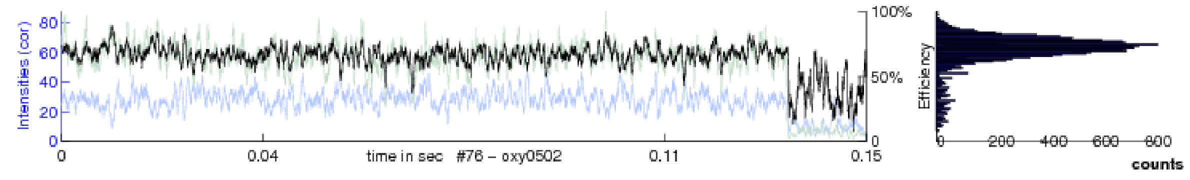

Intensities (cor)
time in sec #71 – oxy0337
Efficiency
counts

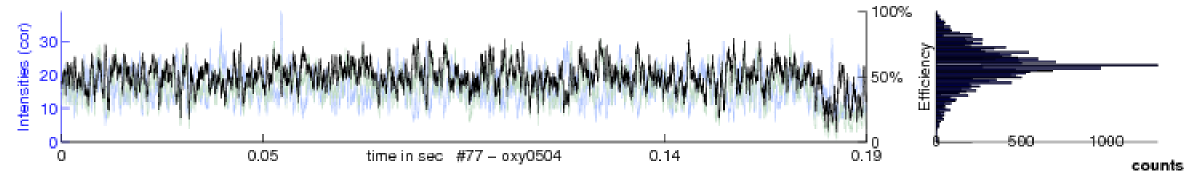

Intensities (cor)
time in sec #72 – oxy0340
Efficiency
counts

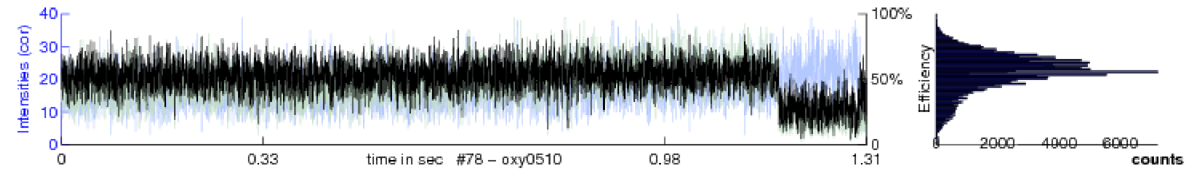

Intensities (cor)
time in sec #73 – oxy0348
Efficiency
counts

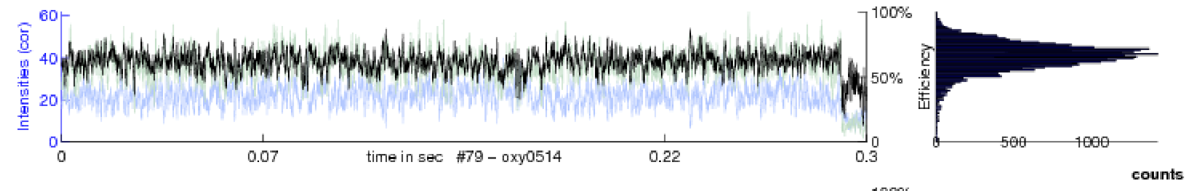

Intensities (cor)
time in sec #74 – oxy0349
Efficiency
counts

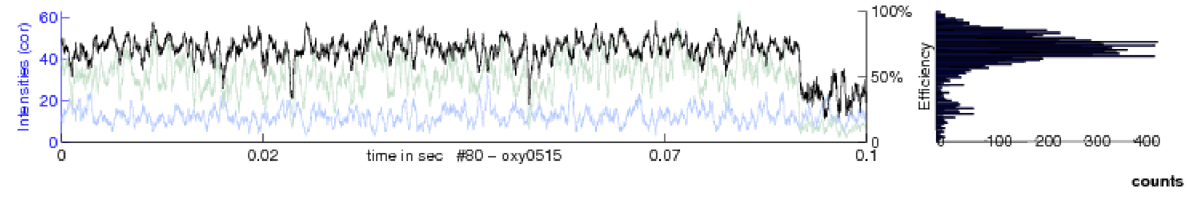

Intensities (cor)
time in sec #75 – oxy0352
Efficiency
counts

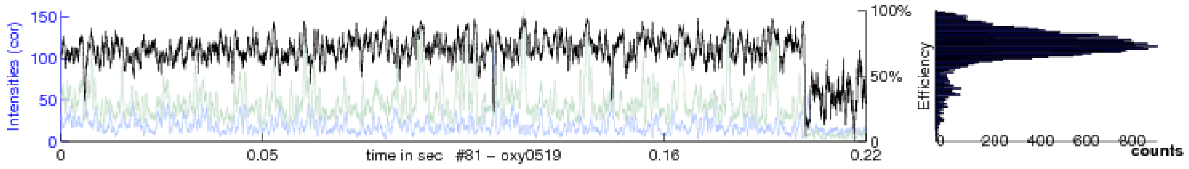

Intensities (cor)
time in sec #76 – oxy0502
Efficiency
counts

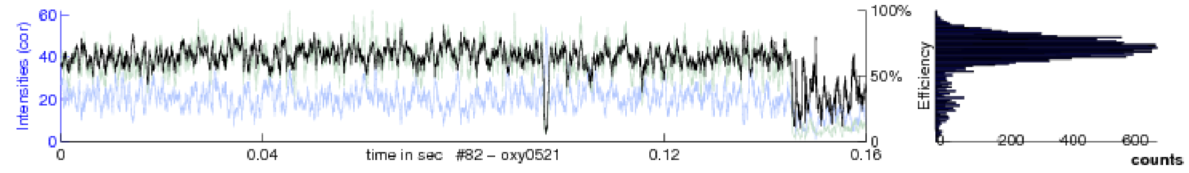

Intensities (cor)
time in sec #77 – oxy0504
Efficiency
counts

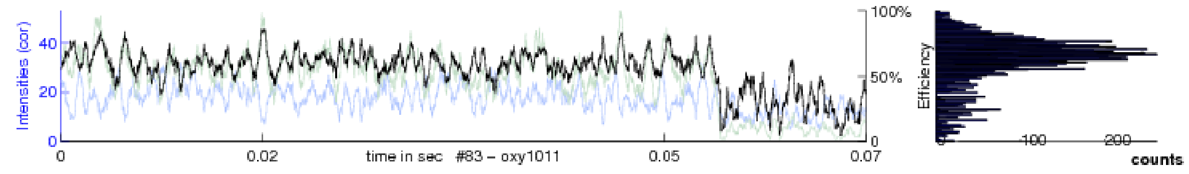

Intensities (cor)
time in sec #78 – oxy0510
Efficiency
counts

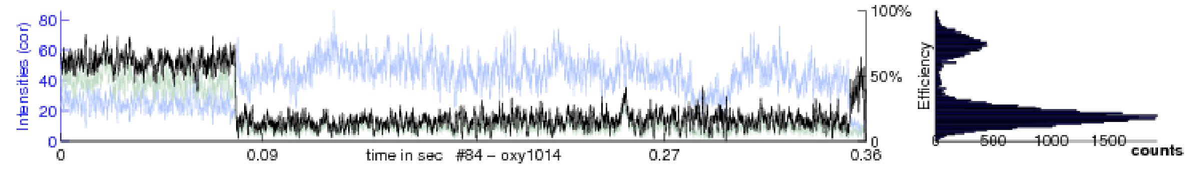

Intensities (cor)
time in sec #79 – oxy0514
Efficiency
counts

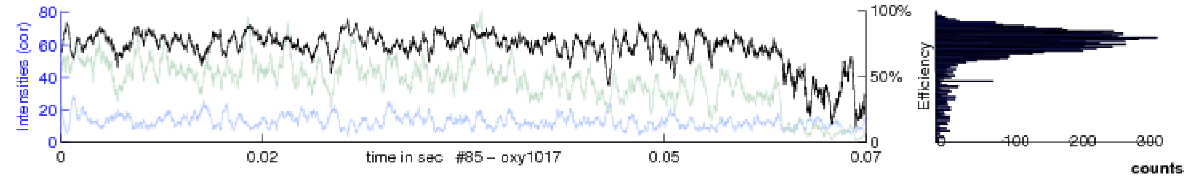

Intensities (cor)
time in sec #80 – oxy0515
Efficiency
counts

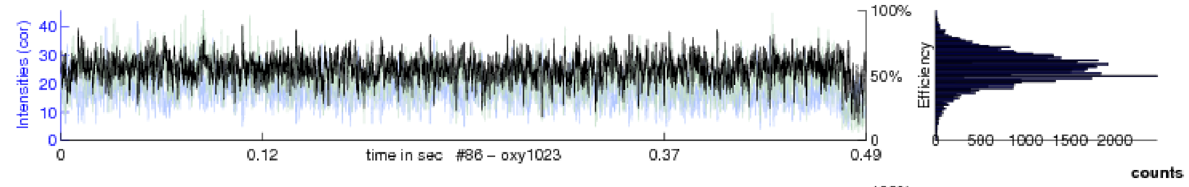

Intensities (cor)
150
100
50
0
0
0.05
time in sec #81 – oxy0519
0.16
0.22
100%
50%
0
Efficiency
0
200
400
600
800
counts

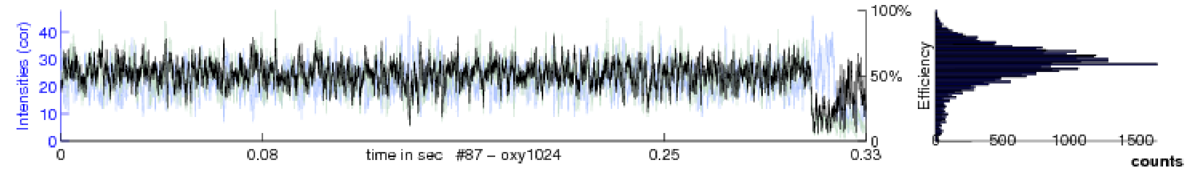

Intensities (cor)
60
40
20
0
0
0.04
time in sec #82 – oxy0521
0.12
0.16
100%
50%
0
Efficiency
0
200
400
600
counts

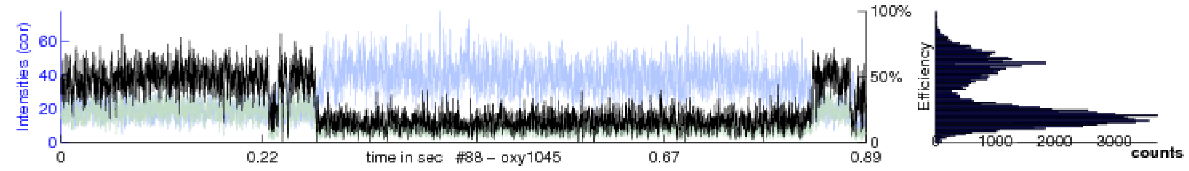

Intensities (cor)
40
20
0
0
0.02
time in sec #83 – oxy1011
0.05
0.07
100%
50%
0
Efficiency
0
100
200
counts

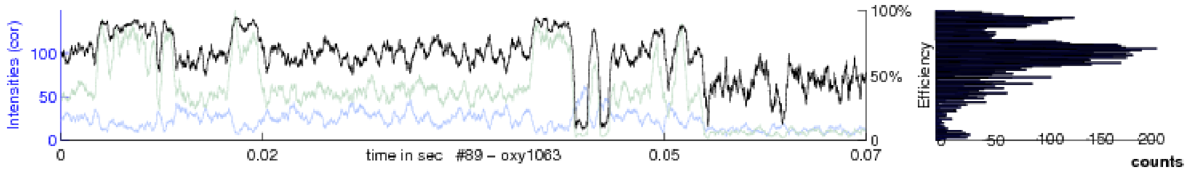

Intensities (cor)
80
60
40
20
0
0
0.09
time in sec #84 – oxy1014
0.27
0.36
100%
50%
0
Efficiency
0
500
1000
1500
counts

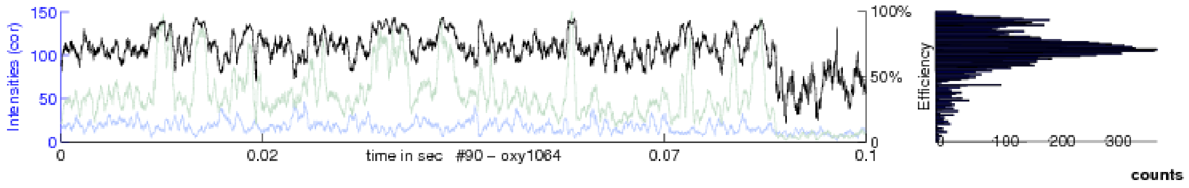

Intensities (cor)
80
60
40
20
0
0
0.02
time in sec #85 – oxy1017
0.05
0.07
100%
50%
0
Efficiency
0
100
200
300
counts

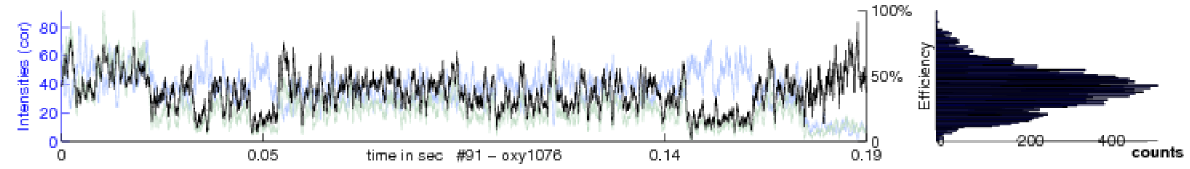

Intensities (cor)
40
30
20
10
0
0
0.12
time in sec #86 – oxy1023
0.37
0.49
100%
50%
0
Efficiency
0
500
1000
1500
2000
counts

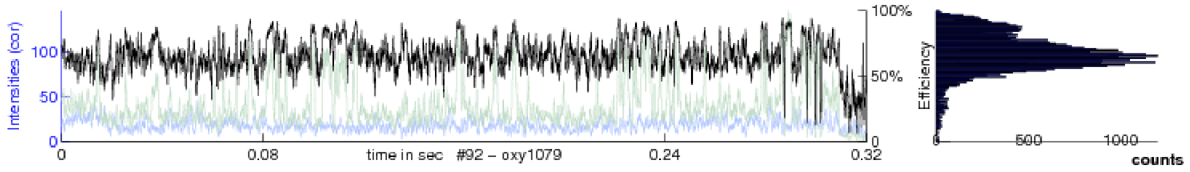

Intensities (cor)
40
30
20
10
0
0
0.08
time in sec #87 – oxy1024
0.25
0.33
100%
50%
0
Efficiency
0
500
1000
1500
counts

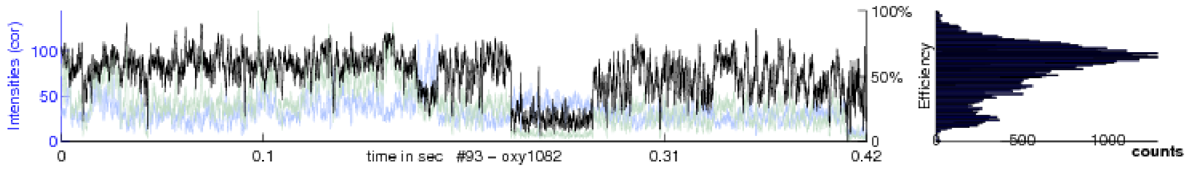

Intensities (cor)
60
40
20
0
0
0.22
time in sec #88 – oxy1045
0.67
0.89
100%
50%
0
Efficiency
0
1000
2000
3000
counts

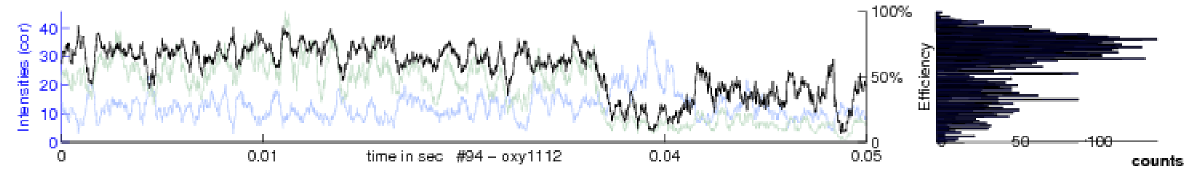

Intensities (cor)
100
50
0
0
0.02
time in sec #89 – oxy1063
0.05
0.07
100%
50%
0
Efficiency
0
50
100
150
200
counts

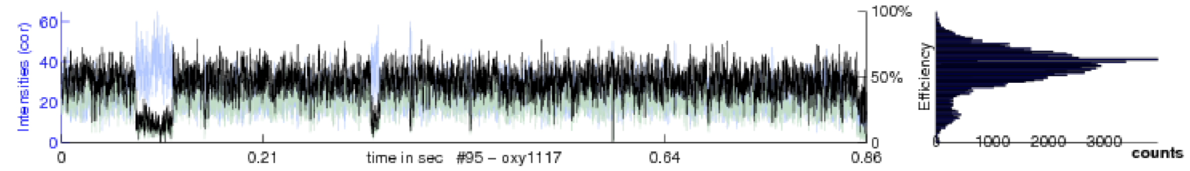

Intensities (cor)
150
100
50
0
0
0.02
time in sec #90 – oxy1064
0.07
0.1
100%
50%
0
Efficiency
0
100
200
300
counts

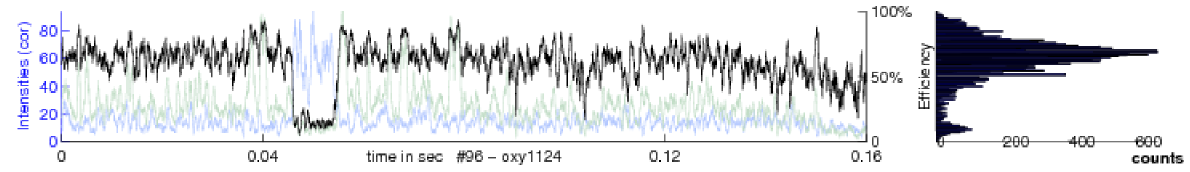

Intensities (cor)
0.05
time in sec #91 – oxy1076
0.14
0.19
100%
50%
Efficiency
200
400
counts

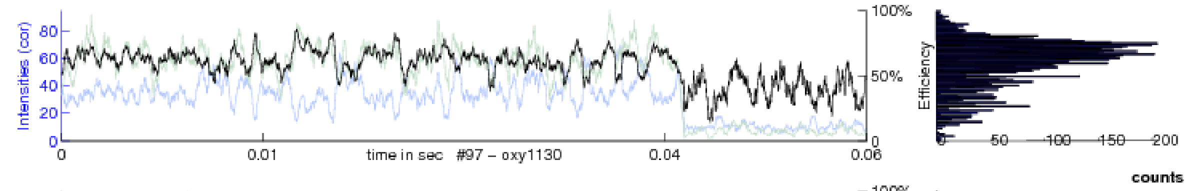

Intensities (cor)
0.08
time in sec #92 – oxy1079
0.24
0.32
100%
50%
Efficiency
500
1000
counts

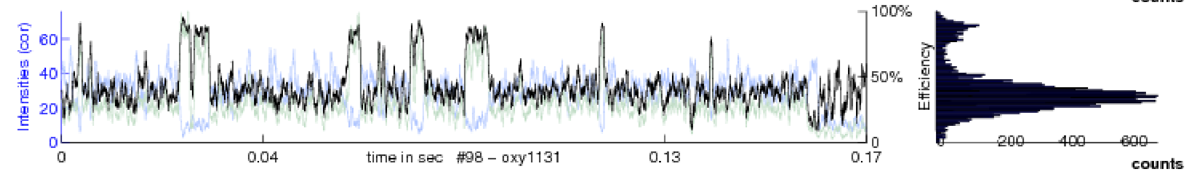

Intensities (cor)
0.1
time in sec #93 – oxy1082
0.31
0.42
100%
50%
Efficiency
500
1000
counts

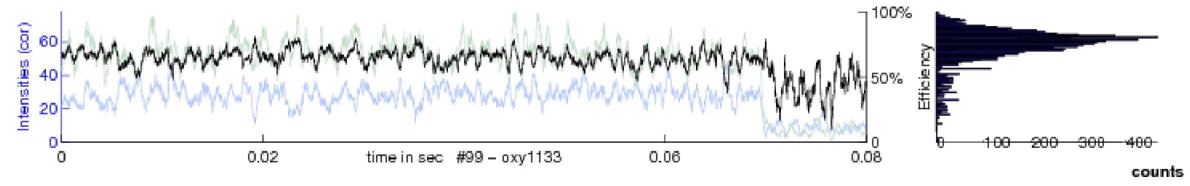

Intensities (cor)
0.01
time in sec #94 – oxy1112
0.04
0.05
100%
50%
Efficiency
50
100
counts

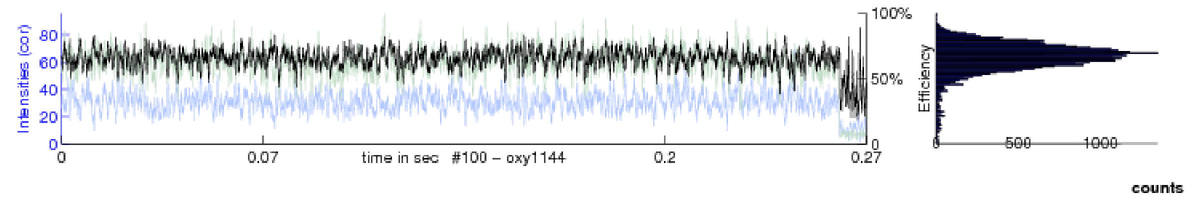

Intensities (cor)
0.21
time in sec #95 – oxy1117
0.64
0.86
100%
50%
Efficiency
1000
2000
3000
counts

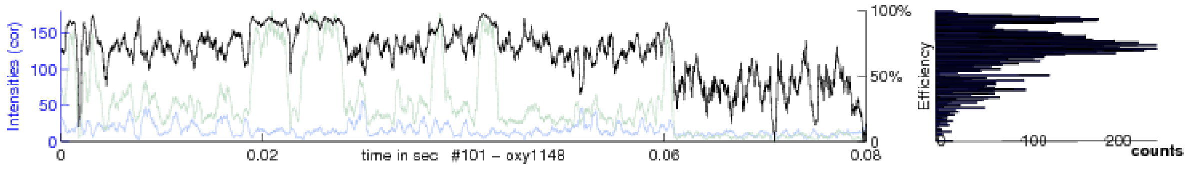

Intensities (cor)
0.04
time in sec #96 – oxy1124
0.12
0.16
100%
50%
Efficiency
200
400
600
counts

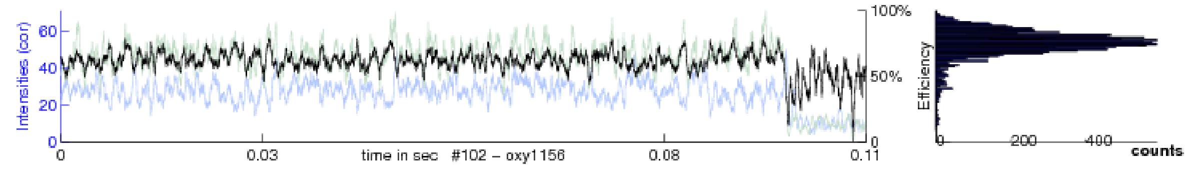

Intensities (cor)
0.01
time in sec #97 – oxy1130
0.04
0.06
100%
50%
Efficiency
50
100
150
200
counts

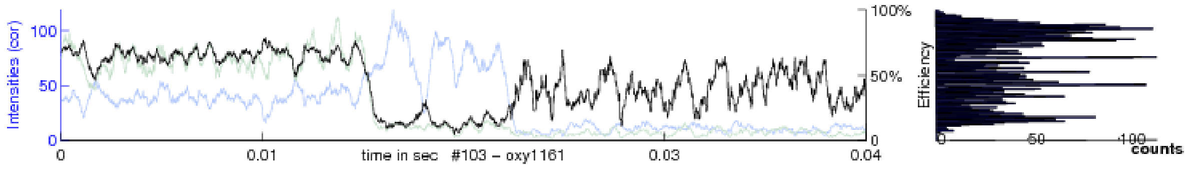

Intensities (cor)
0.04
time in sec #98 – oxy1131
0.13
0.17
100%
50%
Efficiency
200
400
600
counts

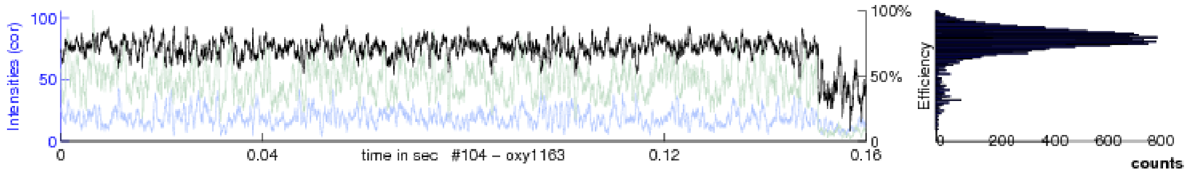

Intensities (cor)
0.02
time in sec #99 – oxy1133
0.06
0.08
100%
50%
Efficiency
100
200
300
400
counts

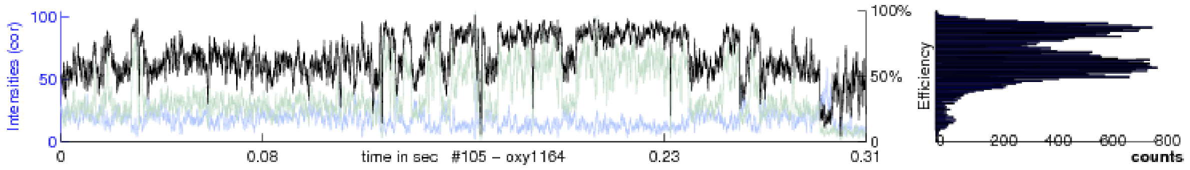

Intensities (cor)
0.07
time in sec #100 – oxy1144
0.2
0.27
100%
50%
Efficiency
500
1000
counts

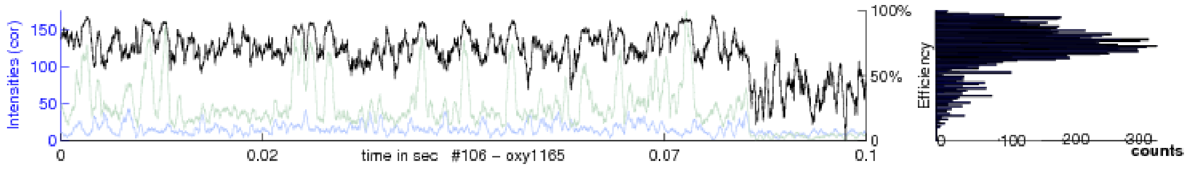

Intensities (cor)
time in sec #101 – oxy1148
Efficiency
counts

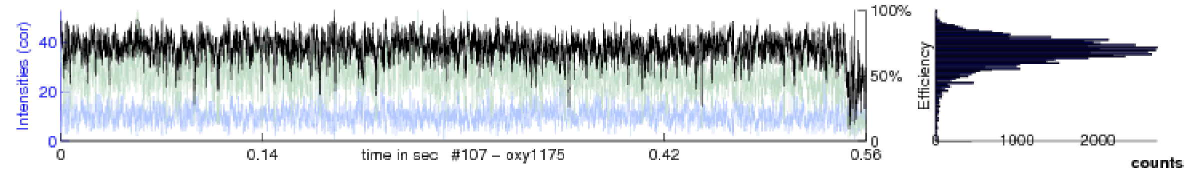

Intensities (cor)
time in sec #102 – oxy1156
Efficiency
counts

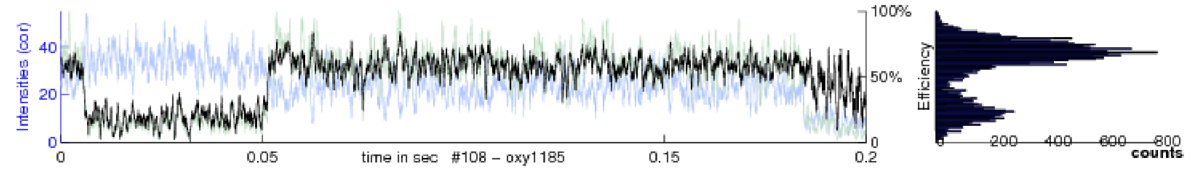

Intensities (cor)
time in sec #103 – oxy1161
Efficiency
counts

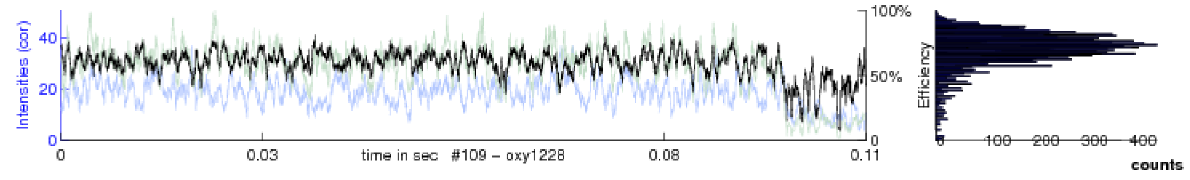

Intensities (cor)
time in sec #104 – oxy1163
Efficiency
counts

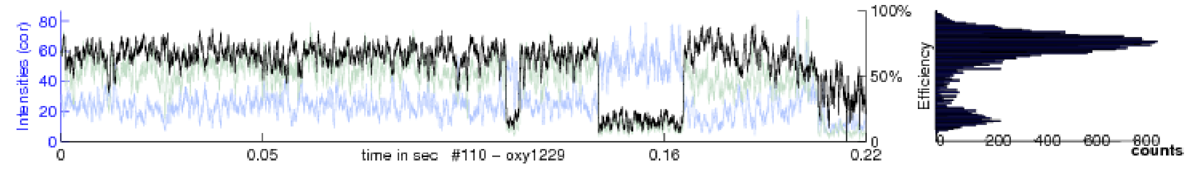

Intensities (cor)
time in sec #105 – oxy1164
Efficiency
counts

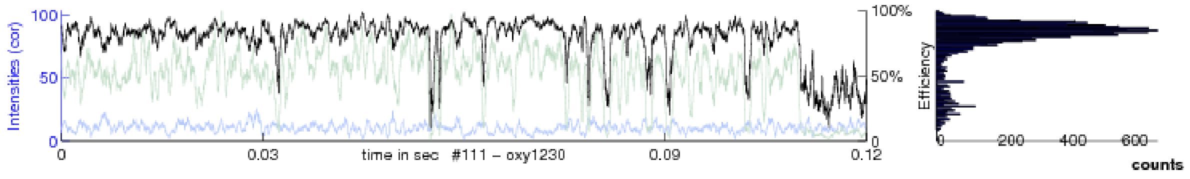

Intensities (cor)
time in sec #106 – oxy1165
Efficiency
counts

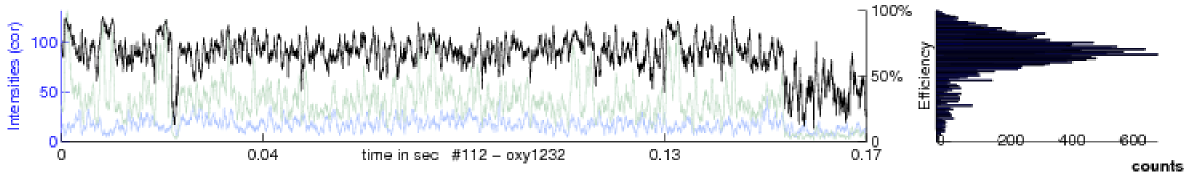

Intensities (cor)
time in sec #107 – oxy1175
Efficiency
counts

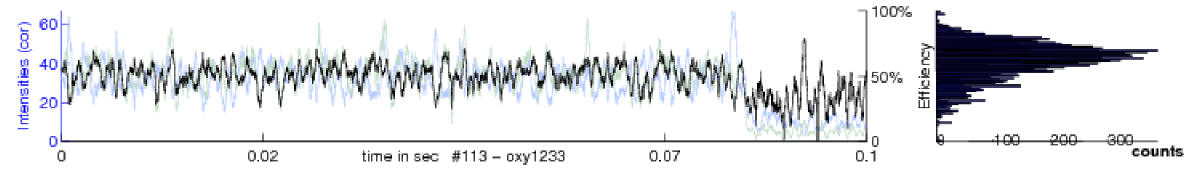

Intensities (cor)
time in sec #108 – oxy1185
Efficiency
counts

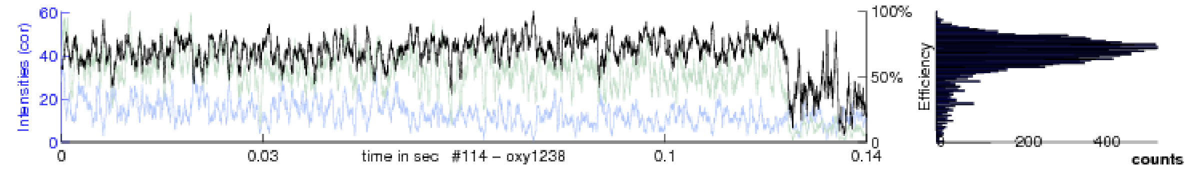

Intensities (cor)
time in sec #109 – oxy1228
Efficiency
counts

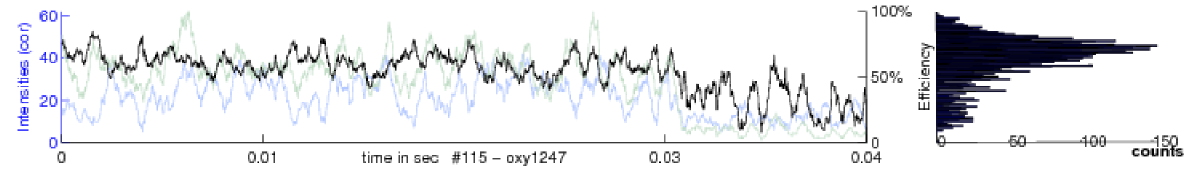

Intensities (cor)
time in sec #110 – oxy1229
Efficiency
counts

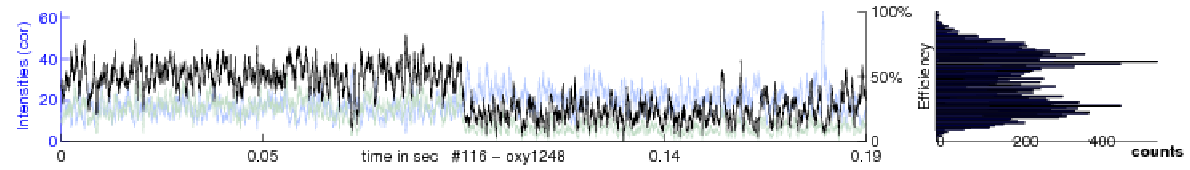

Intensities (cor)
100
50
0
0.03
time in sec #111 – oxy1230
0.09
0.12
100%
50%
Efficiency
200
400
600
counts

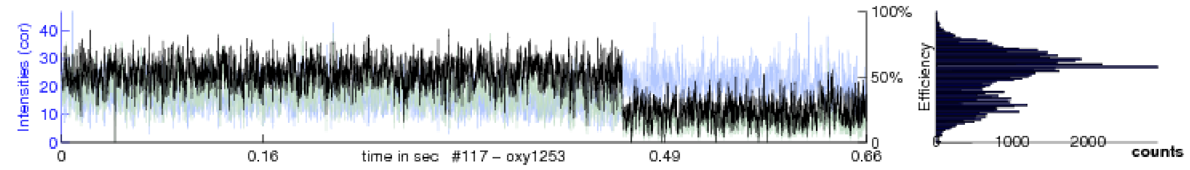

Intensities (cor)
100
50
0
0.04
time in sec #112 – oxy1232
0.13
0.17
100%
50%
Efficiency
200
400
600
counts

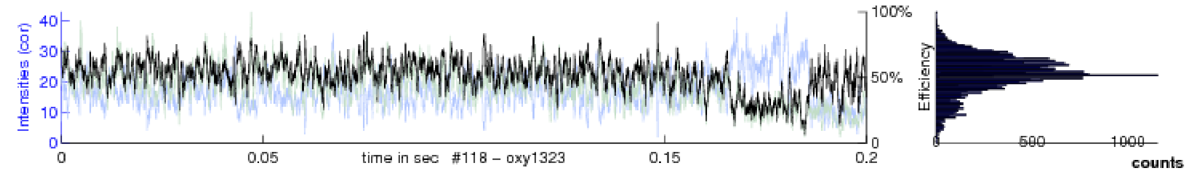

Intensities (cor)
60
40
20
0
0.02
time in sec #113 – oxy1233
0.07
0.1
100%
50%
Efficiency
100
200
300
counts

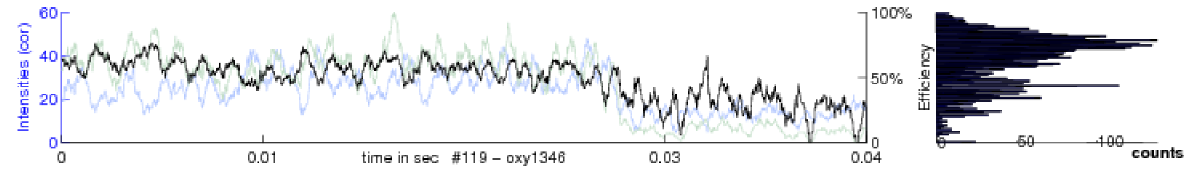

Intensities (cor)
60
40
20
0
0.03
time in sec #114 – oxy1238
0.1
0.14
100%
50%
Efficiency
200
400
counts

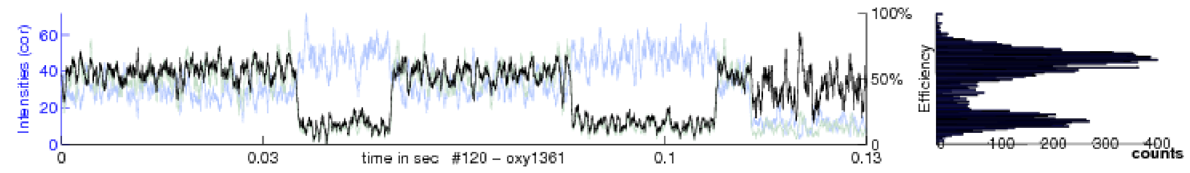

Intensities (cor)
60
40
20
0
0.01
time in sec #115 – oxy1247
0.03
0.04
100%
50%
Efficiency
50
100
150
counts


Intensities (cor)
60
40
20
0
0.05
time in sec #116 – oxy1248
0.14
0.19
100%
50%
Efficiency
200
400
counts

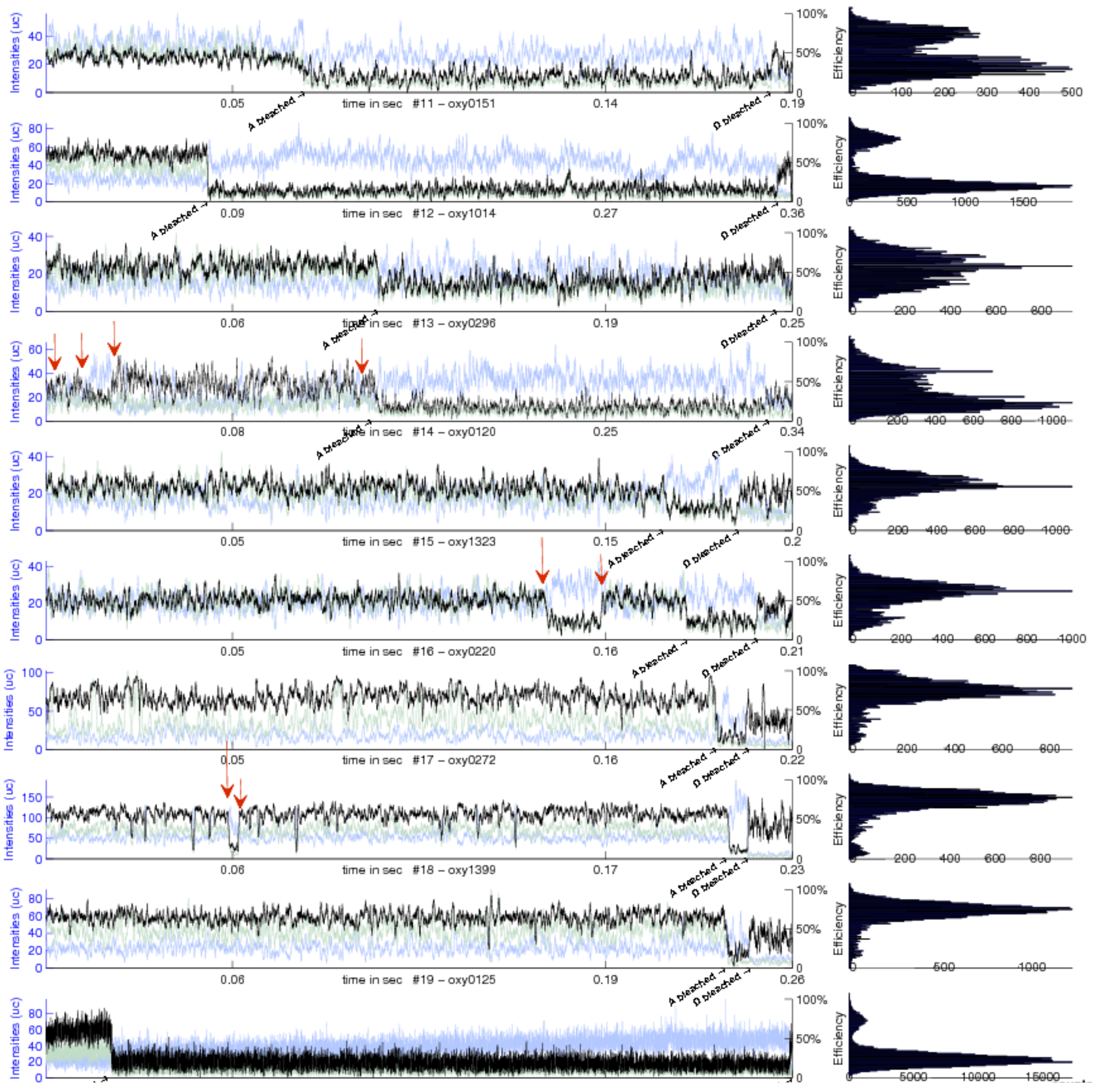

Intensities (cor)
40
30
20
10
0
0.16
time in sec #117 – oxy1253
0.49
0.66
100%
50%
Efficiency
1000
2000
counts


Intensities (cor)
40
30
20
10
0
0.05
time in sec #118 – oxy1323
0.15
0.2
100%
50%
Efficiency
500
1000
counts


Intensities (cor)
60
40
20
0
0.01
time in sec #119 – oxy1346
0.03
0.04
100%
50%
Efficiency
50
100
counts


Intensities (cor)
60
40
20
0
0.03
time in sec #120 – oxy1361
0.1
0.13
100%
50%
Efficiency
100
200
300
400
counts

FRET trajectories with appropriate bleaching phases and obvious state changes marked with red arrows.



/home/cocktail/luedge/DiplomarbeitTorsten/Plots/TrajectorySummary/co

Algorithms

Baum-Welch Algorithm. Also known as the *Forward-Backward algorithm* this generalized EM (Expectation Maximization) version features a guaranteed convergence. The algorithm can reveal the local maximum of the maximum likelihood estimator for hidden Markov models through iteration along the time-series alternating between the following two steps (responsible for the name EM):

Expectation: Evaluate the expectation value for the hidden sequence $\mathcal{S}$ given the parameter set λ_i

Maximization: Refine the previous parameters λ_i towards λ_{i+1} by maximization of

$$\lambda_{i+1} = \arg\max_{\lambda} \mathcal{S}\left(\lambda, \lambda_i\right)$$

The algorithm guarantees that $\mathcal{L}(\lambda_{i+1}) \geq \mathcal{L}(\lambda_i)$.

Applied to a Hidden Markov model this algorithm is named after Leonard E. Baum and Lloyd R. Welch. It utilizes the forward algorithm which computes the probability of ending up in a state i at time t with the partial sequence $d_1, \ldots, d_t$ being emitted is given by:

$$\alpha_i(t) = P(\mathcal{D}_1 = d_1, \ldots, d_t, \mathcal{S}(t) = s_i|\lambda).$$

With the stochastic vector $\pi = (\pi_1, ..., \pi_n)$ as initial state distribution one can define $\alpha_i(t)$ recursively as:

$\alpha_i(1) = \pi \mathbf{P}_i(d_1)$

$\alpha_j(t+1) = \sum_{i=1}^{N} (\alpha_i(t) a_{i,j}) \mathbf{P}_j(d_{t+1})$

$P(\mathcal{D}|\lambda) = \sum_{i=1}^{N} \alpha_i(T)$. That is the probability of the observation of the sequence $s_1, \ldots, s_T$ under given parameter λ. It is given by summation over all α_i at fixed time T.

The backward procedure is similar. The probability of finding a partial sequence $s_{t+1}, \ldots, s_T$ given that we started at state i at time t is:

$$\beta_i(t) = P(d_{t+1} = d_{t+1}, \ldots, d_T = d_T|\mathcal{S}(t) = s_i, \lambda).$$

Accordingly the variable $\beta_i(t)$ is called the backward variable and defined analogue $\beta_i(t)$:

$\beta_i(T) = 1$

$\beta_i(t) = \sum_{j=1}^{N} a_{i,j} \mathbf{P}_j(d_{t+1}) \beta_j(t+1)$

$P(\mathcal{D}|\lambda) = \sum_{i=1}^{N} \beta_i(1) \pi_i \mathbf{P}_i(d_1)$

We proceed and compute the probability $\gamma_i(t)$ of being in state s_i at time t for the state sequence $\mathcal{S}$ by

$$\gamma_i(t) = P(\mathcal{S}(t) = s_i|\mathcal{D}, \lambda)$$

expressed with the use of the forward- and backward-variables we can write

$$P(\mathcal{S}(t) = i|\mathcal{D}, \lambda) = \frac{P(\mathcal{D}, \mathcal{S}(t) = s_i|\lambda)}{P(\mathcal{D}|\lambda)} = \frac{P(\mathcal{D}, \mathcal{S}(t) = s_i|\lambda)}{\sum_{j=1}^{N} P(\mathcal{S}, \mathcal{S}(t) = s_j|\lambda)}.$$

Furthermore because of the conditional independence it is:

$$\alpha_i(t)\beta_i(t) = P(d_1, \ldots, d_t, \mathcal{S}(t) = s_i|\lambda)P(d_{t+1}, \ldots, d_T|\mathcal{S}(t) = s_i, \lambda) = P(\mathcal{D}, \mathcal{S}(t) = s_i|\lambda)$$

thus $\gamma_i(t)$ in terms of $\alpha_i(t)$ and $\beta_i(t)$ reads:

$$\gamma_i(t) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)}$$

The probability of being in state i at time t and being in state j at time $t+1$ is given as

$$\xi_{i,j}(t) = \frac{P(\mathcal{S}(t) = s_i,\, \mathcal{S}(t+1) = s_j,\, \mathcal{D}|\lambda)}{P(\mathcal{D}|\lambda)} = \frac{\alpha_i(t) \cdot a_{i,j}\mathbf{P}_j(d_{t+1})\beta(t+1)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i(t) \cdot a_{i,j}\mathbf{P}_i(d_{t+1})\beta(t+1)}.$$

Through the summation over all time steps t one obtains the expected total number of transitions away from state i for $\mathcal{D}$:

$$\sum_{t=1}^{T} \gamma_i(t).$$

With the same assumptions one obtains for the transitions from state s_i to state s_j for the data $\mathcal{D}$:

$$\sum_{t=1}^{T-1} \xi_{i,j}(t).$$

For the estimation of HMM we obtain for the relative frequency spent in state s_i at time 1:

$$\pi_i = \gamma_i(1).$$

The quantity a_{ij},which is an entry of the transition matrix, is the expected number of transition from state i to state j relative to the expected total number of transition away from state i.

$$a_{i,j} = \frac{\sum_{t=1}^{T-1} \xi_{i,j}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)}$$

ad in the discrete case

$$b_i(k) = \frac{\sum_{t=1}^{T-1} \delta_{d_t,,v_k}\gamma_i(t)}{\sum_{t=1}^{T-1} \gamma_i(t)}$$

is the expected number of times the output observations have been equal to $v_k$while in state i relative to the expected total number of times in state i.

Apendix B - Java Source-Code

The following listing lists all employed Java programs. Their task is mainly to prepare the data for evaluation as well as to start the iterating process, initialize the model and read out the results of the HMMVAR estimation.

```java
import java.util.Random;

import biocomp.moltools.util.Arguments;
import biocomp.moltools.util.FileTools;
import biocomp.moltools.util.IntArrays;

public class RandomSequenceGenerator {

        private int sequenceLength;

        private int state;
        private boolean useRandomSeed;

        /** Transition Matrix */
        private double[][] transitionMatrix;
        private double[][] cumulatedTransitionMatrix;

        /**
         * Constructor
         * @param sequenceLength is the length of the sequence
         * @param t is the transition matrix
         * @param useRandomSeed whether to initialize with random seed,
         * if false the seed 0 is used
         */
        public RandomSequenceGenerator(int sequenceLength, double[][] t,
                        boolean useRandomSeed) {
                this.sequenceLength = sequenceLength;
                this.state = 0;
                this.transitionMatrix = t;
                this.useRandomSeed = useRandomSeed;

                int dim = t.length; // assumption: quadratic matrix t

        cumulatedTransitionMatrix = new double[dim][];
        for (int row = 0; row < dim; row++) {
            cumulatedTransitionMatrix[row] = new double[dim];
        }

        this.cumulateTransactionMatrix(dim);
    }

    /**
     * Generate a new random sequence based on the transition matrix given in the constructor
     */
    public int[] generateSequence() {

        int[] sequence = new int[sequenceLength];

        Random random;
        if (useRandomSeed) {
            random = new Random();
        } else {
            random = new Random(0);
        }

        for (int j = 0; j < sequenceLength; j++) {
            double nextRandom = random.nextDouble();

            int i = 0;
            while (nextRandom > cumulatedTransitionMatrix[state][i]) {
                i++;
            }
            state = i;
            sequence[j] = state;
        }
        return sequence;
    }

    /**
     * Sums up probabilities of transition matrix.
     * One row of the matrix will have entries
     * p1 p2 p3 => p1 p1+p2 p1+p2+p3
     * @param dimensions is the dimension of the stochastic matrix
     */
    protected void cumulateTransactionMatrix(int dimensions) {
        for (int row = 0; row < dimensions; row++) {
```

```
                        double sum = 0.0;
                        for (int column = 0; column < dimensions;
                            column++) {
                                sum += transitionMatrix[row][
                                    column];
                                cumulatedTransitionMatrix[row][
                                    column] = sum;
                        }
                }
        }

        /**
         *
         * @param args
         */
        public static void main(String[] args) {
                if (args == null) {
                        System.out.println("
                            RandomSequenceGenerator " +
                                        "-t <transitionmatrix-file
                                            > " +
                                        "-ts <time steps> " +
                                        "-output [<save file>]" );
                        System.exit(1);
                } else {
                        Arguments arg = new Arguments(args);
                        int ts = arg.getIntArgument("ts");
                        String basepath = "../data/";

                        MatrixFileReader transitionMatrixReader =
                            new MatrixFileReader( basepath + arg.
                            getArgument( "t" ));
                        double[][] transitionMatrix =
                            transitionMatrixReader.getMatrix();

                        RandomSequenceGenerator timeSpaceGenerator
                             = new RandomSequenceGenerator( ts,
                            transitionMatrix, true );
                        int[] timeSpace = timeSpaceGenerator.
                            generateSequence();

                        int noOfoutputArguments = arg.
                            getNArguments("output");
                        if ( noOfoutputArguments == 1 ) {
                                String outputFilename = arg.
                                    getArgument("output", 0 ); //
                                    get first argument of output
```

```java

                                StringBuffer sb = new StringBuffer
                                     ();
                                for (int i=0; i<timeSpace.length;i
                                     ++) {
                                sb.append( timeSpace[i] + "\n" );
                        }

                        FileTools.writeString( outputFilename, sb.
                             toString() );
                        } else if (noOfoutputArguments == 0) {
                                // output to console
                                System.out.println(IntArrays.
                                     toString(timeSpace));
                        } else {
                                System.out.println("Too many
                                     arguments for output");
                        }
                }
        }
}
```

```java
import cern.colt.list.*;
public class Nprobe
{
public static double[] probe(int[] acc, int[] don, int nprobes,
     int w)
{
DoubleArrayList res = new DoubleArrayList();
double inc = 1.0 / (double)nprobes;
for (int i=0; i<acc.length-w+1; i++)
{
int nA = 0, nD = 0;
for (int j=i; j<i+w; j++)
{
int nacc = acc[j];
int ndon = don[j];
for (int k=0; k<(nacc+ndon); k++)
{
if (Math.random() < (double)nacc/((double)(nacc+ndon)))
{
nA ++;
nacc --;
}
else
{
nD ++;
ndon --;
}
if (nA+nD == nprobes)
break;
}
}
if (nA+nD == nprobes)
res.add((double)(nA) * inc);
}
res.trimToSize();
double[] ret = res.elements();
return(ret);
}
public static void main(String[] args)
{
if (args.length == 0)
{
System.out.println("IncreasingWindow <file> <maxwin>");
System.exit(0);
}
int[] acc = FileLineReader.readIntColumn(args[0],1);
int[] don = FileLineReader.readIntColumn(args[0],2);
//int[] accref = FileLineReader.readIntColumn(args[1],1);
//int[] donref = FileLineReader.readIntColumn(args[1],2);
int maxwin = StringTools.toInt(args[1]);
for (int i=2; i<10; i++)
{
double[] p = probe(acc, don, i, maxwin);
double mean = DoubleArrays.mean(p);
double var = DoubleArrays.variance(p);
double stddev = Math.sqrt(var);
double varest = 2*var-mean-mean*mean;
System.out.println(i+"\t"+p.length+"\t"+mean+"\t"+stddev+"\t"+var+
     "\t"+varest);
}
}
}
```

```java
import java.util.Random;

import biocomp.moltools.util.Arguments;
import biocomp.moltools.util.FileTools;
import biocomp.moltools.util.IntArrays;

public class RandomSequenceGenerator {

        private int sequenceLength;

        private int state;
        private boolean useRandomSeed;

        /** Transition Matrix */
        private double[][] transitionMatrix;
        private double[][] cumulatedTransitionMatrix;

        /**
         * Constructor
         * @param sequenceLength is the length of the sequence
         * @param t is the transition matrix
         * @param useRandomSeed whether to initialize with random
             seed,
         * if false the seed 0 is used
         */
        public RandomSequenceGenerator(int sequenceLength, double
            [][] t,
                        boolean useRandomSeed) {
                this.sequenceLength = sequenceLength;
                this.state = 0;
                this.transitionMatrix = t;
                this.useRandomSeed = useRandomSeed;

                int dim = t.length; // assumption: quadratic
                    matrix t

                cumulatedTransitionMatrix = new double[dim][];
                for (int row = 0; row < dim; row++) {
                        cumulatedTransitionMatrix[row] = new
                            double[dim];
                }

                this.cumulateTransactionMatrix(dim);
        }

        /**
         * Generate a new random sequence based on the transition
             matrix given in the constructor
         */
        public int[] generateSequence() {

                int[] sequence = new int[sequenceLength];

                Random random;
                if (useRandomSeed) {
                        random = new Random();
                } else {
                        random = new Random(0);
                }

                for (int j = 0; j < sequenceLength; j++) {
                        double nextRandom = random.nextDouble();

                        int i = 0;
                        while (nextRandom >
                            cumulatedTransitionMatrix[state][i]) {
                                i++;
                        }
                        state = i;
                        sequence[j] = state;
                }
                return sequence;
        }

        /**
         * Sums up probabilities of transition matrix.
         * One row of the matrix will have entries
         * p1 p2 p3 => p1 p1+p2 p1+p2+p3
         * @param dimensions is the dimension of the stochastic
             matrix
         */
        protected void cumulateTransactionMatrix(int dimensions) {
                for (int row = 0; row < dimensions; row++) {
                        double sum = 0.0;
                        for (int column = 0; column < dimensions;
                            column++) {
                                sum += transitionMatrix[row][
                                    column];
                                cumulatedTransitionMatrix[row][
                                    column] = sum;
                        }
                }
        }

        /**
         *
         * @param args
         */
        public static void main(String[] args) {
                if (args == null) {
                        System.out.println("
                            RandomSequenceGenerator " +
                                                "-t <transitionmatrix-file
                                                    > " +
                                                "-ts <time steps> " +
                                                "-output [<save file>]" );
                              System.exit(1);
                    } else {
                              Arguments arg = new Arguments(args);
                              int ts = arg.getIntArgument("ts");
                              String basepath = "../data/";

                              MatrixFileReader transitionMatrixReader =
                                   new MatrixFileReader( basepath + arg.
                                   getArgument( "t" ));
                              double[][] transitionMatrix =
                                   transitionMatrixReader.getMatrix();

                              RandomSequenceGenerator timeSpaceGenerator
                                    = new RandomSequenceGenerator( ts,
                                   transitionMatrix, true );
                              int[] timeSpace = timeSpaceGenerator.
                                   generateSequence();

                              int noOfoutputArguments = arg.
                                   getNArguments("output");
                              if ( noOfoutputArguments == 1 ) {
                                        String outputFilename = arg.
                                             getArgument("output", 0 ); //
                                             get first argument of output

                                        StringBuffer sb = new StringBuffer
                                             ();
                                        for (int i=0; i<timeSpace.length;i
                                             ++) {
                                        sb.append( timeSpace[i] + "\n" );
                              }

                              FileTools.writeString( outputFilename, sb.
                                   toString() );
                              } else if (noOfoutputArguments == 0) {
                                        // output to console
                                        System.out.println(IntArrays.
                                             toString(timeSpace));
                              } else {
                                        System.out.println("Too many
                                             arguments for output");
                              }
                    }
```

```
        }
}
```

```
import java.util.Random;
/**
 *
 * @author luedge
 */
public class GeneratorSDE {
    private double dt;
    private double[] k;
    private double[] D;
    private Random rnd = new Random();


    public GeneratorSDE(double dt, double[] k, double[] D) {
        this.dt = dt;
        this.k = k;
        this.D = D;
    }


    /**
     * Generate a SDE trajectory based on the hiddenStateSequence
         and given parametrization
     * @param hiddenStateSequence
     * @param x_0 is the inital value of the generated sequence
     * @return the sequence based on SDEs
     */
    public double[] generateSDETrajectory( int[]
        hiddenStateSequence, double x_0 ) {
        double[] sdeSequence = new double[ hiddenStateSequence.
            length ];
        sdeSequence[0] = x_0;
        for (int i = 1; i < hiddenStateSequence.length; i++ ) {
            sdeSequence[i] = generateSDEStep( sdeSequence[i-1],
                hiddenStateSequence[i-1] );
        }
        return sdeSequence;
    }

    /**
     *
     * @param x
     * @param hiddenState
     * @return
     */
```

```java
    private double generateSDEStep( double x, int hiddenState ) {
        return x + (
                            (- 2 * k[hiddenState] * x * D[hiddenState]) +
                            ( Math.sqrt( 2 * D[hiddenState] ) * rnd.nextGaussian() )
                        ) * dt;
    }
}
```

```java
import biocomp.metamacs.hmm.newImplementation.HMMVAR;
import biocomp.moltools.util.DoubleArrays;
import biocomp.moltools.util.FileLineReader;
import biocomp.moltools.util.IntArrays;

public class Estimator {

    public static void main( String[] args ) {
        // =======================
        // input params - do by arguments later
        // =======================

        String basepath = "../data/";
        String transitionMatrixFilename = basepath + "test3matrix.txt";
        String sdeParamsFilename = basepath + "testV3param.txt";
        String realDataFile = basepath + "DAse1.dat";
        int sequenceLength;
        double dt = 0.2;
        double x_0 = 0.5;
        int numberOfStates;

        // =======================
        // start of main routine
        // =======================

        // get transition matrix
        MatrixFileReader matrixFileReader = new MatrixFileReader( transitionMatrixFilename );
        double[][] transitionMatrix = matrixFileReader.getMatrix();
        numberOfStates = transitionMatrix.length;

        // get real Data from file
        FileLineReader readRealSequence = new FileLineReader( realDataFile );
        sequenceLength = ( int ) readRealSequence.getNLines();

        System.out.println( "Sequence length of Real Data: " +
            sequenceLength );

        // generate viterbi path based on transition matrix and
            length of real Data
        RandomSequenceGenerator rndSeqGenerator = new
            RandomSequenceGenerator( sequenceLength,
            transitionMatrix, true );
        int[] initPath = rndSeqGenerator.generateSequence();

        System.out.println( IntArrays.toString( initPath ) );

        // read sde potentials from file
        MatrixFileReader potentialReader = new MatrixFileReader(
            sdeParamsFilename );
        double[][] sdeParams = potentialReader.getMatrix();

        GeneratorSDE sdeGenerator = new GeneratorSDE( initPath,
            sdeParams, dt );
        double[] trajectory = sdeGenerator.generateSequence( x_0 )
            ;

        // convert trajectory from <code>double[]<\code> to <code>
            double[][]<\code> for use in Metamacs

        double[][] SimulatedPath =
            convertObservationFromOneDimensionalInput( trajectory
            );
        double[][] RealPath =
            convertObservationFromOneDimensionalInput(
            readRealSequence.readDoubleColumn( 0 ));

        // start hmm-sde for simulated and real data
        HMMVAR hmmreal = new HMMVAR( RealPath, numberOfStates, 1,
            null );
        HMMVAR hmmsim = new HMMVAR( SimulatedPath, numberOfStates,
             1, null);
        // Initialize model
        hmmsim.initializeModel( initPath );
        hmmreal.initializeModel( initPath );
        //start EM algorithm
        double realmaxLikelihood = hmmreal.EMcorrect(1000, 10e-5,
            true);
        double simmaxiLikelihood = hmmsim.EMcorrect(1000, 10e-5,
            true);
        System.out.println("Maximum likelihood after HMM of
            simulated Data has converged:\n " + realmaxLikelihood
            );

        printHmmParameters( hmmsim );

        System.out.println("Maximum likelihood after HMM of real
            Data has converged:\n " + simmaxiLikelihood );

        printHmmParameters( hmmreal);
    }

    /**
     * One dimensional conversion to multi-dimesnional
     * {1, 2, 5, 4}  => {{1}, {2}, {5}, {4}}
     *
     * @param input is one dimensional array
     * @return observation suitable for HMMVAR
     */
    public static double[][]
        convertObservationFromOneDimensionalInput( double[] input
        ) {
        double[][] output = new double[ input.length ][1];
        for ( int i = 0; i < input.length; i++ ) {
            output[i][0] = input[i];
        }
        return output;
    }

    /**
     * Print estimated parameters of hmm
     * @param hmm is the HMMVAR whose estimations are printed
     */
    public static void printHmmParameters( HMMVAR hmm ) {
        double[][][] regressionMatrices = hmm.
            getRegressionMatrices();
        double[][][] covarianceMatrices = hmm.getCovariances();
        double[][] transitionMatrix = hmm.getTransitionMatrix();
        double[] initialDistribution = hmm.getInitialDistribution
            ();
        int[] v = hmm.computeViterbiPath();

        System.out.println("Estimated transition matrix: \n" +
            DoubleArrays.toString( transitionMatrix ));
        System.out.println("Estimated initial distribution: \n" +
            DoubleArrays.toString( initialDistribution ));
        for ( int i = 0; i < regressionMatrices.length; i++ ) {
            System.out.println("Hidden state " + i +": ");
            System.out.println("  Estimated regression matrix: \n"
                 + DoubleArrays.toString( regressionMatrices[i] ))
                ;
            System.out.println("  Estimated covariance matrix: \n"
                 + DoubleArrays.toString( covarianceMatrices[i] ))
                ;
        }
    }
}
```

```
import biocomp.moltools.util.FileLineReader;

/**
 * Read matrix from file
 * @author luedge
 *
 */
public class MatrixFileReader
        extends Object {

    private FileLineReader matrixReader;
    private double[][] matrix;
    private int[] column;

    public MatrixFileReader( String fileName ) {
        matrixReader = new FileLineReader( fileName );
    }

    /**
     *
     * @return matrix as double array
     */
    public double[][] getMatrix() {
        matrix = matrixReader.readDoubleTable();
        return matrix;
    }

    public int[] getIntColumn() {
        column = matrixReader.readIntColumn( 0 );
        return column;
    }
}
```

CHAPTER 6

Apendix C - Matlab Source Code

The first listing lists the use of the following scripts to reproduce the analysis and plots shown and employed in this thesis. It starts with importing the ASCII data at hand and will call the HMMVAR estimation at last.

```
%% Data Analysis

% Get Photon streams through import function
 FRET=stimporter(fileindex)

%% Discretazation if neccessary
photonizer %Automatic
%or
FNorm = normalize(odata) %with Manual selected threshold
%% Error correction
%Select FRET measurements that are suitabel (goodFRET)
% calculate background and crosstalk correction factors through
    bleaching markers

goodCFRET=errorcalc(goodCFRET)

% apply correction factors to trajectory

goodFRET=errorcorr(goodCFRET)

% sort for convenience in order of trajectorie length
for i=1:length(goodCFRET)
sortTime(i)=length(goodCFRET(1,i).counts)
end
[sortedTime Xperm]=sort(sortTime)
for j=1:length(goodCFRET)
goodSCFRET(:,j)=goodCFRET(:,Xperm(j));
end
%
for i=1:length(goodSCFRET)
sortTime2(i)=goodSCFRET(1,i).A_bleached;
end
[sortedTime2 Xperm2]=sort(sortTime2)
for j=1:length(goodSCFRET)
goodTSCFRET(:,j)=goodSCFRET(:,Xperm2(j))
end
```

```

%% State recognition through naive efficiency trajectory
tsInt = timewindow(tsin,n)
effiT = effer(tsInt)

for i=1:length(CSingleStates)
CSingleStates(1,i).ints10=timewindow(CSingleStates(1,i).counts,10)
    ;
CSingleStates(1,i).ints20=timewindow(CSingleStates(1,i).counts,20)
    ;
CSingleStates(1,i).ints50=timewindow(CSingleStates(1,i).counts,50)
    ;
CSingleStates(1,i).ints100=timewindow(CSingleStates(1,i).counts
    ,100);
end

for i=1:length(CSingleStates)
CSingleStates(1,i).eff10=effer(CSingleStates(1,i).ints10);
CSingleStates(1,i).eff20=effer(CSingleStates(1,i).ints20);
CSingleStates(1,i).eff50=effer(CSingleStates(1,i).ints50);
CSingleStates(1,i).eff100=effer(CSingleStates(1,i).ints100);
end

%% for many states per trajectorie use
SortedStateChanges=Statelizer(state_INFO)
%reorder $SortedStateChanges to $SChange
SingleStates=Singelizer(nuFRET, visualStates, SChange)

%% for trajectories staying in one state

for i=1:13
CSingleStates(i).counts=goodTSCFRET(3,i).counts(1:goodTSCFRET(1,i)
    .A_bleached,1:2);
CSingleStates(i).names=goodTSCFRET(3,i).names;
CSingleStates(i).part='1/1';
end

%% Converting to single photon trajectory
for i=1:length(CSingleStates)
CSingleStates(1,i).maxPhoton=max(max(CSingleStates(1,i).counts));
CSingleStates(3,i).monocounts=SinglePhotonizer(CSingleStates(1,i).
    counts);
end


% Enhancement: HMM for efficiency estimation
multifret(exp_data,tw1,tw2,tw3,start,time)
```

```
fret_filter



%% Photon Analysis
%  NProbe Efficiency

for i=1:length(CSingleStates)
CSingleStates(3,i).NPmaxPeff=NPanalyse(CSingleStates(3,i).
    monocounts,CSingleStates(1,i).maxPhoton,10);
end

for i=1:length(CSingleStates)
CSingleStates(3,i).NPmaxmax=NPanalyse(CSingleStates(3,i).
    monocounts,CSingleStates(1,i).maxPhoton,2*CSingleStates(1,i).
    maxPhoton);
end

for i=1:length(CSingleStates)
CSingleStates(3,i).NPmaxmax=NPanalyse(CSingleStates(3,i).
    monocounts,CSingleStates(1,i).maxPhoton,2*CSingleStates(1,i).
    maxPhoton);
end
for i=1:length(CSingleStates)
CSingleStates(3,i).NPmaxund2=NPanalyse(CSingleStates(3,i).
    monocounts,CSingleStates(1,i).maxPhoton,CSingleStates(1,i).
    maxPhoton+2);
end


% State recognition through efficiency Analysis
%TransitionMatrix from Trajectory
TransM=TransitionMatrixFromTraj(conIn, samples, stepsize,
    countmode)

% Divide Trajectories into
% I) Visual recognized
% and/or
% II) automatic clustered States.
% For uncorrected Photonstreams the bleaching startingpoint is set
     as end of trajectory.
% Make a list of trajectories showing state changes.

visualStates = [] ;

SortedStateChanges=Statelizer(state_INFO)
SingleStates=Singelizer(nuFRET, visualStates, SChange)
```

```
% Visual States are imported through marking every change point
    with the
% DataTip tool, export the cursor_info structure of all
    trajectories to create $StateChange

StateChange=Statelizer(cursor_info);

SingleStates = Singelizer(oxyFRET, visualStates, SChange );
StateAnalysis=SingleStateAnalyser(struct SingleStates)

%% HMM Analysis
% random Viterbi path
RandVitPath=VPGenerate(length,TransitionMatrix)

% Viterbipath from Transition Matrix
TMviterbiPfad=randViterbi(States,length)

%% NProbe Analysis
NPAnalysis=Npanalyse(Np, tw)
```

```
function DAeffR = effer(tsInt)
%% Calcuates efficiency for each two values in the the row-vector
    $tsInt(:,1:2)
length = uint32(size(tsInt,1));

   for i=1:length
       DAeffR(i,1) = tsInt(i,2)/(tsInt(i,1)+tsInt(i,2));
   end
```

```
function Est=Estimate(trajec, dimHmm, previterbi, steps, G_SDE)
%% Estimates the input trajectorie $trajec with the HMM Variants
    class of %% the Metamacs Package (change javaclasspath to your
     metamacs lib)
% * Eike Meeerbach et al.: Sequential change point detection in
    molecular dynamics trajectories,  % * submitted to the Journal
     of multivariate Analysis (2008).
% $dimHmm is the number of States,  % Â§steps is the aboirt
    criterioin for the EM algorithm
javaclasspath({    '/home/cocktail/luedge/NetBeansProjects/
    Metamacs/lib/colt.jar'...    '/home/cocktail/luedge/
    NetBeansProjects/Metamacs/lib/concurrent.jar'...    '/home/
    cocktail/luedge/NetBeansProjects/Metamacs/netbeans_project/
    build/classes/'...    '/home/cocktail/luedge/NetBeansProjects/
    Metamacs/netbeans_project/dist',...
 });
import biocomp.metamacs.hmm.newImplementation.HMMVAR; precision
    =1.0000e-04;
```

```
%% -------- Create random Viterbipath if not given (poor results)
    % if  previterbi==0; previterbi=randViterbi(dimHmm,size(trajec
    ,1)); Est.preVit=previterbi;       else end
%% -       Estimate Maximum Likelihood of HMMSDE - Model with
    HMMVAR( , , 1, )    - hmm=HMMVAR(trajec, dimHmm, G_SDE, []);
    hmm.initializeModel(previterbi) Est.hmm=hmm; % Est.MLH = hmm.
    EMcorrect(steps, precision, true); Est.MLH = hmm.EM(steps,
    precision, true);
%% --------------------Get-estimated parameter------------ Est.
    TransitionMatrix=hmm.getTransitionMatrix(); % Est.ObsProb =
    hmm.getObservationProbabilities(); Est.FwBw = hmm.
    getForwardBackwardVariables();
CovM=hmm.getCovariances(); Phi=hmm.getRegressionMatrices;
Est.Phi=Phi; Est.RegM = squeeze(Phi(1,:,:)); if G_SDE==1 for i=1:
    size(Phi,1) Cov(i)=squeeze(CovM(i,:,:)); %#ok<AGROW> S(:,i)=
    squeeze(Est.Phi(i,:,:)); %#ok<AGROW> mu(i)=S(1,i)/(1-S(2,i));
    %#ok<AGROW> % muM=inv(eye(size(S))-S)*Phi(1,:); end Est.
    expTauF=S; Est.mu=mu; Est.Cov=Cov; else
% Est.muM=muM; Est.CovM=CovM;
end
%% -       final Viterbi-path as initial distribution        - Est
    .viterbi=hmm.computeViterbiPath();  Est.InitDist= hmm.
    getInitialDistribution();
```

```
function NPAnalysis=NPanalyse(singleFRET, Np, tw)
javaclasspath({    '/home/cocktail/meerbach/Netbeans_projects/
    Metamacs/dist/lib/colt.jar'...    '/home/cocktail/meerbach/
    javawerkstatt/Metamacs/lib/concurrent.jar'...    '/home/
    cocktail/meerbach/Netbeans_projects/Metamacs/build/classes/'
    ...    '/home/cocktail/luedge/code/mbtoolkit/torsten/
    FRETanalyser/netbeans_project/build/classes'...    '/home/
    cocktail/luedge/code/mbtoolkit/torsten/FRETanalyser/NProbe/
    build/classes'...    '/home/cocktail/luedge/NetBeansProjects/
    Metamacs/build/classes/biocomp/metamacs/hmm/newImplementation'
    ...    }); if isstruct(singleFRET)==1;     NPAnalysis=
    singleFRET;     for i=1:length(singleFRET)      Np=2;      tw
    =20;           aFRET=Nprobe.probe(singleFRET(1,i).counts(:,2),
    singleFRET(1,i).counts(:,1),Np,tw);     end           NPmean=
    mean(aFRET);            NPvar=var(aFRET);         NPAnalysis
    (1,i).NPeff=aFRET;         NPAnalysis(1,i).NPmean=NPmean;
            NPAnalysis(1,i).NPvar=NPvar;         NPAnalysis(1,i).
    NPstd=sqrt(NPvar);         NPAnalysis(1,i).NPvarest1=2*NPvar-
    NPmean-NPmean*NPmean;         NPAnalysis(1,i).NPvarest1=2*
    NPvar-NPmean+NPmean*NPmean;
else     NPAnalysis=Nprobe.probe(singleFRET(:,2),singleFRET(:,1),
    Np,tw); end
```

```
function NPAnalysis=NPanalyse(singleFRET, Np, tw)
javaclasspath({    '/home/cocktail/meerbach/Netbeans_projects/
    Metamacs/dist/lib/colt.jar'...    '/home/cocktail/meerbach/
    javawerkstatt/Metamacs/lib/concurrent.jar'...    '/home/
    cocktail/meerbach/Netbeans_projects/Metamacs/build/classes/'
    ...    '/home/cocktail/luedge/code/mbtoolkit/torsten/
    FRETanalyser/netbeans_project/build/classes'...    '/home/
    cocktail/luedge/code/mbtoolkit/torsten/FRETanalyser/NProbe/
    build/classes'...    '/home/cocktail/luedge/NetBeansProjects/
    Metamacs/build/classes/biocomp/metamacs/hmm/newImplementation'
    ...    }); if isstruct(singleFRET)==1;     NPAnalysis=
    singleFRET;     for i=1:length(singleFRET)      Np=2;      tw
    =20;           aFRET=Nprobe.probe(singleFRET(1,i).counts(:,2),
    singleFRET(1,i).counts(:,1),Np,tw);     end           NPmean=
    mean(aFRET);            NPvar=var(aFRET);         NPAnalysis
    (1,i).NPeff=aFRET;         NPAnalysis(1,i).NPmean=NPmean;
            NPAnalysis(1,i).NPvar=NPvar;         NPAnalysis(1,i).
    NPstd=sqrt(NPvar);         NPAnalysis(1,i).NPvarest1=2*NPvar-
    NPmean-NPmean*NPmean;         NPAnalysis(1,i).NPvarest1=2*
    NPvar-NPmean+NPmean*NPmean;
else     NPAnalysis=Nprobe.probe(singleFRET(:,2),singleFRET(:,1),
    Np,tw); end
```

```
function SortedStateChanges=Statelizer(state_INFO) %% Creates a
    Trajectory/State Matrix containing statechange points (cols)
    %% for every trajectory (rows)
k=1; l=1; DataTips=length(state_INFO);
for i=1:DataTips     current_trajectory=state_INFO(1,DataTips+1-i)
    .Target;     StateChanges(k,l)=state_INFO(1,DataTips+1-i).
    Position(1,1);      l=l+1;          if i==DataTips &&
    state_INFO(1,2).Target==current_trajectory       %
    StateChanges(k,l)=state_INFO(1,DataTips+1-i).DataIndex;
                 elseif i==DataTips && state_INFO(1,2).Target~=
    current_trajectory         StateChanges(k,1)=state_INFO(1,1).
    Position(1,1);
         elseif state_INFO(1,DataTips-i).Target==
             current_trajectory         StateChanges(k,l)=
             state_INFO(1,DataTips+1-i).Position(1,1);
                          elseif state_INFO(1,DataTips-i).Target
             ~= current_trajectory         l=1;         k=k+1;
                     StateChanges(k,l)=state_INFO(1,DataTips+1-i).
             Position(1,1);
    end end
PreSorted=sort(StateChanges,2,'ascend'); T=size(StateChanges,1);
    for i=1:T     row=find(PreSorted(i,:));     e=length(row);
        SortedStateChanges(i,1:e)=PreSorted(i,row(1):row(end));
         end
```

```
function stemplots(inFRET,n) figure;  stem (inFRET(1,n).counts
    (:,1), 'Marker', 'p', 'MarkerSize',4,'Linestyle' , ':');  stem
     (inFRET(1,n).counts(:,2), 'MarkerSize', 2 , 'Linestyle' , '--
    ');
figure; stem (inFRET(2,n).counts(:,1), 'DisplayName', 'corrected',
     'Linestyle' , ':'); stem (inFRET(2,n).counts(:,2), '
    DisplayName', 'corrected', 'Linestyle' , '--');
h = stem(inFRET(1,n).counts(:,1:2), 'DisplayName', 'oxy 0272 D
    8460ms', 'fill'); set(h(1),'LineWidth', 0.2 , 'Marker', 'p', '
    MarkerSize',4,'Linestyle' , ':') set(h(2),'LineWidth', 0.1 , '
    MarkerSize',4,'Linestyle' , '--')
```

```
function tenPlotsDAEnhanced2(uFRET,uc,savePDF,part,fromto) %%
         MultiPlot  %   corrected for $uc=2 (uncorrected $uc=1) %
      saves .pdf of every plot for $savePDF=1, if not set = 0 %
    pages $fromto for specific pages if nargin<5     fromto(1,1)
    =1/10;     fromto(1,2)=ceil(size(uFRET,2)/10);     t=1; end if
     nargin<4     Tstart(1,1)=1;     part=0; end
if nargin<3    savePDF=0; end
if nargin<2     uc=1; end
if part==0     pt=1; elseif part==1     pt=2;     Tstart(1,1)=1;
     elseif part==2     pt=3;      end     t=fromto(1,1)*10; for s=
     fromto(1,1)*10:fromto(1,2)*10         figure(s)          for
     p1=1:10
         ende(1,1)=size(uFRET(uc,t).eff100,1);         ende(2,1)=
              uFRET(1,t).A_bleached;         ende(3,1)=size(uFRET(uc
              ,t).eff100,1);         xTix=[0,round(ende(pt)/4),ende(
              pt)-round(ende(pt)/4),ende(pt)];         xTix(end)=
              ende(pt);         xTixL=round(xTix/1000);         yMax
              =max(max(uFRET(uc,t).ints100));         xDbleach=uFRET
              (1,t).D_bleached;         yDbleach=uFRET(1,t).eff100(
              uFRET(1,t).D_bleached,1);         xAbleach=uFRET(1,t).
              A_bleached;         yAbleach=uFRET(1,t).eff100(uFRET
              (1,t).A_bleached,1);         Tstart(1,1)=1
              Tstart(3,1)=xAbleach-250;         Tstart(2,1)=1;
                                        subplot(10,4,4*p1-3:4*p1-1)
                            plotI=line(Tstart(pt,1):ende(pt),uFRET(uc,
              t).ints100(Tstart(pt,1):ende(pt),1:2));
                                           text(xAbleach,yAbleach,'\
              bf{A} bleached \rightarrow','Rotation',33,'Fontsize'
              ,8, 'HorizontalAlignment','right');
                                    text(xDbleach,yDbleach,'\bf{D}
              bleached \rightarrow','Rotation',33,'Fontsize',8, '
              HorizontalAlignment','right');                    set(
              plotI(1),'Color',[0.431 0.612 0.522]);          set(
              plotI(2),'Color',[0.894 0.525 0.612]);
                                   ax1 = gca;          xyPos=get(ax1,'
              Position')-[0.09 0 -0.1 0];                set(ax1, '
              Position',xyPos,...                        'YColor','b','
              YAxisLocation','left',...                          'Color',
              'none','xtick',[],...                         'XLim',[
              Tstart(pt,1),ende(pt)],'YLim',[0,yMax]);          if
              strcmp(uFRET(uc,t).names(1),'r')==1;              set(
              get(ax1,'XLabel'),'String',...                  ['time
              in sec   #',num2str(t),' - ',uFRET(uc,t).names(7:9),
              uFRET(uc,t).names(17:20)])          else
              set(get(ax1,'XLabel'),'String',...                    ['
              time in sec   #',num2str(t),' - ',uFRET(uc,t).names
              (1:2),uFRET(uc,t).names(11:14)])          end
```

```
        if (uc==2)              ylabel(ax1,'Intensities (cor)');
                    else              ylabel(ax1,'Intensities (uc)'
            );                        end                ax2 =
            axes('Position',get(ax1,'Position')+[0.05 0 0.08
            0],...              'YAxisLocation','right',...
                        'Color','none','XColor','k','YColor','k'
            ,...             'xtick', xTix, 'XTickLabel', xTixL, '
            XLim',[Tstart(pt,1),ende(pt)],...              'YLim'
            ,[0,1.0],'YTick',[0,0.5,1],'YTickLabel',{'0';'50%';'
            100%'});
        line(1:ende(pt),uFRET(uc,t).eff100(1:ende(pt)),'Color','k'
            ,'Parent',ax2);          ylabel(ax2,'Efficiency')
                    set(ax2,'Position',get(ax1,'Position'));
        subplot(10,4,4*p1)
        hist(uFRET(uc,t).eff100(Tstart(pt,1):ende(pt)),100,'
            EdgeColor',[0.55 0.17 0.08]);
        ax3=gca;         xyhPos=get(ax3,'Position');
            xyhPos(1)=0.76;           xyhPos(3)=0.2;         xyhPos
            (4)=0.065;         set(ax3, 'Position', xyhPos,...
                        'YAxisLocation','right','XAxisLocation','
            bottom',...              'xtick', [], 'XTickLabel','','
            YLimMode','auto',...              'YColor',[0.04 0.1
            0.5],...             'Color','none','YColor','k','Box'
            ,'off',...              'CameraUpVector', [-1,1,0],'
            XDir','reverse')          axis tight         maxCounts=
            get(ax3,'YLim');                    if p1==10
            ylabel(ax3,'counts','Rotation',[0],'Position',[-0.02,
            maxCounts(1,2),1],'fontweight','demi'); %#ok<NBRAK>
                    %newCamera=get(gca,'CameraPosition')*[1 0 0;0
            1 0;0 0 -1]   ...'CameraPosition',newCamera, ;
                    end         t=t+1;
    end     maximize     if savePDF==1         if (uc==2)
                    save2pdf(['CorFRET',num2str(s)],gcf,600)
                else         save2pdf(['UncorFRET',num2str(s)],gcf
        ,600)         end     else      end   end end
```

```
function out01=SinglePhotonizer(inN) %% generates a photonstream
     with max photons 1 per count from multiple %% counting streams
      through assuming identical distributed arrivaltimes maxN=max(
     max(inN)); out01=zeros(maxN*length(inN),2);
% length(inN) % length(out01) % inN=a; % i=5 % maxN=max(max(a)); %
      out01=zeros(maxN*length(a),2);
k=1;
for i=1:length(inN)
```

```
    if inN(i,1:2)==[0 0];     k=k+maxN  ;        else
        photonsupplyD(:,1)=[ones(1,inN(i,1)),zeros(1,maxN-inN(i,1)
        )];      out01(k:k+maxN-1,1)=photonsupplyD(randperm(maxN))
        ';             photonsupplyA(:,1)=[ones(1,inN(i,2)),zeros
        (1,maxN-inN(i,2))];      out01(k:k+maxN-1,2)=photonsupplyA
        (randperm(maxN))';      k=k+maxN;    end    end
```

```
 function SingleStates=Singelizer(inFRET, SortedStateChanges, from
     ) %% Creates the structured array $SingleStates of all
     Trajectories (row) and States (col) %  Trajectories $from are
      pointer to Data collection structure nuFRET which %  to be
     statelized %  visually defined states in $Schange are divided
      into $SingleStates
nstreams = size(SortedStateChanges,1); last=zeros(nstreams+1,1);
     for i=1:nstreams         last(i+1,1)=find(SortedStateChanges(i
     ,:)==0,1);     if isfield(inFRET, 'A_bleached')==0         out
     =length(inFRET(1,from(1,i)).counts);     else     out=inFRET
     (1,from(1,i)).A_bleached;     end     SortedStateChanges(i,
     last(i+1,1))=out;          anf=1;     SingleStates=struct([]);
          from(1,i)
   for l=1:last(i+1)         k=l+sum(last(1:i));
        SingleStates(1,k).names=inFRET(1,from(1,i)).names;
        SingleStates(1,k).part=[num2str(l),'/',num2str(last(i+1,1))
        ];          ende=SortedStateChanges(i,l)-20;
        SingleStates(1,k).counts(:,1:2)=inFRET(1,from(1,i)).counts(
        anf:ende,1:2) ;            % SingleStates(1,k).monocounts
        (:,1:2)=inFRET(1,from(1,i)).monocounts(anf:ende,1:2) ;
                  anf=ende+40;    end          end
```

```
function viterbipfad=randViterbi(States,length) viterbipfad=zeros
     (1:length,1); a=0:1:States-1; R = rand(length,1);
    for i=1:States viterbipfad(find(R<(i*(1/States))&R>(i-1)*(1/
        States)),1)=a(i);     end
```

```
function NPAnalysis=NPanalyse(singleFRET, Np, tw)
javaclasspath({    '/home/cocktail/meerbach/Netbeans_projects/
    Metamacs/dist/lib/colt.jar'...    '/home/cocktail/meerbach/
    javawerkstatt/Metamacs/lib/concurrent.jar'...    '/home/
    cocktail/meerbach/Netbeans_projects/Metamacs/build/classes/'
    ...    '/home/cocktail/luedge/code/mbtoolkit/torsten/
    FRETanalyser/netbeans_project/build/classes'...    '/home/
    cocktail/luedge/code/mbtoolkit/torsten/FRETanalyser/NProbe/
    build/classes'...    '/home/cocktail/luedge/NetBeansProjects/
    Metamacs/build/classes/biocomp/metamacs/hmm/newImplementation'
    ...    }); if isstruct(singleFRET)==1;     NPAnalysis=
    singleFRET;     for i=1:length(singleFRET)      Np=2;      tw
    =20;           aFRET=Nprobe.probe(singleFRET(1,i).counts(:,2),
    singleFRET(1,i).counts(:,1),Np,tw);     end           NPmean=
    mean(aFRET);             NPvar=var(aFRET);          NPAnalysis
    (1,i).NPeff=aFRET;          NPAnalysis(1,i).NPmean=NPmean;
            NPAnalysis(1,i).NPvar=NPvar;         NPAnalysis(1,i).
    NPstd=sqrt(NPvar);          NPAnalysis(1,i).NPvarest1=2*NPvar-
    NPmean-NPmean*NPmean;          NPAnalysis(1,i).NPvarest1=2*
    NPvar-NPmean+NPmean*NPmean;
else     NPAnalysis=Nprobe.probe(singleFRET(:,2),singleFRET(:,1),
    Np,tw); end
```

```
function FNorm = normalize(odata) % entfernt das Rauschen und
    ersetzt die Intensiï¿½ten mit ganzen % Photonenanzahlen
total_rows = uint32(size(odata,1)) z1 = input('Rauschschwelle des
    Akzeptor Kanals: '); a1 = input('Gernzwert einzelnes Akzeptor
    Photonen: '); b1 = input('Grenzwert von hï¿½chstens 2 Akzeptor
     Photonen: '); z2 = input('Rauschschwelle des Donator Kanals:
    '); a2 = input('Gernzwert einzelnes Donator Photonen: '); b2 =
     input('Grenzwert von hï¿½chstens 2 Donator Photonen: '); %
    alles darï¿½ber sind erstmal 3 Photonen a=a1 b=b1 z=z1 tic;
    for m = 1:2     for n = 1:total_rows         if  odata(n,m) <
    z             FNorm(n,m) = z;          elseif odata(n,m) < a
                    FNorm(n,m) = 1;              elseif odata(n,
    m) < b                 FNorm(n,m) = 2;          else
                FNorm(n,m) = 3;         end     end     a=a2     b
    =b2     z=z2 end; toc t=toc;
%schneller mit odata(odata<z1=0); odata(odata>a1=1...)
```

```
function viterbipfad=multiViterbi(tlength, TransM, init) %%
    generates Vierbipath from transition Matrix, or if given a
    positiv %% integer it generates a iid random stateseries
    viterbipfad=zeros(1:tlength,1); R = rand(tlength,1); if nargin
     < 3     init=ceil(length(TransM)*rand(1,1)); end
```

```
if max(size(TransM))==1     States=TransM;     a=0:1:States-1;
         for i=1:States viterbipfad(find(R<(i*(1/States))&R>(i-1)
     *(1/States)),1)=a(i); %#ok<FNDSB>     end
else     States=size(TransM,2);     viterbipfad(1)=init;
     cTransM=zeros(States);     cTransM(:,States)=1;     for i=1:
     States         for m=1:States-1          cTransM(i,m)=sum(
     TransM(i,1:m));          end     end
     for t=1:tlength         viterbipfad(t+1)=find(cTransM(
         viterbipfad(t),:)>R(t),1);     end
end
```

```
function TSout=makeOneTrajvar(inFRET,vars,tw)
anf=1; for i=1:length(inFRET)      eff=effer(timewindow(inFRET(1,i
     ).counts,tw));      ende=length(eff);      OutStates(i,1)=find
     (vars>inFRET(1,i).NPvarmeans(1,1),1)-1; TSout(anf:anf-1+ende
     ,1:2)=[eff,i*ones(ende,1)-1]; anf=anf+ende; end for i=1:length
     (vars)-1 FoundStates=find(OutStates(:,1)==i); for k=1:length(
     FoundStates) TSout(TSout(:,2)==FoundStates(k),3)=(i-1)*ones;
     end clear FoundStates end
```

```
function outFRET=gammacor(inFRET) %% corrects discrete photon
     streams for background and crosstalk given in %% the $unFRET
     structure in .bg and .cross outFRET=inFRET; frets=size(inFRET
     ,2); for i=1:frets    outFRET(3,i).counts(:,1:2)=inFRET(3,i).
     counts; %% find nonzero values                nonZD=find(inFRET
     (3,i).counts(:,1));               %% caluclate probabilities
     for bg and x-talk photons to occur
     Dchances_pseudoG=rand(length(nonZD),1);
         del_pseudoG_D=Dchances_pseudoG>inFRET(1,i).gamma2(1,1);
         toDelD_pseudoG=nonZD(del_pseudoG_D);                   lD_g
              =length(toDelD_pseudoG);                   outFRET(1,i)
              .del_Gamma=zeros(lD_g,1);
%% delete photons from timeseries (and store deleted positions)
         outFRET(1,i).del_Gamma(1:lD_g,1)=toDelD_pseudoG;
                                   outFRET(3,i).counts(toDelD_pseudoG,1)=
              inFRET(3,i).counts(toDelD_pseudoG,1)-1;
              outFRET(3,i).counts(:,2)=inFRET(3,i).counts(1:end,2);
                     end
```

```
function outFRET=gammacalc(inFRET,where) if nargin<2     where=1
     else end outFRET=inFRET; for k=1:length(inFRET)
tA=inFRET(1,k).A_bleached; tD=inFRET(1,k).D_bleached; t_Dsolo=tD-
     tA;
ID_Dsolo=sum(inFRET(where,k).counts(tA:tD,1)); IA_Dsolo=sum(inFRET
     (where,k).counts(tA:tD,2));
```

```
pseudo_gamma1=((ID_Dsolo+IA_Dsolo)/t_Dsolo)/(sum(sum(inFRET(where,
     k).counts(1:tA,1:2)))/tA); pseudo_gamma2=((ID_Dsolo+IA_Dsolo)/
     t_Dsolo)/(sum(sum(inFRET(where,k).counts(tA-500:tA,1:2)))/500)
     ; outFRET(1,k).psG1=pseudo_gamma1; outFRET(1,k).psG2=
     pseudo_gamma2; end
```

```
function flucTraj=fluctuator(Viterbipath, flucVersion, params) %%
     set $flucVersion to 0 for gaussian noise, 1 for random walk
     adn 2 for %% stochastic differential equations %%
dur=length(Viterbipath); if flucVersion==0     for i=1:dur
     flucTraj(i,1)=Viterbipath(i)+params(Viterbipath(i))*randn;
         end      elseif=flucVersion==1     flucTraj(i,1)=
     Viterbipath(1)     for i=2:dur        flucTraj(i,1)=flucTraj(i
     -1,1)+params(Viterbipath(i))*rand     end      elseif=
     flucVersion==2      flucTraj(i,1)=Viterbipath(1)     for i=2:
     dur        flucTraj(i,1)=flucTraj(i-1,1)+flucTraj(i-1,1)*
     params(Viterbipath(i))*rand     end else     error('usage: "
     flucTraj=fluctuator(Viterbipath, flucVersion, params)", see
     mfile for versions') end
```

```
function corFRET=errorcorr(unFRET) %% corrects discrete photon
     streams for background and crosstalk given in %% the $unFRET
     structure in .bg and .cross frets=length(unFRET);
corFRET=unFRET; for i=1:frets corFRET(3,i).counts=unFRET(1,i).
     counts;     %% find nonzero values                nonZD=find(
     unFRET(1,i).counts(:,1));         nonZA=find(unFRET(1,i).
     counts(:,2));          %% caluclate probabilities for bg and x
     -talk photons to occur                           DchancesBG=
     rand(length(nonZD),1);         AchancesBG=rand(length(nonZA)
     ,1);          % there is assumend to be no crosstalk to the
     Donor channel!          AchancesCROSS=rand(length(nonZA),1);
                  del_bg_D=DchancesBG<unFRET(1,i).bg(1,1);
             del_bg_A=AchancesBG<unFRET(1,i).bg(1,2);
                  del_cross_A=AchancesCROSS<unFRET(1,i).cross;
             toDelA_cross=nonZA(del_cross_A);
     toDelD_bg=nonZD(del_bg_D);         toDelA_bg=nonZA(del_bg_A);
                 lA_bg=length(toDelA_bg);         lD_bg=length(
     toDelD_bg);                  lA_cross=length(toDelA_cross);
                 winner_bg=max(lA_bg,lD_bg);                %
     corFRET(1,i).Del_BGphotons(:,1:2)=zeros(winner_bg,2);
     corFRET(1,i).Del_CROSSphotons=zeros(lA_cross,1);
%% delete photons from timeseries (and store deleted positions)
       corFRET(1,i).Del_BGphotons(1:lD_bg,1)=toDelD_bg;
          corFRET(1,i).Del_BGphotons(1:lA_bg,2)=toDelA_bg;
```

```
        corFRET(1,i).Del_CROSSphotons(1:lA_cross,2)=toDelA_cross;
                              corFRET(3,i).counts(toDelD_bg,1)=
            unFRET(1,i).counts(toDelD_bg,1)-1;         corFRET(3,i
            ).counts(toDelA_bg,2)=unFRET(1,i).counts(toDelA_bg,2)
            -1;                     corFRET(3,i).counts(toDelA_cross
            ,2)=unFRET(1,i).counts(toDelA_cross,2)-1;
            corFRET(3,i).counts(toDelA_cross,1)=unFRET(1,i).counts
            (toDelA_cross,1)+1;          end
%%    %corFRET(1,i).counts(nonZA(find(del_bg_A)),2)=unFRET(1,i).
    counts(nonZA(find(del_bg_A)),2)-1; %corFRET(1,i).counts(nonZD(
    find(del_bg_D)),1)=unFRET(1,i).counts(nonZD(find(del_bg_D)),1)
    -1;
```

```
function cfFRET=errorcalc(nuFRET) cfFRET=nuFRET; %% Reads
    structured Array with bleaching markers (A_bleached,D_bleached
    ) %% and calculates background, crosstalk and no gamma
    correction for k=1:length(nuFRET) tA=nuFRET(1,k).A_bleached;
    tD=nuFRET(1,k).D_bleached; t_Dsolo=tD-tA;
ID_Dsolo=sum(nuFRET(1,k).counts(tA:tD,1)); IA_Dsolo=sum(nuFRET(1,k
    ).counts(tA:tD,2)); duration=size(nuFRET(1,k).counts,1); tdark
    =duration-tD; cfFRET(1,k).bgsum=sum(nuFRET(1,k).counts(tD:end
    ,:)); bg=cfFRET(1,k).bgsum/tdark;
cfFRET(1,k).bg=bg; cfFRET(1,k).cross=(IA_Dsolo-bg(1,2)*t_Dsolo)/
    t_Dsolo; cfFRET(1,k).machine_gamma=1; clear tA tD IDsolo
    IAsolo duration tdark bg
end
```

```
function DAeffR = effiR(tsInt1, tsInt2, tsInt3)
length = uint32(size(tsInt,1)); %einlesen der Parameter
                      R0 = input('FÃ¶rster Radius ');
                    n1 = input('Zetimittel 1 ');  n2 = input('
    Zetimittel 2 '); n3 = input('Zetimittel 3 '); row = input('
    EinfÃ¼gen ab Spalte'); %erstellen der Effizienz & Radius
    Matrix DAeffR=zeros(length,5+row)    %Errechnen und
    Positionieren der Effizienzen     for i=1:length-n1
    DAeffR(i,row) = tsInt(i,2)/(tsInt(i,1)+tsInt(i,2));
    DAeffR(i,row+1) = (R0.^6/DAeffR(i,1)-R0.^6).^(1/6);    end;
               for i=1:length-n2        DAeffR(i,row+2) = tsInt(i
    ,6)/(tsInt(i,5)+tsInt(i,6));        DAeffR(i,row+3) = (R0.^6/
    DAeffR(i,3)-R0.^6).^(1/6);    end;       for i=1:length-n3
           DAeffR(i,row+4) = tsInt(i,10)/(tsInt(i,9)+tsInt(i,10));
            DAeffR(i,row+5) = (R0.^6/DAeffR(i,5)-R0.^6).^(1/6);
    end;
```

```
function tenPlotsDAEnhanced2(uFRET,uc,savePDF,part,fromto) %%
         MultiPlot  %   corrected for $uc=2 (uncorrected $uc=1) %
      saves .pdf of every plot for $savePDF=1, if not set = 0 %
    pages $fromto for specific pages if nargin<5     fromto(1,1)
    =1/10;     fromto(1,2)=ceil(size(uFRET,2)/10);     t=1; end if
     nargin<4     Tstart(1,1)=1;     part=0; end
if nargin<3    savePDF=0; end
if nargin<2     uc=1; end
if part==0     pt=1; elseif part==1     pt=2;     Tstart(1,1)=1;
    elseif part==2     pt=3;      end     t=fromto(1,1)*10; for s=
    fromto(1,1)*10:fromto(1,2)*10          figure(s)          for
    p1=1:10
        ende(1,1)=size(uFRET(uc,t).eff100,1);         ende(2,1)=
            uFRET(1,t).A_bleached;         ende(3,1)=size(uFRET(uc
            ,t).eff100,1);         xTix=[0,round(ende(pt)/4),ende(
            pt)-round(ende(pt)/4),ende(pt)];         xTix(end)=
            ende(pt);         xTixL=round(xTix/1000);         yMax
            =max(max(uFRET(uc,t).ints100));         xDbleach=uFRET
            (1,t).D_bleached;         yDbleach=uFRET(1,t).eff100(
            uFRET(1,t).D_bleached,1);         xAbleach=uFRET(1,t).
            A_bleached;         yAbleach=uFRET(1,t).eff100(uFRET
            (1,t).A_bleached,1);         Tstart(1,1)=1
            Tstart(3,1)=xAbleach-250;         Tstart(2,1)=1;
                                    subplot(10,4,4*p1-3:4*p1-1)
                        plotI=line(Tstart(pt,1):ende(pt),uFRET(uc,
            t).ints100(Tstart(pt,1):ende(pt),1:2));
                                       text(xAbleach,yAbleach,'\
            bf{A} bleached \rightarrow','Rotation',33,'Fontsize'
            ,8, 'HorizontalAlignment','right');
                              text(xDbleach,yDbleach,'\bf{D}
            bleached \rightarrow','Rotation',33,'Fontsize',8, '
            HorizontalAlignment','right');                   set(
            plotI(1),'Color',[0.431 0.612 0.522]);         set(
            plotI(2),'Color',[0.894 0.525 0.612]);
                           ax1 = gca;         xyPos=get(ax1,'
            Position')-[0.09 0 -0.1 0];              set(ax1, '
            Position',xyPos,...                    'YColor','b','
            YAxisLocation','left',...                    'Color',
            'none','xtick',[],...                   'XLim',[
            Tstart(pt,1),ende(pt)],'YLim',[0,yMax]);         if
            strcmp(uFRET(uc,t).names(1),'r')==1;              set(
            get(ax1,'XLabel'),'String',...                 ['time
            in sec   #',num2str(t),' - ',uFRET(uc,t).names(7:9),
            uFRET(uc,t).names(17:20)])         else
            set(get(ax1,'XLabel'),'String',...                 ['
            time in sec   #',num2str(t),' - ',uFRET(uc,t).names
            (1:2),uFRET(uc,t).names(11:14)])         end
        if (uc==2)              ylabel(ax1,'Intensities (cor)');
                    else              ylabel(ax1,'Intensities (uc)'
            );                          end                     ax2 =
            axes('Position',get(ax1,'Position')+[0.05 0 0.08
            0],...              'YAxisLocation','right',...
                        'Color','none','XColor','k','YColor','k'
            ,...              'xtick', xTix, 'XTickLabel', xTixL, '
            XLim',[Tstart(pt,1),ende(pt)],...            'YLim'
            ,[0,1.0],'YTick',[0,0.5,1],'YTickLabel',{'0';'50%';'
            100%'});
        line(1:ende(pt),uFRET(uc,t).eff100(1:ende(pt)),'Color','k'
            ,'Parent',ax2);          ylabel(ax2,'Efficiency')
                    set(ax2,'Position',get(ax1,'Position'));
        subplot(10,4,4*p1)
        hist(uFRET(uc,t).eff100(Tstart(pt,1):ende(pt)),100,'
            EdgeColor',[0.55 0.17 0.08]);
        ax3=gca;         xyhPos=get(ax3,'Position');
            xyhPos(1)=0.76;          xyhPos(3)=0.2;         xyhPos
            (4)=0.065;         set(ax3, 'Position', xyhPos,...
                        'YAxisLocation','right','XAxisLocation','
            bottom',...             'xtick', [], 'XTickLabel','','
            YLimMode','auto',...             'YColor',[0.04 0.1
            0.5],...               'Color','none','YColor','k','Box'
            ,'off',...              'CameraUpVector', [-1,1,0],'
            XDir','reverse')          axis tight         maxCounts=
            get(ax3,'YLim');                  if p1==10
            ylabel(ax3,'counts','Rotation',[0],'Position',[-0.02,
            maxCounts(1,2),1],'fontweight','demi'); %#ok<NBRAK>
                    %newCamera=get(gca,'CameraPosition')*[1 0 0;0
            1 0;0 0 -1]   ...'CameraPosition',newCamera, ;
                    end         t=t+1;
    end     maximize     if savePDF==1         if (uc==2)
                    save2pdf(['CorFRET',num2str(s)],gcf,600)
                else         save2pdf(['UncorFRET',num2str(s)],gcf
        ,600)         end     else      end   end end
```

This page is intentionally left blank

Bibliography

[1] C. Singer: *Notes on the early history of microscopy*, Proceedings of the Royal Society of Medicine **7**, 247 (1914), URL `http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2003539`.

[2] E. N. Lorenz: *Deterministic nonperiodic flow*, Journal of the Atmospheric Sciences **20**, 130 (1963), URL `http://dx.doi.org/10.1175/1520-0469(1963)020\%3C0130:DNF\%3E2.0.CO;2`.

[3] S. D. Poisson: *Recherches sur la probabilité des jugements en matière criminelle et matière civile*, Bachelier **55** (1837), URL `http://books.google.com/books?`

[4] T. Levi-Civita: *La teoria di einstein e il principio di fermat*, Nuovo Cimento **16**, 105 (1918).

[5] Klaas: *A history of science in the Netherlands* (Brill Academic Publishers # Publisher: Brill Academic Publishers (September 1,), 1998), URL `\#`.

[6] R. Brown: *A Brief Account on Microscopical Observations on the Particles Contained in the Pollen of Plants and on the General Existence of Active Molecules in Organic and Inorganic Bodies*, in *Brown, Robert. 1866. The miscellaneous botanical works of Robert Brown: Volume 1. (Edited by John J. Bennett). R. Hardwicke, London. and later: Mabberley, D.J. 1985. Jupiter Botanicus: Robert Brown of the British Museum. Lubrecht and Cramer Ltd, London.* (1828).

[7] A. Einstein: *Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen*, Annalen der Physik **322**, 549 (1905), URL `http://dx.doi.org/10.1002/andp.19053220806`.

[8] E. Abbe: *Beiträge zur theorie des mikroskops und der mikroskopischen wahrnehmung*, Archiv für Mikroskopische Anatomie **9**, 413 (1873), URL `http://dx.doi.org/10.1007/BF02956173`.

[9] G. B. Airy: *On the supposed alteration in the amount of astronomical aberration of light, produced by the passage of the light through a considerable thickness of refracting medium*, Proceedings of the Royal Society of London **20**, 35 (1872), URL `http://gallica.bnf.fr/ark:/12148/bpt6k56114d`.

[10] *R. zsigmondy (1865-1929)*, Nature **206**, 139 (1965), URL `http://dx.doi.org/10.1038/206139a0`.

[11] George: *Microscopy and image analysis*, Current Protocols in Human Genetics **4** (2005), URL `\#`.

[12] F. zernike: *How I Discovered Phase Contrast*, Science **121**, 345 (1955), URL `http://adsabs.harvard.edu/cgi-bin/nph-bib\_query?bibcode=1955Sci...121..345Z`.

[13] T. A. Klar, E. Engel, and S. W. Hell: *Breaking abbe's diffraction resolution limit in fluorescence microscopy with stimulated emission depletion beams of various shapes*, Physical Review E **64**, 066613+ (2001), URL `http://dx.doi.org/10.1103/PhysRevE.64.066613`.

[14] M. Minsky: *Memoir on inventing the confocal scanning microscope*, Scanning **10**, 128 (1988), URL `http://web.media.mit.edu/\%7Eminsky/papers/ConfocalMemoir.html`.

[15] E. A. Jares-Erijman and T. M. Jovin: *Fret imaging.*, Nat Biotechnol **21**, 1387 (2003), URL `http://dx.doi.org/10.1038/nbt896`.

[16] M. Yoshida, E. Muneyuki, and T. Hisabori: *Atp synthase–a marvellous rotary engine of the cell.*, Nat Rev Mol Cell Biol **2**, 669 (2001), URL `http://dx.doi.org/10.1038/35089509`.

[17] R. Yasuda, T. Masaike, K. Adachi, H. Noji, H. Itoh, and K. Kinosita: *The atp-waiting conformation of rotating f1-atpase revealed by single-pair fluorescence resonance energy transfer.*, Proceedings of the National Academy of Sciences of the United States of America **100**, 9314 (2003), URL `http://dx.doi.org/10.1073/pnas.1637860100`.

[18] A. Y. Kobitski, A. Nierth, M. Hengesbach, A. Jäschke, M. A. R. K. Helm, and U. G. Nienhaus: *Exploring the free energy landscape of small rna molecules by single-pair förster resonance energy transfer.*, Biophysical Reviews and Letters **03**, 439+ (2008), URL `http://dx.doi.org/10.1142/S1793048008000873`.

[19] Kobitski, Andrei, Nierth, Alexander, Helm, Mark, Jaschke, Andres, Nienhaus, and G. Ulrich: *Mg2+-dependent folding of a diels-alderase ribozyme probed by single-molecule fret analysis*, Nucleic Acids Research **35**, 2047 (2007), URL `http://dx.doi.org/10.1093/nar/gkm072`.

[20] T. Förster: *Fluoreszenz organischer Verbindungen* (Vandenhoeck & Ruprecht, 1982).

[21] X. Zhang and T. C. Bruice: *Diels-alder ribozyme catalysis: a computational approach.*, J Am Chem Soc **129**, 1001 (2007), URL `http://dx.doi.org/10.1021/ja067416i`.

[22] G. F. Schröder and H. Grubmüller: *Maximum likelihood trajectories from single molecule fluorescence resonance energy transfer experiments*, J. Chem. Phys. **119**, 9920 (2003), URL `http://dx.doi.org/10.1063/1.1616511`.

[23] S. A. Mckinney, C. Joo, and T. Ha: *Analysis of single-molecule fret trajectories using hidden markov modeling.*, Biophysical journal **91**, 1941 (2006), URL `http://dx.doi.org/10.1529/biophysj.106.082487`.

[24] M. Andrec, R. M. Levy, and D. S. Talaga: *Direct determination of kinetic rates from single-molecule photon arrival trajectories using hidden markov models*, The Journal of Physical Chemistry A **107**, 7454 (2003), URL `http://dx.doi.org/10.1021/jp035514+`.

[25] I. Horenko, E. Dittmer, A. Fischer, and C. Schütte: *Automated Model Reduction for Complex Systems exhibiting Metastability*, Mult. Mod. Sim., 5 (3). pp. 802-827. Official URL: http://dx.doi.org/10.1137/050623310 (2005).

[26] E. Meerbach and C. Schütte: *Sequential Change Point Detection in Molecular Dynamics Trajectories*, Preprint submitted to Elsevier, to be published (2008).

[27] D. Talaga: *Markov processes in single molecule fluorescence*, Current Opinion in Colloid & Interface Science **12**, 285 (2007), URL `http://dx.doi.org/10.1016/j.cocis.2007.08.014`.

[28] D. S. Talaga: *Information theoretical approach to single-molecule experimental design and interpretation*, J. Phys. Chem. A **110**, 9743 (2006), URL `http://dx.doi.org/10.1021/jp062192b`.

[29] S. Park and V. S. Pande: *Validation of markov state models using shannon's entropy*, The Journal of Chemical Physics **124** (2006), URL `http://dx.doi.org/10.1063/1.2166393`.

[30] J. R. Lakowicz: *Principles of Fluorescence Spectroscopy* (Kluwer Academic/-Plenum Publishers, 1999), 2nd ed.

[31] C. Theiss, F. J. Schmitt, S. Andree, C. Cardenas-Chavez, K. Wache, J. Fuesers, M. Vitali, M. Wess, S. Kussin, H. J. Eichler, and Eckert: *Excitation energy transfer in the phycobiliprotein antenna of acaryochloris marina studied by transient fs absorption and fluorescence spectroscopy* (2008), pp. 339–342, URL `http://dx.doi.org/10.1007/978-1-4020-6709-9\_77`.

[32] R. Roy, S. Hohng, and T. Ha: *A practical guide to single-molecule fret*, Nature Methods **5**, 507 (2008), URL `http://dx.doi.org/10.1038/nmeth.1208`.

[33] T. Ha, A. Y. Ting, J. Liang, B. W. Caldwell, A. A. Deniz, D. S. Chemla, P. G. Schultz, and S. Weiss: *Single-molecule fluorescence spectroscopy of enzyme conformational dynamics and cleavage mechanism*, Proceedings of the National Academy of Sciences of the United States of America **96**, 893 (1999), URL `http://www.pnas.org/content/96/3/893.abstract`.

[34] N. K. Lee, A. N. Kapanidis, Y. Wang, X. Michalet, J. Mukhopadhyay, R. H. Ebright, and S. Weiss: *Accurate fret measurements within single diffusing biomolecules using alternating-laser excitation*, Biophys. J. **88**, 2939 (2005), URL `http://dx.doi.org/10.1529/biophysj.104.054114`.

[35] H. D. Kim, U. G. Nienhaus, T. Ha, J. W. Orr, J. R. Williamson, and S. Chu: *Mg2+-dependent conformational change of rna studied by fluorescence correlation and fret on immobilized single molecules*, Proceedings of the National Academy of Sciences of the United States of America **99**, 4284 (2002), URL `http://dx.doi.org/10.1073/pnas.032077799`.

[36] G. Caliskan, C. Hyeon, U. Perez-Salas, R. M. Briber, S. A. Woodson, and D. Thirumalai: *Persistence Length Changes Dramatically as RNA Folds*, Physical Review Letters **95** (2005), URL `http://dx.doi.org/10.1103/PhysRevLett.95.268303`.

[37] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling: *Numerical Recipes in C : The Art of Scientific Computing* (Cambridge University Press, 1992), URL `http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\&amp;path=ASIN/0521431085`.

[38] B. Schuler, E. A. Lipman, and W. A. Eaton: *Probing the free-energy surface for protein folding with single-molecule fluorescence spectroscopy*, Nature **419**, 743 (2002), URL `http://dx.doi.org/10.1038/nature01060`.

[39] X. Zhuang, L. E. Bartley, H. P. Babcock, R. Russell, T. Ha, D. Herschlag, and S. Chu: *A single-molecule study of rna catalysis and folding.*, Science **288**, 2048 (2000), URL `http://dx.doi.org/10.1126/science.288.5473.2048`.