

## **Distribution Agreement**

In presenting this dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I agree that the Library of the University shall make it available for inspection and circulation in accordance with its regulations governing materials of this type. I agree that permission to copy from, or to publish, this dissertation may be granted by the professor under whose direction it was written, or in his absence, by the Dean of the Graduate School when such copying or publication is solely for scholarly purposes and does not involve financial gain. It is understood, that any copying from, or publication of, this dissertation which involves potential financial gain will not be allowed without written permission.

Signature

---

Haowei Wang

---

Date

# A Single Molecule Study of Two Bacteriophage Epigenetic Switches

By  
Haowei Wang

Doctor of Philosophy  
Physics

Advisor: Dr. Laura Finzi

Approved for the Department by:

---

Advisor

---

Dr. David Dunlap  
Committee Member

---

Dr. Kurt Warncke  
Committee Member

---

Dr. Keith Berland  
Committee Member

---

Dr. Ivan Rasnik  
Committee Member

Accepted:

---

Lisa A. Tedesco, Ph.D. Dean of the Graduate School

---

Date

# A Single Molecule Study of Two Bacteriophage Epigenetic Switches

By

Haowei Wang

B.S., University of Science and Technology of China

China, 1998

Advisor

Laura Finzi, Ph. D.

An Abstract of  
A Dissertation submitted to the Faculty of the  
James T. Laney Graduate School Studies of Emory University  
in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy  
in Physics

2011

# Abstract

## A Single Molecule Study of Two Bacteriophage Epigenetic Switches

By Haowei Wang

Epigenetic switches allow organisms to evolve into different states by activating/repressing different sets of genes without mutations of the underlying DNA sequence. The study of epigenetic switches is very important to understand the mechanism of human development, the origin of cancer, mental illness and fundamental processes such as gene regulation.

The coliphage  $\lambda$  epigenetic switch, which allows switching from lysogeny to lysis, has been studied for more than 50 years as a paradigm, and has recently received renewed attention. Atomic force microscopy (AFM) was used here to show that the  $\lambda$  repressor oligomerizes on DNA, primarily as a dodecamer, to secure a DNA loop, which is the basis of the  $\lambda$  switch. This study also provides support for the idea that specifically bound repressor stabilizes adjacent, non-specifically bound repressor molecules, which confers robustness to the switch.

186 is a member of a different coliphage family. One of the major differences between the two coliphage families is that lambda phages can be induced to switch from the lysogenic to the lytic state by UV radiation, but most coliphages of P2 family, to which 186 belongs, cannot. Interaction between coliphage 186 repressor and DNA is characterized by AFM and tethered particle motion (TPM). To expedite analysis of the AFM data, MatLab codes were written to automate the laborious, manual tracing procedures. The programs automatically recognize DNA segments and protein particles in an image, in order to measure the DNA length and position of bound particles as well as their height, diameter and volume. Application of these algorithms greatly improved the efficiency of AFM analysis. It was showed that 186 CI dimers form heptameric wheels, which induce DNA wrapping and different kinds of DNA looping producing various conformations of nucleoprotein complexes. Information about the dynamics of DNA wrapping and looping on 186 CI particles was also obtained by TPM.

# A Single Molecule Study of Two Bacteriophage Epigenetic Switches

By

Haowei Wang

B.S., University of Science and Technology of China

China, 1998

Advisor

Laura Finzi, Ph. D.

A dissertation submitted to the Faculty of the  
James T. Laney Graduate School Studies of Emory University  
in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy  
in Physics

2011

# Acknowledgements

I owe my deepest gratitude to my advisor, Prof. Laura Finzi, for her generous encouragement, guidance and support throughout my academic study. These pages would be impossible without her kind mentorship and friendship over my five years of long stay. I sincerely thank Dr. David Dunlap for the kindness advising and patient teaching during my research.

I am heartily thankful to my Ph.D. advising committee members: Dr. Kurt Warncke, Dr. Keith Berland and Dr. Ivan Ransik for their encouraging words, thoughtful criticism, time and attendance to my annual meetings.

I am grateful to my two previous coworkers, Dr. Chiara Zurla and Dr. Carlo Manzo. They greatly helped me mastering a variety of lab techniques and developing critical thinking with constructive discussions during my first years of research in the lab.

I would like to thank the group members, Dr. Qing Shao, Dr. Sachin Goyal, Leila Afjehi-Sada for their thoughtful discussions and suggestions.

I was fortunate benefit from the support of our great collaborators Dr. Iann Dodd and Dr. Keith Shearwin not only for generously supplying proteins and DNA plasmids, but for their guidance and attentive comments to my results and analyses.

Many thanks also to the undergraduate students who worked with me, Andrew Robinson, Simon Yohannes, Nathan Cho, Il Kyu and John Mack. They helped me greatly in experiments and image analysis and did a great job.

I would also like to thank my parents and my girlfriend Sherry. Although they do not quite understand what I am doing, they always support me. They encouraged me when I felt low.

I owe much to my advisors in China, Prof. Yinmei Li and Prof. Liren Lou. They are the first people who lead me into this field of research and taught me a lot in my early time of scientific research.

I received guidance and assistance from many more people and friends over the years. It is impossible to list and thank them all. This dissertation is dedicated to all of them as well.

# Table of Contents

<b>Chapter 1</b> Introduction .....	1
§1.1 Epigenetic switches .....	2
§1.2 Prophage and the $\lambda$ epigenetic switch .....	4
§1.3 The 186 prophage .....	7
§1.3.1 Transcriptional interference .....	7
§1.3.2 Coliphage 186 .....	9
§1.3.3 Chromatin and DNA wrapping .....	11
<b>Chapter 2</b> AFM Studies of $\lambda$ Repressor Oligomers Securing DNA Loops .....	13
§2.1 Background .....	14
§2.2 Material and method .....	17
§2.2.1 Material .....	17
§2.2.2 DNA contour length on mica surface .....	19
§2.3 Result and discussion .....	21
§2.3.1 DNA contour length measured by AFM .....	21
§2.3.2 Specific binding to operator sites .....	23
§2.3.3 Weak affinity for the $O_{R3}$ operator site .....	25
§2.3.4 Multiple operators may recruit dimers .....	26
§2.3.5 Loop equilibrium .....	28
§2.3.6 Volume calibration .....	29
§2.3.7 Loop closures are prevalently dodecamers .....	33
§2.3.8 Alternative loop closures .....	35
§2.3.9 Conclusion .....	38
<b>Chapter 3</b> AFM and TPM Studies of DNA wrapping and looping of phage 186 .....	40
§3.1 Background and introduction .....	41
§3.2 Material and method .....	43
§3.2.1 AFM sample preparation .....	43

§3.2.2 TPM sample preparation .....	45
§3.2.3 Measurement of wheel diameter .....	46
§3.3 Result and discussion .....	49
§3.3.1 Conformation of basic model .....	49
§3.3.1.1 The 186 repressor and its assembling .....	49
§3.3.1.2 CI regulated mechanism .....	54
§3.3.2 Pseudo site on FL .....	59
§3.3.3 Asymmetric DNA wrapping on pR region .....	61
§3.3.4 DNA wrapping/unwrapping .....	62
§3.3.5 TPM study of DNA wrapping and looping .....	62
§3.3.6 DNA looping .....	67
§3.3.7 Other CI binding forms and non-specific binding .....	75
<b>Chapter 4</b> Automated DNA segmentation and protein recognition from AFM images	
§4.1 Background .....	77
§4.2 Method and algorithm.....	79
§4.2.1 Filtering.....	79
§4.2.2 Threshold and segmentation .....	80
§4.2.3 Thinning and selection of DNA skeleton .....	83
§4.2.4 DNA length estimation .....	87
§4.3 Application and programming .....	88
§4.3.1 DNA tracing .....	88
§4.3.2 Masking and interactive modifying .....	90
§4.3.3 DNA contour length measuring.....	95
§4.3.4 Automatic particle analysis on DNA or surface .....	97
§4.3.5 Auto-analysis of DNA protein interaction .....	101
§4.3.6 Data conversion.....	103
§4.4 Discussion .....	103
<b>References</b> .....	104



<b>Appendices</b> .....	110
Appendix A AFM studies of $\lambda$ repressor oligomers securing DNA loops .....	111
Appendix B The lambda bacteriophage epigenetic switch: new insight from single-molecule microscopy .....	119
Appendix C A missing link between transcription factors and nucleosomes: the bacteriophage 186 CI repressor wraps and loops DNA .....	139
Appendix D Matlab code for protein particle measurement and simulation .....	148
Appendix E Matlab code for automated tracing and DNA protein analysis programs	155

# List of Figures

## Chapter 1: Introduction

Figure 1.1 Life cycle of the prophage and the formation of a lysogen .....	5
Figure 1.2 CI protein and phage DNA interaction.....	7
Figure 1.3 Three promoter arrangements that lead to transcriptional interference .....	8
Figure 1.4 X-ray crystallography revealed 186 CTD .....	10
Figure 1.5 X-ray crystal structure of the nucleosome .....	12

## Chapter 2: AFM Studies of Repressor Oligomers Securing DNA Loops

Figure 2.1 Schematic representation of non-specific binding nucleation .....	15
Figure 2.2 Schematic drawing of an AFM .....	16
Figure 2.3 300 simulated 1500 bp polymer chain .....	19
Figure 2.4 Measured DNA basepair rise .....	21
Figure 2.5 AFM images of $\lambda$ CI and DNA .....	23
Figure 2.6 AFM measurement of the position of CI particles .....	25
Figure 2.7 Pairs of CI particles bound to adjacent sites .....	27
Figure 2.8 Loop ratio under different CI concentration .....	29
Figure 2.9 AFM measurements of the volume of CI protein particles .....	30
Figure 2.10 AFM measurements of the volume of lac repressor .....	31
Figure 2.11 AFM protein volume calibration .....	31
Figure 2.12 Oligomerization of CI securing DNA loops .....	33
Figure 2.13 Volume of particles securing DNA loops .....	35
Figure 2.14 Specifically bound protein particles .....	37

### **Chapter 3: AFM and TPM Study of DNA Wrapping and Looping of Phage 186**

Figure 3.1 Three groups of binding sites on 186 .....	41
Figure 3.2 Schematic drawing of TPM .....	43
Figure 3.3 AFM tip effects .....	44
Figure 3.4 A cross section of the image of a DNA molecule .....	47
Figure 3.5 Cross section of different shape particles .....	48
Figure 3.6 X-ray crystal structure of 14mer wheel of 186 CTD .....	50
Figure 3.7 Diameter of 186 particles.....	51
Figure 3.8 AFM volume calibration for 186 .....	53
Figure 3.9 Schematic representation of 186 repressor-DNA interaction .....	55
Figure 3.10 Frequency distribution histogram of 186 wheel location .....	56
Figure 3.11 186 wheel positioning at FL .....	60
Figure 3.12 DNA coverage of 186 particles on pR .....	61
Figure 3.13 TPM calibration curve .....	64
Figure 3.14 Frequency distribution of TPM data .....	65
Figure 3.15 Frequency distribution of TPM data .....	66
Figure 3.16 TPM trajectory .....	66
Figure 3.17 Frequency distribution of TPM data (wild type) .....	67
Figure 3.18 Frequency distribution of TPM data .....	72
Figure 3.19 Frequency distribution of TPM data .....	73
Figure 3.20 Frequency distribution histogram of occupancy .....	74
Figure 3.21 AFM images of 186 wt DNA with 300nm CI .....	75

## Chapter 4: Automated DNA Segmentation and Protein Recognition from AFM

### Images

Figure 4.1 The background of AFM images .....	79
Figure 4.2 Frequency distribution of z values .....	80
Figure 4.3 Zoomed view of the tail of fig. 4.2 .....	81
Figure 4.4 An original AFM image .....	82
Figure 4.5 Identification of critical pixels .....	84
Figure 4.6 One process of thinning .....	84
Figure 4.7 Working flow of the thinning process .....	85
Figure 4.8 The process of thinning .....	85
Figure 4.9 DNA skeleton .....	86
Figure 4.10 Interface of masking program. ....	90
Figure 4.11 Delete subfunction .....	91
Figure 4.12 Connect subfunction .....	92
Figure 4.13 The two DNA connection .....	93
Figure 4.14 User interface of a segment length measurement .....	94
Figure 4.15 Calculating the length of a part in the DNA segment .....	94
Figure 4.16 Dialog box for DNA length measurement .....	95
Figure 4.17 Contour length of 1934 bp DNA .....	96
Figure 4.18 Contour length of 1500 bp polymer chain .....	96
Figure 4.19 User interface panel of the particle analysis program .....	98
Figure 4.20 Output of particle analysis program .....	100
Figure 4.21 Saved excel data of particle analysis .....	101
Figure 4.22 Distribution histogram of protein binding position .....	102

# List of Tables

## Chapter 1: Introduction

## Chapter 2: AFM Studies of Repressor Oligomers Securing DNA Loops

Table 2.1 Percentages of CI-mediated loops .....	28
Table 2.2 Segment length for DNA loops .....	38

## Chapter 3: AFM and TPM Study of DNA Wrapping and Looping of Phage 186

Table 3.1 Statistics on the interaction between 186 and wt DNA .....	57
Table 3.2 Calibration data of TPM .....	63
Table 3.3 Statistics on the interaction between 186 CI and delta Pr mutation .....	68
Table 3.4 Statistical result of interaction on FL+pRFR- mutation .....	69
Table 3.5 Condensed statistical result of Tab. 3.4 .....	69

## Chapter 4: Automated DNA Segmentation and Protein Recognition from AFM

### Images

Table 4.1 Measured DAN contour length with different estimator .....	87
Table 4.2 Comparison of different tracing methods on DNA images .....	97

# **Chapter 1**

## **Introduction**

## § 1.1. Epigenetic switches

The Greek prefix *epi-* in epigenetic implies aspects “in addition to” genetics. According to the classical definition given by Ptashne and Gann [1] (p100), an epigenetic switch is the change of gene expression states which can be inherited, and maintained even though the signal that initiated that change is absent. In other words, an organism can evolve into different states by activate or repress different sets of genes thanks to epigenetic switches even if there is no mutation in the underlying DNA sequence and the environment is the same.

Scientists suspect epigenetic changes may represent a form of memory that allows organisms to remember their experience [2]. Reversible phenotype variability of epigenetic changes may provide a pathway for short-tem adaptation of species. For example, vernalization is the process by which plants flower only after having experienced a period of cold temperature and is the result of a epigenetic mechanism [3]. Transgenerational epigenetic inheritance is also observed in humans [4]. Different epigenetic features can be associated to different mutation rates. Indeed, organisms may control the mutation rate of particular genes epigenetically [5]. Furthermore, some epigenetic features are heritable from one generation to another when a DNA mutation takes place in the sperm or egg cell of an individual [6]. This indicates how epigenetic switches may affect adaptation and evolution. Similarly, epigenetic switches could be related to the differences between identical human twins [7].

Cellular differentiation in eukaryotes is another example of epigenetic switches. Totipotent stem cells develop into various pluripotent cell lines and finally transform

to fully differentiated cells during morphogenesis without any change of their gene sequences [8].

Currently, epigenetic modifications explicitly include some current field of research like DNA methylation and chromatin remodeling [9, 10], RNA interference [11, 12], prions [13] etc.. The importance of epigenetic marking on the development and failure of cloned sheep and bovines is well proved by several works [14-16]. The development of some human disorders also involve epigenetic effects [17, 18]. In the particular case of the Angelman syndrome and Prader-Willi syndrome, patients will develop different syndromes depending on the genomic imprinting inherited from their parents even though the syndrome is caused by same genetic mutation [17]. It is also reported that many agents that disturb the structural development of embryos or fetuses (for example, cleft lip or two headed new born creatures) affect the fetus by epigenetic mechanisms [19]. In addition, abnormal DNA methylation is also detected when a benign proliferation develops into an invasive cancer [20].

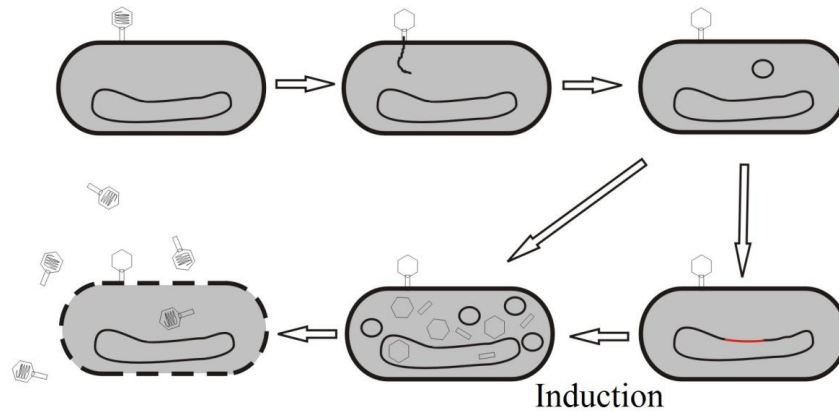
According to these findings, the study of epigenetic switches is important to understand the mechanism of human development, the origins of cancer, mental illness, as well as fundamental processes such as gene regulation etc. In 2008, the National Institutes of Health announced more than \$190 million funding for a new epigenomics initiative. According to past NIH Director Elias A. Zerhouni, “Epigenomics-based research is now a central issue in biology.”



## § 1.2. The prophage and the $\lambda$ epigenetic switch

Prophage is a state of coexistence of the host genome and the phage genome. Once a coliphage such as  $\lambda$  infects an *Escherichia coli* bacterium, it needs to make a decision between two developmental modes. Sometimes, the coliphage takes a lytic developmental pathway. In this case, various phage genes are turned on so that the phage genome is extensively replicated and new phage proteins are synthesized. After approximately 45 minutes, the cell lyses and releases about 100 new phages as illustrated in figure 1.1. Other times, the bacteriophage can go into a lysogenic state by inserting its genome into the bacterium's genome to form a lysogen. Once a lysogen is formed, all the phage genes except one are repressed and the cell becomes immune to other phages.

In a lysogen, the cell can grow and divide with the phage genome and the prophage passively replicates with the host cell. The cell can stay in this lysogenic state for a very long time until it is induced by some environmental change such as UV irradiation or starvation. For example, when a lysogen is irradiated by UV light virtually all the lysogen will switch to a lytic response, lyse the cell and produce a new crop of phages.



*Figure 1.1:* Life cycle of the prophage and the formation of a lysogen. Once a coliphage infects a bacteria, it can make a decision between lytic and lysogenetic responses. In the lytic response, the phage reproduces itself causing the lysis of the host, and releasing about 100 new phages (virulent reproduction pathway). In the lysogenic response, the phage inserts its genome into the cell and represses most of its genes to form a prophage. The prophage state is inheritable (quiescent reproduction pathway) and will be stable until changes in the environment stimulate the prophage to switch to a lytic response.

$\lambda$  is one of the most comprehensively studied bacteriophages. The relationship between phage  $\lambda$  and its host *Escherichia coli* is the archetype model system for the investigation of many fundamental biological processes, especially gene regulation [21].

The epigenetic switch between lysogeny and lysis in bacteriophage  $\lambda$  is controlled by one sole protein named  $\lambda$  repressor, or CI. The function of CI in the cell was established more than twenty years ago [22, 23]. CI maintains lysogeny by preventing transcription of multiple phage early genes such as N, cro, O, P and Q which are necessary for lytic development. It has already been understood that the establishment and maintenance of a lysogen require repression of both the pL and pR promoters that

are controlled by CI [21]. Furthermore, the evidence that CI affects transcription of pRM, the promoter that encodes CI, has been well described in 1981 [22].

The different roles of the CI binding sites in gene regulation have already been well examined in previous works [21, 24]. After dimerization, CI protein can bind on six binding sites cooperatively. It is believed that a CI dimer binding on  $O_{R1}$  will turn off the transcription of pR, but have no effect on pRM. Also, pL can be turned off by a CI dimer bound to  $O_{L1}$  without any other effect. However, if a CI dimer is bound to  $O_{R1}$ , a second will cooperatively bind to  $O_{R2}$ . CI binding on  $O_{R2}$  stimulates transcription from P<sub>RM</sub>. This leads to an over-expression of CI and eventually would prevent the lysogen to efficiently switch to lytic growth when necessary. Experiments conducted only on the  $O_R$  region of  $\lambda$  DNA had showed that CI on  $O_{R3}$  represses P<sub>RM</sub> and provides a mechanism of negative auto-regulation which would allow control of CI concentration. However,  $O_{R3}$  is a very weak site and can only be occupied at non-physiological concentrations of CI [21].

Since the two CI binding sites  $O_L$  and  $O_R$  are separated by a couple of thousands of base pairs, a long range cooperative mechanism involving DNA looping was demonstrated in 2005 and 2006 [24, 25]. According to this looping mechanism, the CI protein in its dimeric form can regulate three different promoters by binding to six different binding sites of the DNA in different ways as illustrated in figure 1.2. The protein binding on  $O_{L1-2}$  and  $O_{R1-2}$  sites can interact face-to-face and form a DNA loop. In this way, it stably represses transcription from pR and pL. It also brings  $O_{L3}$ , which is a strong binding site to face  $O_{R3}$  as indicated in figure 1.2. Therefore, the CI dimer binding on  $O_{L3}$  can stabilize a CI dimer on  $O_{R3}$  via a protein-protein interaction

and pRM is turned off at a physiological CI concentration. The first evidence of CI-mediated DNA loop formation and breakdown was provided *in vitro* in the Finzi lab [25-27].

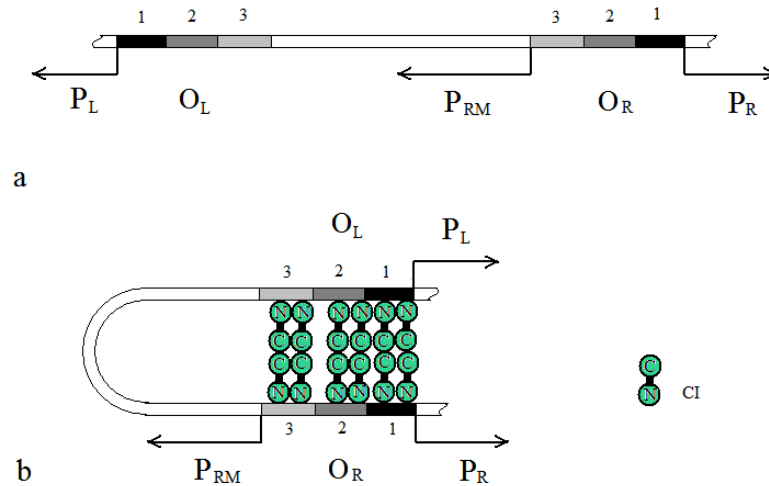


Figure 1.2: CI protein and phage DNA interaction. There are six different CI binding sites named  $O_{R1,2,3}$  and  $O_{L1,2,3}$  on the phage DNA through which transcription of three promoters (pR, pRM, pL) can be regulated (a). The model predicts that CI dimers may mediate DNA looping (b).

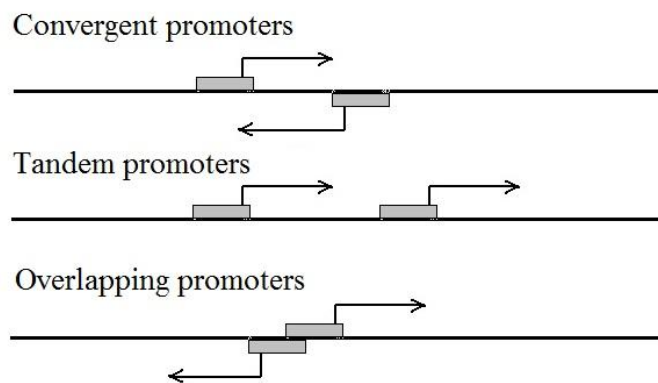
### § 1.3. The 186 bacteriophage

#### § 1.3.1 Transcriptional interference

Transcription of one gene may interrupt the transcription of a neighboring gene *in cis*. This ‘promoter occlusion’ was first found in prokaryotes [28] and later named ‘transcriptional interference’ [29]. It provides a new mechanism of gene regulation, especially for the not-well-studied function of untranslated RNAs [29, 30].

Transcriptional interference is widely found in coliphage [31], yeast [32], mammals [33] and drosophila [30, 34], and is used in the research of human diseases like cancer [35] and HIV [36, 37], and in strategies for drug development.

Normally, in transcriptional interference a strong promoter suppresses another weaker promoter. The three promoter arrangements that lead to transcriptional interference are illustrated in figure 1.3. They are: convergent promoters like the lytic and lysogenic promoters of coliphage 186 [31]; tandem promoters, like the yeast SRG1 and SER3 promoters [38] and overlapping promoters such as the *aroP* P1 and P3 promoters of *E. coli* [39].



*Figure 1.3:* Three promoter arrangements that lead to transcriptional interference. The two gray bars on the line represent two promoters on DNA. The arrows represent the direction of transcription. In the convergent case, RNA polymerase transcribing from one promoter will impact the polymerase sitting on or transcribing from the other promoter and kick it away. When the two promoters are in tandem, the RNA polymerase transcribing from the back promoter may approach and remove the RNAP on the other promoter. If two promoters overlap, RNA polymerase binding on one promoter will sterically prevent another RNAP from binding on the other promoter.

Based on these three promoter structural arrangements, five transcriptional interference mechanisms are demonstrated by Shearwin in 2005 [29]. When the two

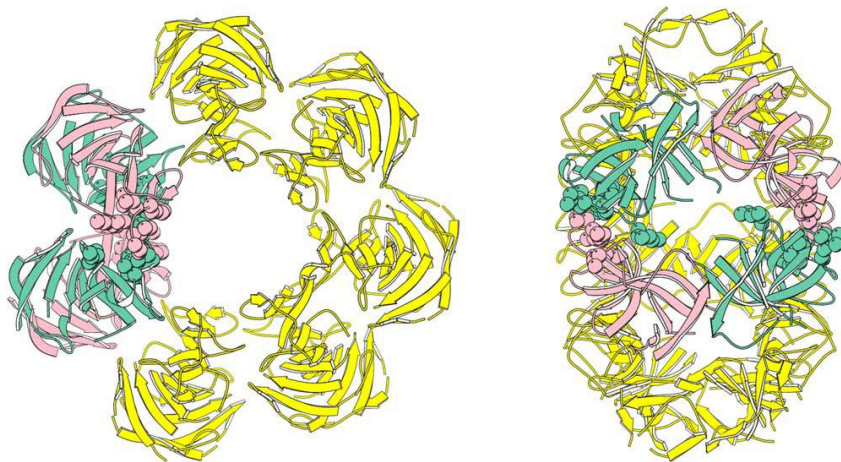
promoters are overlapping, occupancy of RNA polymerase on one promoter will preclude another polymerase from binding on the other promoter. A bound polymerase (but not transcribing) can be kicked off the DNA by a polymerase which is transcribing in the opposite direction from a different promoter. Polymerase binding at a given promoter can be prevented by another enzyme who had started from another promoter located either in a convergent or tandem geometry. Two transcribing polymerases may collide, and both leave the DNA. Finally, a tightly bound RNAP can act as a roadblock if it does not fall off the DNA by a transcribing RNAP.

### **§ 1.3.2 Coliphage 186**

Many coliphages exist in lysogenic hosts as prophages rather than free phage particles [40]. These phages can be roughly divided into inducible/noninducible groups by the ability to switch from a lysogenic growth to lytic growth under UV induction. The family of bacteriophages to which  $\lambda$  belongs consists of all inducible phages because they can all switch from lysogeny to a lytic response after exposure to UV light. As a member of the P2 family, coliphage 186 provides a noninducible counterpoint to  $\lambda$  phage [41]. Although 186 is almost unrelated to  $\lambda$  in DNA sequence [42], the lifecycles are almost the same [31]. Both phages maintain a genetic switch between lytic and lysogenic growth with one sole protein named in both cases CI. 186 CI, the lysogenic repressor of the 186 phage, is not sequence-related to  $\lambda$  CI even though they have very similar structure [31]. 186 CI and  $\lambda$  CI

both have one C terminal domain and one N terminal domain linked by a free peptide linker. Both of them bind to DNA with NTD and interact with other molecules of repressor with CTD. Unlike in the case of  $\lambda$  DNA, the 186 DNA contains three strong binding sites at pR and two flanking sites FL, FR [43]. Therefore, although 186 CI and  $\lambda$  CI can regulate transcription of their own gene both positively and negatively, depending on repressor concentration, their mechanisms must be very different.

X-ray studies show that the CTD of 186 repressor can form a wheel of seven dimers (fig. 1.4) [41]. Therefore, it is reasonable to suspect that the intact protein may also form a heptamer of dimers in nature. Even though the protein concentration for crystallographic studies is approximately 20 times higher than in normal bulk experiments [41], the idea of a wheel shaped repressor oligomer is intriguing. It could bind cooperatively to the multiple binding sites at pR and induce DNA wrapping and looping which, in turn, could explain how 186 can positively and negatively regulate the production of its repressor and maintain the lysogenic state [31].



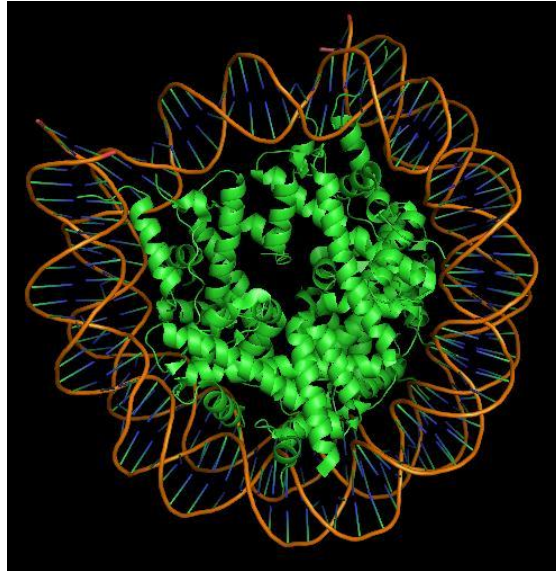
*Figure 1.4:* X-ray crystallography [41] revealed that 186 CTD can assemble into wheel-like particles. Each particle contains seven dimers. The wheel is approximately 102Å in diameter and 57 Å thick according to X-ray crystallography.

### § 1.3.3 Chromatin and DNA wrapping

Organismal genomes seldom exist as naked DNA. Their DNA is often bound by other proteins such as HU, IHF or histone proteins. In eukaryotes, DNA often wraps on histone proteins to form nucleosomes (fig.1.5) and chromatin. It is believed that the main function of chromatin is to package DNA to fit in the small volume of the cell nucleus. These nucleoprotein complexes can also strengthen the DNA during mitosis or meiosis and prevent DNA damage. In the 1980's, alternations of chromatin composition, structure and function were noticed and related to aging [44, 45]. In the past 20 years, more evidence that chromatin structure determines transcriptional control were presented [46]. For example, gene silencing in eukaryotes has been found to be related to DNA methylation [47].

The remodeling of chromatin provides a platform for gene silencing and activation [46]. If the hypothesis that DNA can wrap on the wheel-like particle of 186 repressor is correct, there might be functional similarities between DNA wrapped around the histone octamers and DNA wrapped around the 186 heptamer. Therefore, the study of DNA wrapping and unwrapping the 186 wheels may serve as a simplified model for chromatin remodeling.





*Figure 1.5:* X-ray crystal structure of the nucleosome. (PDB: 3AV1) [48]. DNA wraps around histone proteins by 1.67 turns.

## Chapter 2

# AFM Studies of $\lambda$ Repressor Oligomers Securing DNA Loops<sup>1</sup>

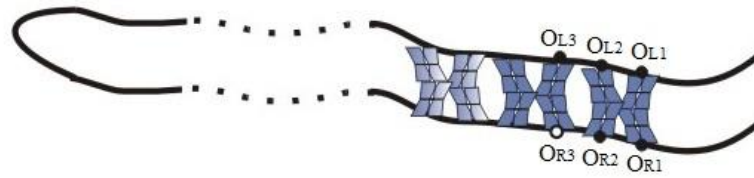
---

<sup>1</sup> This work was published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers.

## § 2.1. Background

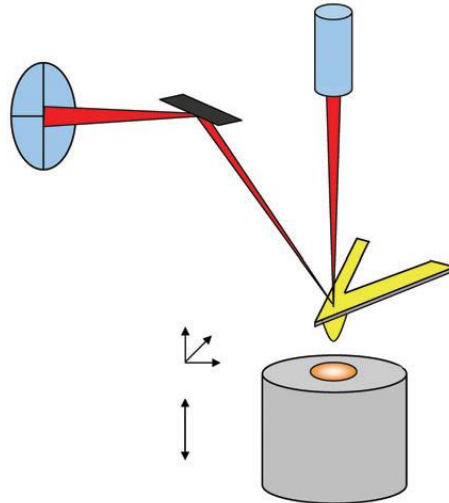
Prophage  $\lambda$  regulates repressor transcription by looping [21]. When a lysogen is formed, the phage DNA is looped by head-to-head interaction between CI tetramers binding respectively on two group of specific sites  $O_{RI-2}$  and  $O_{LI-2}$  separated by 2.3 bp [24-27] (fig. 1.2). The occupancy of  $O_{R1&2}$  will repress all lytic genes and stimulate the expression of the CI protein itself. Once the DNA loop is formed, the CI dimer binding at  $O_{L3}$  site may help another CI dimer to bind at  $O_{R3}$ , a weaker binding site, by head-to-head interaction. Thus, the transcription of CI protein is turned off and CI concentration is maintained at a level such that the lysogenic state can be maintained and the prophage can still switch to a lytic response if needed [21].

This looping model well explains the mechanism of the phage  $\lambda$ 's genetic switch. However, some recent work indicated that nonspecific interactions between CI and DNA may play a role in the formation of dynamic loop [49]. In particular, a dimer bound at either of the  $O_3$  operators could interact "side-by-side" with an adjacent dimer bound non-specifically. In principle, the  $O_L$  and  $O_R$  regions may function as a seeding spots for extensive CI oligomerization and loop closure [50-52] (fig. 2.1).



*Figure 2.1:* Schematic representation of non-specific binding nucleation. The solid circles on DNA (solid lines) are strong binding sites ( $O_{L1-3}$ ,  $O_{R1\&2}$ ). The hollow circle is the weaker site  $O_{R3}$ . CI dimers (blue boxes) first bind on  $O_{L1\&2}$  sites and loop the DNA with head-to-head interaction. After that, the CI dimer binding on  $O_{L3}$  can help another dimer binding to  $O_{R3}$  [21]. Because the two dimers binding on  $O_3$  sites do not have neighbor CIs to interact side-by-side, it is proposed that they can help other two dimers to bind non-specifically [49]. Further CI oligomerization may then occur inside the loop (light blue boxes) [50-52].

Atomic force microscopy (AFM) or scanning force microscopy (SFM) is a powerful microscopy technique where a probe is scanned on a surface to obtain its topographical profile [53]. The probe is constituted by a tip with a very sharp end. The tip is mounted on a soft cantilever which carries a mirror on its back side (fig. 2.2). Once the tip is approaching the surface, van der Waals forces will act on it before it contacts the surface. Thus, the cantilever will be bent to an angle that can be detected by the reflection of the laser beam shining on the back side of the cantilever. This optical detection of the change in force of interaction between tip and surface can provide the topography of the surface over which the sample is deposited [54]. AFM imaging is commonly achieved in “contact” mode.



*Figure 2.2:* Schematic drawing of an AFM [54]. A very sharp tip is attached at the bottom of a cantilever (yellow). The sample (orange spot) is put onto a piezoelectric ceramics (gray). A laser beam (red) is reflected by a mirror on the top of the cantilever and reflected to the detector (blue circle). The small change of distance between the sample and the tip will bend the cantilever and successively change the angle of reflected laser beam.

AFM can also produce images of the sample in “tapping” mode. In this case, the cantilever is driven by a piezo motor and made oscillate according to its resonance frequency. The oscillation range is then recorded by the detector. When the tip approaches the surface, its oscillation will be disturbed by the surface-tip interaction and the oscillation range will be changed also. Commonly, the height of the sample is controlled by another piezo motor so that the oscillation is kept within a constant range when scanning. Therefore, the voltage changes applied on the second piezo reflect the curvature of the sample surface. Using this method, the 3-D profile of the sample can be reconstructed. Compared to the contact mode, the tapping mode significantly decreases the damage done to the surface by lowering the force applied on it. Therefore, all the images in this study were obtained in tapping mode.

Unlike tethered particle microscopy (TPM) or magnetic tweezers (MT), the ability of AFM to obtain the 3-D topographic description of the sample surface can be used to directly visualize the structure of the protein-DNA complex and provide information that TPM and MT cannot.

## § 2.2. Materials and methods

### § 2.2.1 Material

1555 bp DNA fragments were produced by PCR amplification of segments of plasmids pDL944 and pDL965 using 5'-CGCAATTAATGTGAGTTAGCTCACTCA TTAGGCACCCCAGGC-3' and 5'-GCATTGCTTATCAATTTGTTGCAACGAACA GGTCACTATCAGTC-3' as forward and reverse primers. These fragments contained wild-type or mutant lambda operator regions ( $O_L$  and  $O_R$ ), respectively. The distance between the midpoints of operator sites  $O_{L3}$  and  $O_{R3}$  was 393 bp. pDL965 contains CC to AT mutations in  $O_{L3}$  and  $O_{R3}$ , which abrogate CI binding [55]. PCR using the same plasmid templates was also used to generate 505 or 392 bp DNA fragments that contained only one group of binding sites ( $O_R$  or  $O_L$ ).

732 bp DNA fragment containing two high affinity lac operators *Oid* (5'-TGTGAGCGCTCACA-3') and *O1* (5'-AATTGTGAGCGGATAACAATT-3') [17, 18] separated by 70 bp was provided by Opher Gileadi (Quantomix Ltd, Rehovot, Israel). It was produced by PCR using the plasmid *pOid-O1* from the Müller-Hill laboratory as a template and 5'-GCCACCTCTGACTTAAGCGTCG-3' and 5'-TTGAGGGGACGTCGACAGTATC-3' as forward and reverse primers.

Another 1584 bp DNA fragment was cut from pBluescript plasmid with two restriction enzymes: Xma I and Ngo MIV (New England Biolabs, Ipswich, MA). This fragment does not contain any lambda CI sites.

The wild-type CI protein (7.25  $\mu\text{g}/\mu\text{l}$ ) was purified from pEA305 in the laboratory of Sankar Adhya. 20 nM CI and 2 to 4 nM DNA were gently mixed in a buffer containing 50 mM HEPES, 150 mM NaCl and 0.1 mM EDTA (pH 7.0) and incubated at RT for 10 min. Shortly before deposition, a 10  $\mu\text{l}$  drop of 0.1  $\mu\text{g}/\text{ml}$  poly-L-ornithine (1 kDa MW, product #P5666, Sigma-Aldrich, St. Louis, MO) was incubated on freshly cleaved mica for one minute at RT. The poly-L-ornithine-coated mica was then washed with 0.4 ml HPLC water and dried with compressed air. Then 5  $\mu\text{l}$  of the solution containing DNA and protein was quickly diluted with 40  $\mu\text{l}$  of buffer, and a 10  $\mu\text{l}$  droplet of this solution was deposited on the poly-L-ornithine-coated mica and incubated for one minute at RT. The droplet was rinsed away with 0.4 ml HPLC water and dried gently with compressed air. The sample was left overnight in a dessicator at RT before imaging.

In the study about loop probability, 50-200 nM his-tag wild-type lambda CI were incubated with 1 nM wild-type lambda DNA. 92  $\mu\text{M}$  his-tag CI were a kind gift from Keith Shearwin.

Images were acquired with a NanoScope MultiMode AFM microscope (Digital Instrument, Santa Barbara, CA) operated in tapping mode using a 50-60 mV oscillation amplitude of uncoated, etched silicon tips with a resonance frequency of

75 kHz (NSC18, MirkoMasch, San Jose, CA). Areas of  $1 \times 1 \mu\text{m}^2$  were scanned at a rate of 1.2 Hz and a resolution of  $512 \times 512$  pixels.<sup>2</sup>

AFM raw images cannot be analyzed directly for two reasons: first, because images are generated by scanning in successive lines, and there could be an offset between successive scan lines; second, because the piezoelectric motor response is not perfectly linear, the image surface is often bowing even if the sample surface is flat. After filtering, these two effects can be removed and DNA molecules may be interactively traced with NeuronJ [56], a plug-in function for ImageJ [57]. The volume of protein particles are measured with a basal threshold about 0.08 nm above the background. The base value in following measurement was then calculated as the mean value of all pixels below this threshold. For each isolated protein particles, the sum of the pixel heights above the base within the area of the particle protruding above the basal threshold was calculated as its volume. A second “DNA” threshold was selected just above the DNA to cut off the DNA from DNA bound protein particles. Therefore, only pixels protruding above the “DNA” threshold were considered as a part of the particles. The Matlab routine which performs this analysis can be found in appendix D and E.

### **§ 2.2.2 DNA contour length on mica surface**

In order to localize the specific location of CI binding on DNA, the position of the protein particles on the DNA revealed by the AFM images needs to be measured accurately. First of all, since the AFM images are obtained by scanning a tip over the

---

<sup>2</sup> This section was published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers.



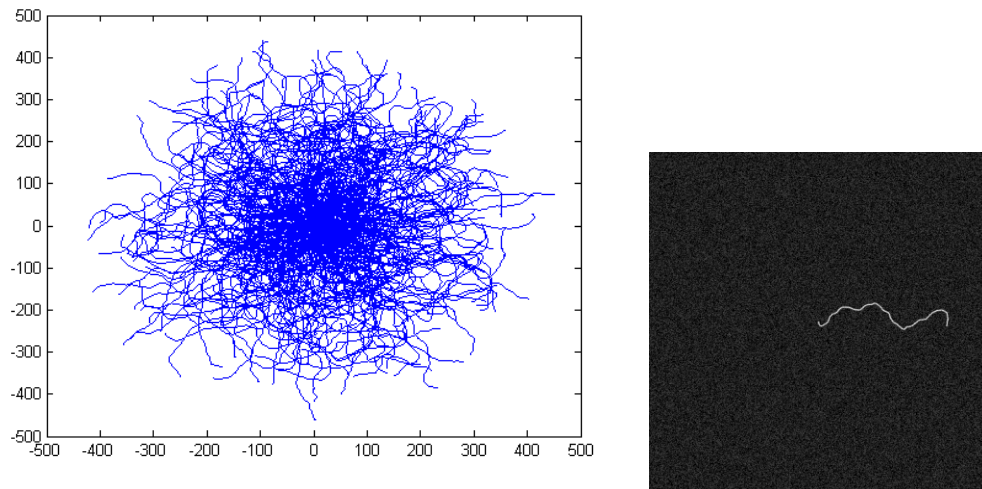
sample surface, the shape and size of the tip will smooth and enlarge the DNA fiber and make it appear wider. For the same reason, a DNA molecule that follows a zigzag contour might be smoothed during imaging and its overall contour length might be underestimated.

On the other hand, measured DNA contour lengths may be overestimated because of pixilation. The DNA fibers are recorded in AFM images as quantified pixels instead of continuous smooth curvatures. Therefore, some extra zigzags might be introduced and DNA contour length can be estimated in this process. Different DNA length estimators are available to balance the overestimation and underestimation factors due to the effects described above. [58-60].

Furthermore, the dried mica surface is very different from the natural aqueous cellular environment. It is suspected that DNA dried on mica may experience a partial transformation from B to A-form [58]. This conformational change would alone cause the DNA adsorbed and dried on mica to shorten since the A-form helix has a shorter helical pitch than that of the B helix.

Because of these considerations, simulated DNA polymer chains were used to evaluate the effect of tracing. A matlab routine was used to generate 300 polymer chains with two different persistence lengths (25 nm, 53 nm). Each polymer chain contains 1500 0.34 nm long segments, corresponding to 1500 bp B-form DNA. Then a virtual tip with a 2.7 nm radius end was used to scan the simulated DNA. The scanning signals were quantized into 512×512 pixel images and supplemented with random noise (Fig. 2.3). The final images were saved in tiff files which is the same format as that of real AFM images. The matlab codes of simulation (hundreds.m) and

image construction (imageG.m) can be found in appendix C. Then the images were traced and measured exactly the same way as normal AFM images.



*Figure 2.3:* 300 simulated 1500 bp polymer chain (left) and one example of simulated image of one polymer chain. Unlike the real DNA, the length of simulated polymer chains is well known and is not affected by the sample preparation. Tracing such polymer chains from simulated images can give an estimate of the error that is introduced by the tracing process.

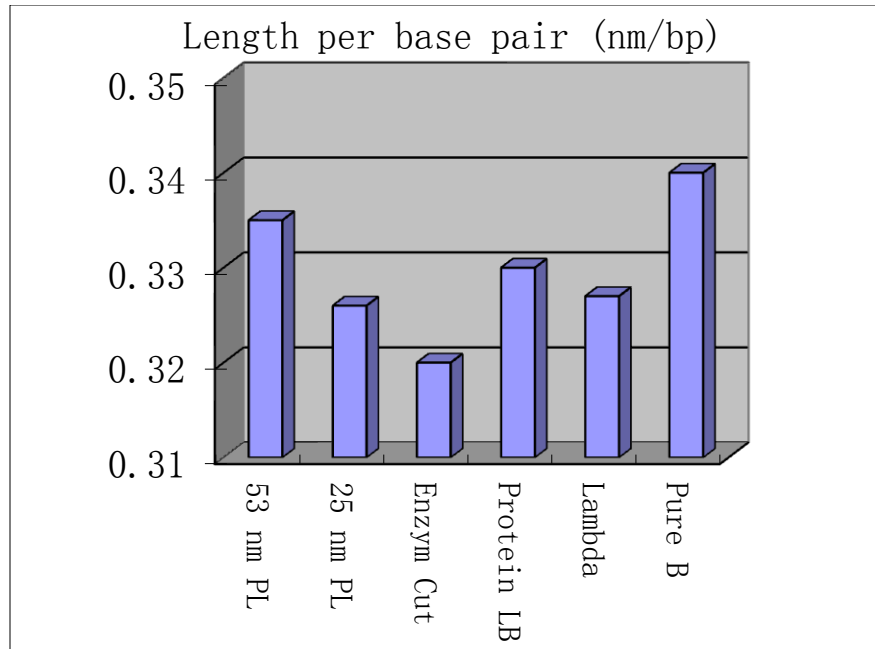
Finally, DNA segments with different number of basepairs were used to determine the exact ratio between length and the base pair rise.

## § 2.3. Result and discussion

### § 2.3.1 DNA contour length measured by AFM.

The base pair rise was measured from experimental or simulated images (fig.2.4). The simulated images show a decrease of measured contour length when DNA persistence length decreases from 53 nm to 25 nm. Because softer DNA, with a shorter persistence length, meanders more on the surface than a stiffer molecule, it

contains more bends that will be smoothed by the AFM tip. Therefore, it is not surprising that softer DNA will look shorter than stiffer DNA. The DNA fragment obtained by digestion with restriction enzymes gave a 0.322 nm/bp rise. While, measuring the distance between two protein particles sitting on two specific binding sites separated by 461 bp gave a rise of 0.33 nm/bp; 1555 bp long DNA, produced by PCR reaction, gave a rise of 0.327 nm/bp. All these values are 1.5-2.4% shorter than that found for the 53 nm persistence length simulation. Since the well accepted DNA persistence length in such condition is between 45 nm and 55 nm from different studies [61, 62], there might be some extra shortening of DNA rise per base pair. Some researcher attributed this part of shortening to partial B- to A- transformation because despite all the other effects, the measured DNA basepair rise is still shorter than pure B-DNA (0.34 nm/bp) [58, 63]). In summary, our measured DNA length is underestimated compared to the DNA in aqueous conditions.



*Figure 2.4:* Measured DNA basepair rise from simulated polymer chain and real DNA. From left to right: simulated polymer chain with 53 nm persistence length (0.335 nm/bp); simulated polymer chain with 25 nm persistence length (0.326 nm/bp); 1584 bp DNA cut by restriction enzyme (0.322 nm/bp); distance between protein particles binding on  $O_L$  and  $O_R$  site on  $O_3$ . lambda DNA, the two binding sites are separated by 461 bp (0.33 nm/bp); 1555 bp lambda DNA produced by PCR (0.327 nm/bp); B-DNA from crystal structure (0.34 nm/bp).

### § 2.3.2 Specific binding to operator sites.

In real experiments, DNA segments containing different numbers of binding sites are incubated with protein and imaged by AFM (fig. 2.5). And, the positions of CI particles along unlooped DNA were measured. Schematic diagrams of the molecules along with the positions of the right and left operator regions were showed in figure 2.6. The positions of the center of bound CI particles on DNA containing both wild-type operator regions were measured by tracing and put into histograms of frequency distributions (Fig. 2.5, upper center and left; Fig. 2.6, middle-left). The vast majority of particles centralized near the  $O_R$  and  $O_L$  regions 118 and 265 nm from one end of the molecules and non specific bindings were very rare.

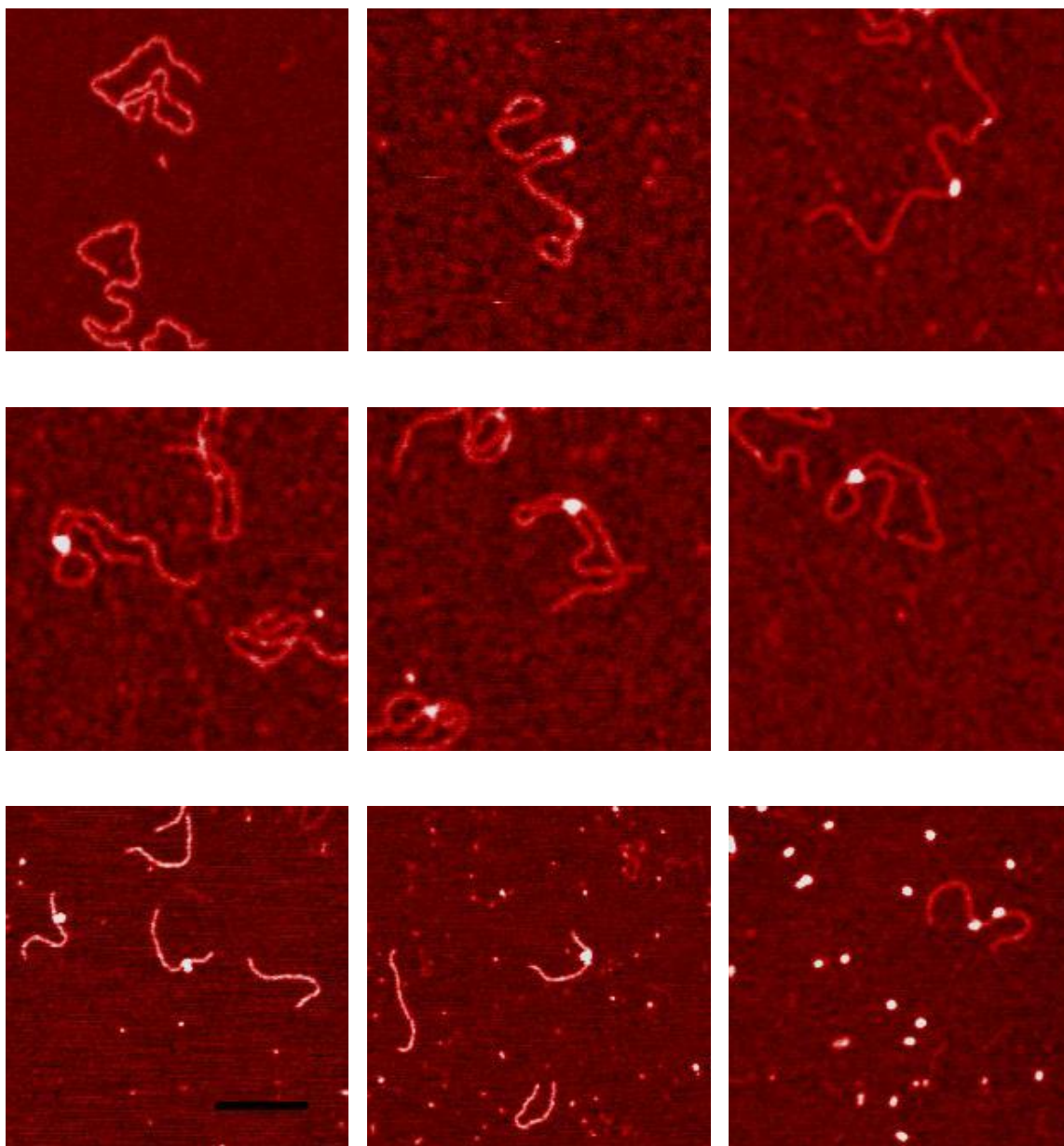


Figure 2.5: AFM images of  $\lambda$  CI and DNA: (upper left) 1555 bp DNA containing  $O_L$  and  $O_R$ , (upper center and right) CI protein bound to 1555 bp DNA, (middle row) CI-mediated loops in 1555 bp DNA, (bottom left) CI bound to DNA containing  $O_{L1,2&3}$  (wild type), (bottom center) CI protein bound to DNA containing  $O_{L1&2}$  ( $O_{3-}$ ), (bottom right) *lac* repressor bound to  $O_{id}$  and  $O_I$  containing DNA. The black bar represents 100 nm.<sup>3</sup>

<sup>3</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

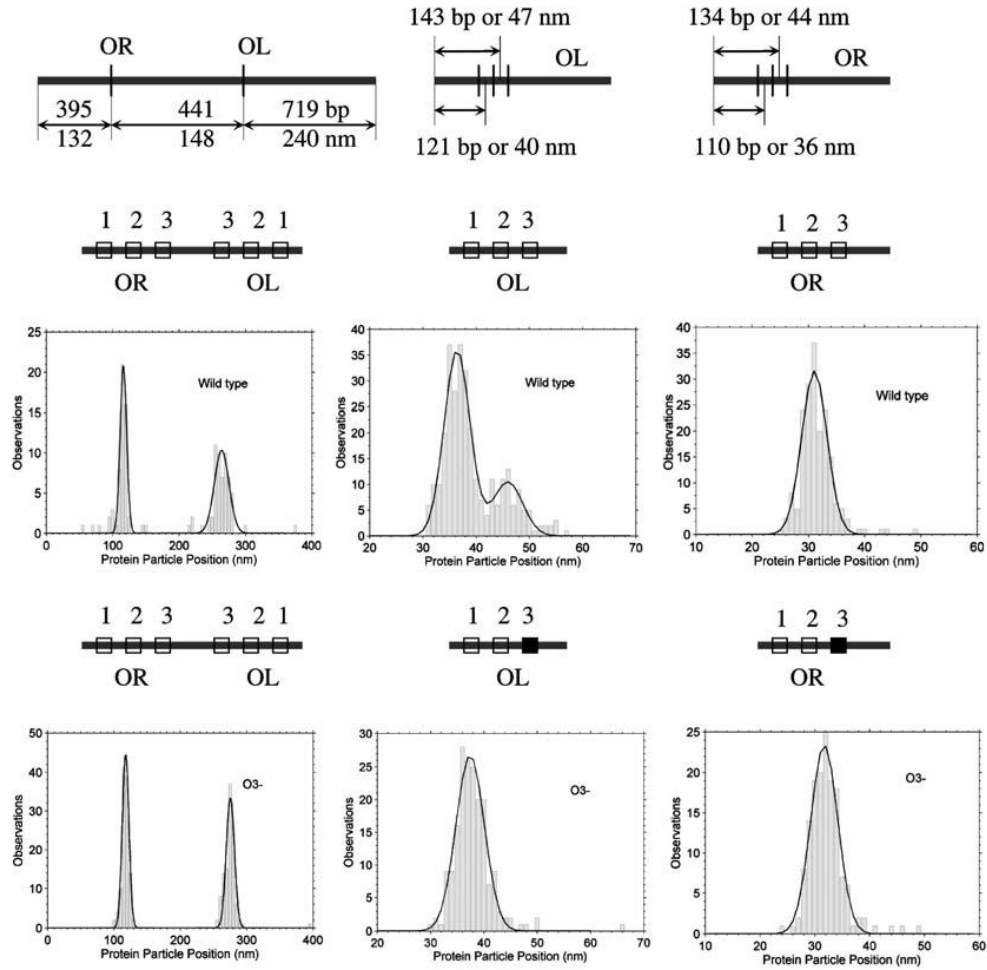


Figure 2.6: AFM measurements of the positions of CI particles bound to DNA. A schematic (upper) of the DNA construct with  $O_L$  and  $O_R$  operators. Histograms (lower) show the AFM measurements of the position of CI particles bound to different DNA fragments with wild-type and  $O_3$  operators as indicated.<sup>4</sup>

### § 2.3.3 Weak affinity for the $O_{R3}$ operator site

The noticeably broader peak at  $O_L$  is explained by cooperative binding of two CI dimers on adjacent operator sites; with consequent formation of tetramers occupying either operators 1 and 2 or 2 and 3. This is not likely to happen at the  $O_R$  region because the experimentally determined affinities of the operator sites [64] indicates

<sup>4</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

that the affinity of CI dimers for  $O_{R3}$  is much weaker than that for  $O_{R1}$  and  $O_{R2}$ . Experiments that abrogated the binding of CI dimers to  $O_{L3}$  and  $O_{R3}$  with DNA mutation ( $O_{3-}$ ) in the third binding sites supported this interpretation. Similar to the wild-type DNA, CI binding to the  $O_R$  region of  $O_{3-}$  DNA established a narrow peak at 119 nm (Fig. 2.5, bottom-left). In agreement with this interpretation is also the finding that the peak corresponding to the binding of CI to the  $O_L$  region of  $O_{3-}$  DNA shifted to produce a narrow peak at 275 nm, reflecting the disappearance of cooperative binding of CI to  $O_{L2}$  and  $O_{L3}$ .

Experiments with short fragments containing either  $O_R$  or  $O_L$  (Fig. 2.5, bottom left and center) were used to demonstrate the weak affinity for the  $O_{R3}$  site further. The histogram of particle locations on the wild-type  $O_L$ -containing fragment shows two peaks separated by 9.5 nm (Fig. 2.6, middle-center). This distance is slightly larger than the expected value for a tetramer bridging either sites  $O_{L1}$  and  $O_{L2}$  or  $O_{L2}$  and  $O_{L3}$  (20 bp or 6.7 nm). However, the peak which corresponds to the  $O_{L3}$  site (located at 47 nm), disappeared for DNA with the  $O_{L3-}$  mutation (Fig. 2.6, bottom-center) while the peak at  $O_R$  (32 nm) remained unchanged (Figure 2.6, compare middle-right and bottom-right). The simplest interpretation is that the occupancy of weak binding site  $O_{R3}$  does not significantly change with or without mutation while the strong  $O_{L3}$  binding was affected dramatically by a similar mutation.

#### **§2.3.4 Multiple operators may recruit dimers**

A few DNA molecules carrying small adjacent protein particles bound in positions that were commensurate with the  $O_I$  and  $O_3$  operator sites (Fig. 2.7) among the hundreds of molecules in the recorded topographs. The mean volume of these

particles was measured as  $174 \text{ nm}^3$ , which could be identified as CI oligomers of 2-4 monomers based on the calibration that was performed and is described below. According to the DNA construct, the center-to-center distance from  $O_{L1}$  to  $O_{L3}$  is 44 bp, corresponding to 14.7 nm, and 47 bp (15.7 nm) for  $O_{R1}$  to  $O_{R3}$ . Because the distance between pairs of adjacent particles was 15.4 and 14.0 nm for the  $O_R$  or  $O_L$  region respectively; the experiment revealed non-cooperative binding to the  $O_I$  and  $O_3$  sites. These experiments suggested that perhaps the presence of the third operator sites in each region can contribute in capturing CI dimers and thus help to secure a loop when a random collision between  $O_R$  and  $O_L$  occurs. However, it cannot be excluded that these species might be formed by broken looped molecules during deposition and washing in sample preparation.

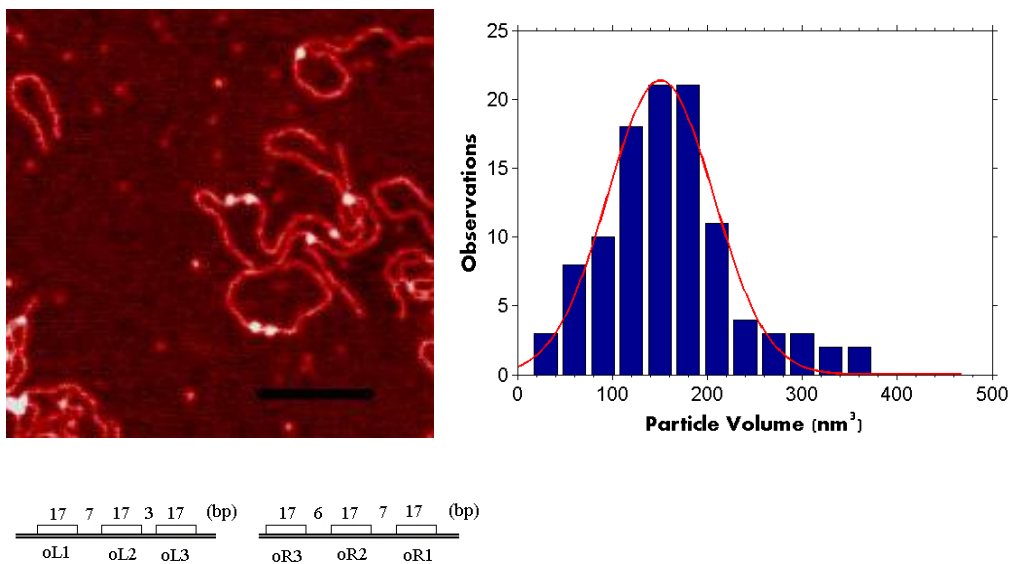


Figure 2.7: Pairs of CI particles bound to adjacent  $o1$  and  $o3$  sites were observed in AFM images (upper). The black bar represents 100 nm. (lower) The mean volume of these particles was  $174 \pm 70 \text{ nm}^3$ .<sup>5</sup>

<sup>5</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers



### § 2.3.5 Looping equilibrium

Indeed, the deposition process for protein-DNA complex binding to the surface was reported to affect the measured equilibrium by distorting the 3D topology [65]. The slight helical shift between the  $O_2$  and  $O_3$  operator sites might add some three-dimensionality to the loop structure. However 43.9 and 17.8% estimated looping probabilities were obtained from 884 and 354 molecules for wild-type and  $o3$ -DNA, respectively, at a 50 nM concentration (Table 2.1) by scoring as either “looped” or “unlooped”. The measured looping equilibrium suggested that the connection between molecular species in the AFM images and CI-mediated looping should be further characterized.

Table 2.1: Percentages of CI-mediated loops in wild-type and  $o3$ -DNA molecules visualized using AFM.<sup>6</sup>

	<i>Wild-type 10 min incubation</i>	<i>o3- 10 min incubation</i>
Number of molecules	884	354
% Looped	43.9%	17.6%

Looping percentage under different CI concentration was studied with his-tag CI protein as well. Figure 2.8 shows that the loop percentage increases according to [CI]. The loop percentage increased with [CI]. Since the his-tag CI concentration was given in monomer, the activity of his-tag protein is a little lower than normal CI (37.8% looping with 100 nM CI monomer compared to 43.9% looping with 50 nM CI dimer).

<sup>6</sup> This table was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

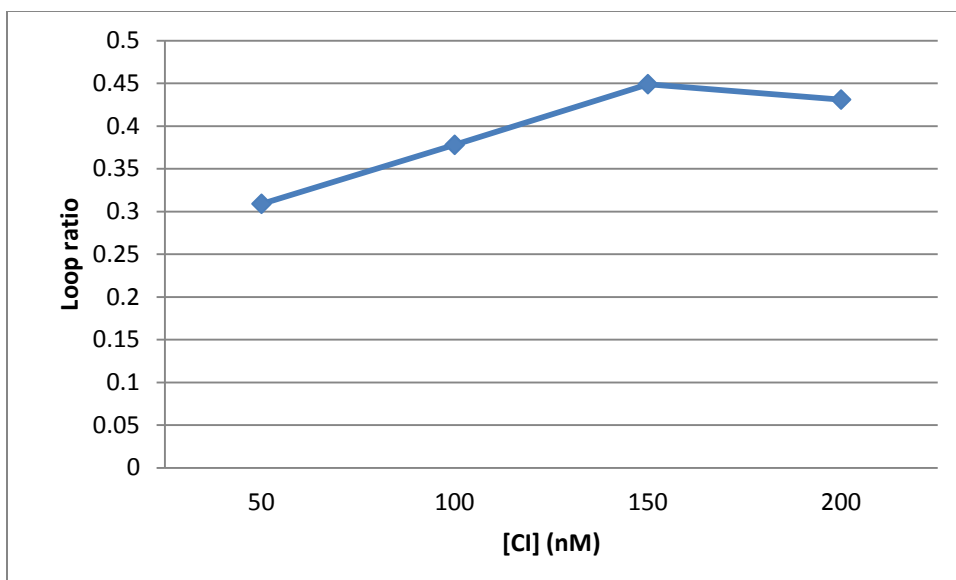


Figure 2.8: Loop ratio *under* different his-tag CI (monomer) concentration. The trend that loop percentage increases with CI concentration is proved by the curve.

### § 2.3.6 Volume calibration

Given the possibility for oligomerization of CI, the number of CI dimers securing a DNA loop may play an important role in the dynamics of loop formation. However, there are few experimental methods apart from direct visualization with which to determine this oligomerization on looped molecules. AFM is well suited for this type of analysis, since the volume of the particle at the closure of a DNA loop can be measured directly in the topographs. However, a calibration to relate the measured volume to the molecular weight, and hence the oligomerization, of the protein is essential.<sup>7</sup>

Several calibration curves have been produced previously for AFM images of proteins obtained in tapping mode both with silicon nitride [66] and etched silicon probes [67, 68]. Both the convolution of the probe shape and the compression that results from the tapping force affect the relationship, and linear fits to volume vs.

<sup>7</sup> This paragraph was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) by © 2009 Bentham Science Publishers

molecular weight calibrations have slopes ranging from 1.2 to 1.75 for probes with spring constants near 40 N/m and area thresholds set low or at half-height. For the experiments reported here, *lac* repressor (*lacI*) was a convenient reference which maintains a tetrameric state both free and bound to the DNA [69] while free CI was expected to partition into a 7:1 ratio of monomeric and dimeric forms at a concentration of 20 nM. The distributions of protein particles measured for CI and *lacI* without DNA exhibited peaks at 75, 150 and 320 nm<sup>3</sup> (figs 2.9, 2.10, 2.11). For the etched silicon probes with a 3.5 N/m spring constant that were used in these experiments, a calibration considering monomeric and dimeric CI and tetrameric *lacI* proteins deposited on poly-L-ornithine-coated-mica gave a slope of 1.9 (Fig 2.11). This higher value most likely reflects both the softer cantilever which reduces compression and the low threshold used to delimit the area of individual proteins.<sup>8</sup>

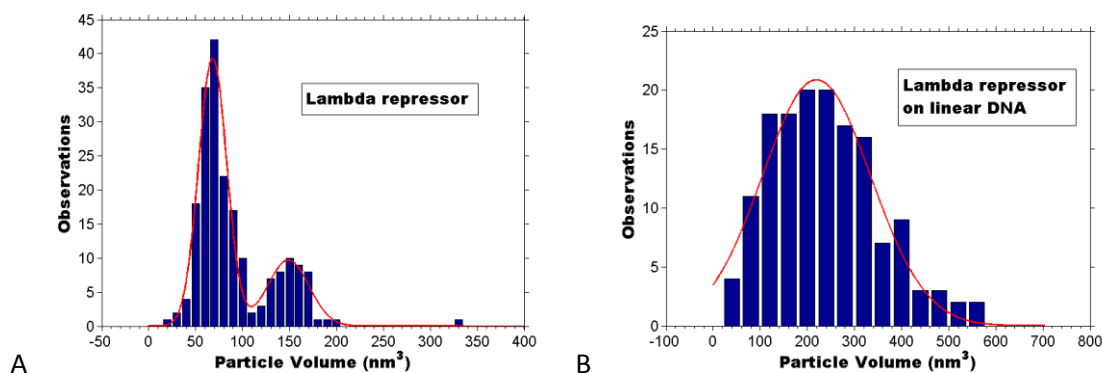


Figure 2.9: AFM measurements of the volume of CI protein particles free and bound to DNA. (A): volume of CI protein particles on mica surface. Fitting result disclosed that there are two peaks corresponding to CI monomer and dimer. (B): volume of CI protein particles binding on DNA. CI protein can only binding on DNA as dimers. Since two dimers binding on adjacent binding sites can interact with each other and thus stabilize each other, most of observed particles contain two dimers (or one tetramer).<sup>9</sup>

<sup>8</sup> The paragraph was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) by © 2009 Bentham Science Publishers

<sup>9</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) by © 2009 Bentham Science Publishers

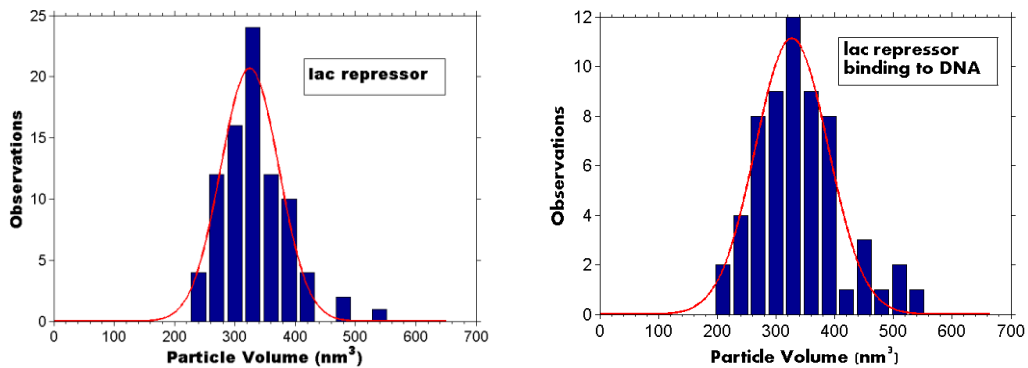


Fig 2.10: AFM measurements of the volume of lac repressor protein particles alone (upper) and bound to DNA (lower). Lac repressors form a stable tetramer in solution or binding on DNA.<sup>10</sup>

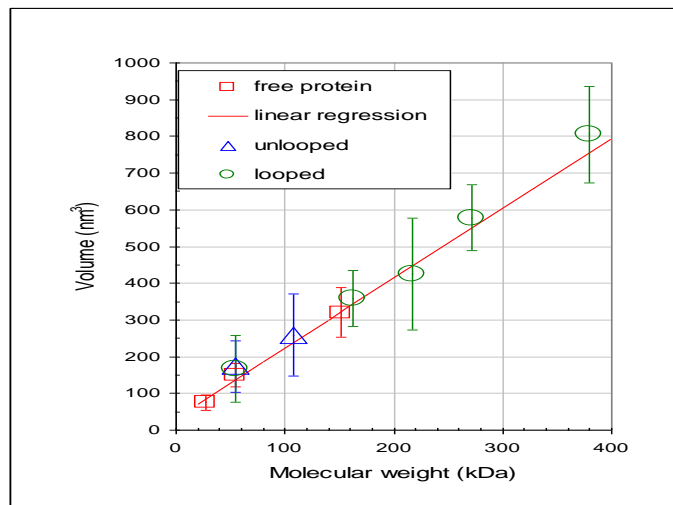


Figure 2.11: AFM measurements of the volume of protein particles both free and bound to DNA. Standard deviations are indicated for all points. Linear regression of volume measurements of unbound lambda and lac repressor proteins (red squares) gave the calibration line (red). The volumes of CI protein particles were measured on unlooped (blue triangles) and looped (green circles) DNA and CI oligomerization values were assigned to the nearest dimer multiple using the calibration line.<sup>11</sup>

<sup>10</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) by © 2009 Bentham Science Publishers

<sup>11</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) by © 2009 Bentham Science Publishers

The volumes of lacI and CI oligomers bound to DNA were also measured. The lacI DNA contains two *lac* repressor binding sites,  $O_{id}$  and  $O_I$ . The specificity of particle binding was verified by tracing DNA segments as described for the CI data shown in Figure 2.5. The average volume of particles binding on linear DNA was  $355 \pm 73 \text{ nm}^3$ . Since *lac* repressor was expected to remain tetrameric in the conditions of the experiment (5 nM) [69], this volume was associated with an oligomer weighing 155 kDa. The difference between the measured volumes for protein free and bound to the DNA was about 30 nm which corresponds well to the volume of a segment of DNA the length of the lacI binding site, 21 bp.

The average volume of CI particles on unlooped DNA measured  $259 \text{ nm}^3$ . Employing the calibration curve and considering that the molecular weight of CI monomer is 26-28 kDa [70, 71] indicated that the average particles in the experiment could have corresponded to CI tetramers ( $240 \text{ nm}^3$  from the calibration curve). Of course the standard deviation of these measurements was larger than those of *lac* repressor, because the  $\lambda$  operator regions contain three adjacent binding sites, so that several stoichiometries of CI binding were possible. In fact some higher molecular weight particles were observed that are difficult to reconcile with the idea that a looped DNA scaffold is required to promote “head-to-head” binding between CI tetramers to give octamers [72, 73]. One interpretation is that the specific binding nucleated adjacent non-specific binding.<sup>12</sup>

---

<sup>12</sup> This paragraph was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

### § 2.3.7 Loop closures prevalently contains nonspecific binding dimers

Similarly large volume, high molecular weight CI particles were commonly found securing looped DNA molecules. In figure 2.12, the lower panel shows measurements of DNA segments corresponding to the length: from one end to the  $O_R$  site, of the loop, and from  $O_L$  to the other end of the DNA. The narrowly distributed measurements and the good correspondence with the expected values based on the DNA construct indicated loops secured by specifically bound CI. The volumes of these CI particles were distributed as shown in the upper panel of figure 2.12. The red curve exhibits three central peaks in the distribution that roughly correspond to oligomers of (from right to left): 6-8, 10-12, and 14-16. This interpretation was developed using the calibration shown in figure 2.11 and assigning molecular weights to the nearest multiple of a dimer, since CI binds DNA as a dimer. The rightmost and leftmost peaks were negligibly small and were not considered further.<sup>13</sup>

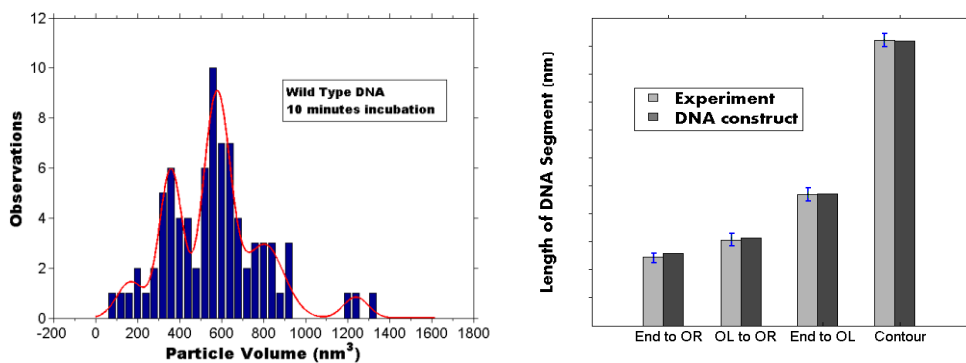


Figure 2.12: Oligomerization of CI securing DNA loops. (upper) AFM measurements of the volumes of single CI particles securing DNA loops. (lower) The lengths of segments in the looped DNA correspond well with those expected from the design of the construct.<sup>14</sup>

<sup>13</sup> This paragraph was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

<sup>14</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

Oligomers of 10-12 monomers were observed most frequently securing loop closures. Such oligomers would nearly or fully saturate the operator sites in the juxtaposed *oL* and *oR* regions and are consistent with the loop stabilization conferred by “ocamer+tetramer” protein binding found using modeling of tethered particle motion data [55]. A significant number of oligomers of 6-8 monomers were also observed at loop closures, but very little tetrameric CI, which corresponds well with the weaker loop stabilization afforded by these oligomers that probably lacked contacts between *o3* regions [55]. Oligomers of more than 12 monomers constituted a minor fraction which suggested that CI specifically bound to operators in one region might nucleate adjacent binding of non-specifically bound CI. These additional CI dimers might further stabilize the closure through interaction with corresponding dimers from the opposite region.<sup>15</sup>

Experiments of different CI concentration also disclosed that the average particle volume increase with the CI concentration (fig. 2.13). This result indicated that population of large protein-DNA complexes (10-12mer or higher oligomer) increase when CI concentration goes higher and higher.

---

<sup>15</sup> This paragraph was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) by © 2009 Bentham Science Publishers

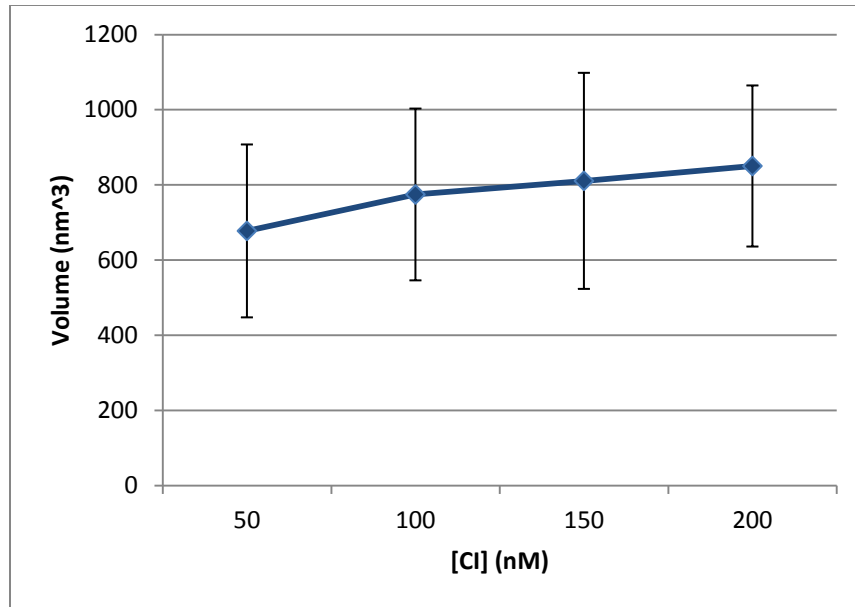


Figure 2.13: Volume of particles securing DNA loops under different his-tag CI concentration. The wide distribution of measured volume under each CI concentration indicates the particles may differ from each other by the number of dimers inside them. The average volume constantly increasing with [CI] tells the DNA loop can grab more CI dimers under higher CI concentration.

### § 2.3.8 Alternative loop closures

A small number of DNA loops (3.2%) contained two adjacent CI particles (Fig 2.14). The average volume of these particles was 425 nm<sup>3</sup> which identified them as CI octamers. By tracing the DNA in a subset of particularly distinct two-particle-loops (Tab. 2.2), two conformer types were established. One type was modeled with directly juxtaposed operators in which one octamer apparently included four specifically bound dimers at  $O_1$  and  $O_2$  (or  $O_2$  and  $O_3$ ), and another consisting of two specifically bound CI dimers at  $O_3$  (or  $O_1$ ) flanked by two non-specifically bound dimers to form a second octamer (Fig 2.14c). Whether non-specifically bound dimers preferentially flanked  $O_1$  or  $O_3$  could not be determined. The other type of conformers was modeled with staggered  $O_R$  and  $O_L$  regions leaving  $O_{R3}$  unoccupied (Fig 2.14b)



and CI oligomers bridging non-specific sites adjacent to  $O_{RI}$ . Table 2.2 shows the results of measuring segments in these looped molecules as schematically shown in Figure 2.14d. For such a small number of cases, statistically significant differences could not be established, but, as suggested by the schematic diagrams, segments  $a$  and  $e$  were longer in the directly juxtaposed conformation while  $c$  was longer in the staggered conformation. These few conformers might represent early intermediates in the looping process that result from collisions between  $O_L$  and  $O_R$  regions that are nearly saturated with CI dimers. Such intermediates may include CI tetramers that bind “semi-specifically” between  $O_{LI}$  and a non-specific site adjacent to  $O_{RI}$ . Subsequent shifting to create complete juxtaposition of all of the specific operators would be expected to increase the stability of the loop and sterically repress the CI promoter,  $P_{RM}$ , near  $O_{R3}$ .<sup>16</sup>

---

<sup>16</sup> This paragraph was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers.

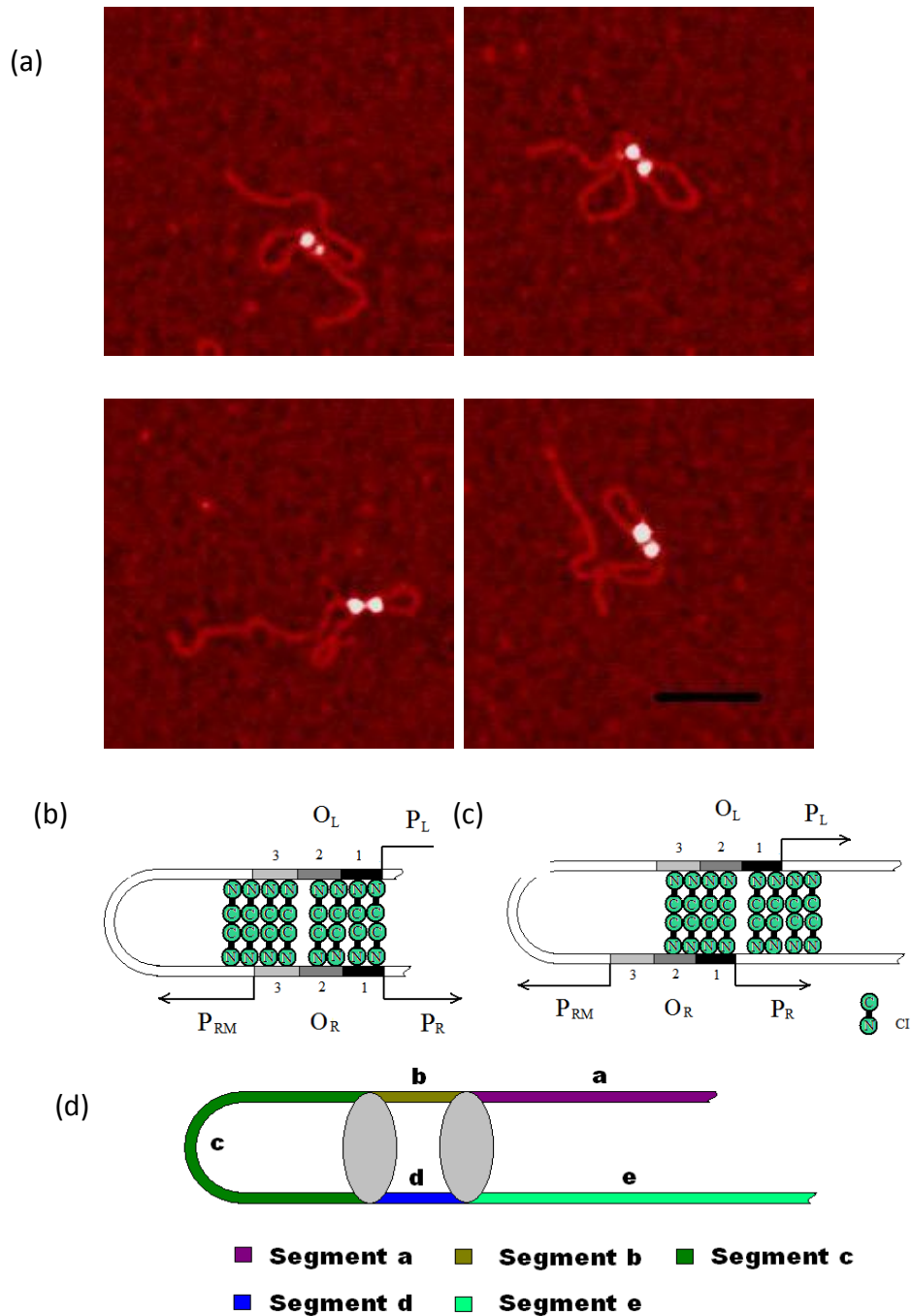


Figure 2.14: Specifically bound protein particles may nucleate adjacent semi-specific binding to secure DNA loops. (a) A small number of DNA loops were secured by two CI particles. Possible CI binding to (b) directly juxtaposed or (c) staggered *OL* and *OR* regions. (d) Labeled segments of looped DNA molecules secured by two CI particles. Scale bar represents 100 nm.<sup>17</sup>

<sup>17</sup> This figure was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

Table 2.2: Segment lengths (nm) for DNA loops secured by two protein particles (shown in Fig 6).<sup>18</sup>

Segment DNA molecule	a	b	c	d	e
<b>Directly juxtaposed operators</b>					
<i>expected</i>	129.0	14.0	125.7	13.0	237.5
<b>1</b>	126.8	20.2	113.7	19.6	233.4
<b>2</b>	127.1	14.1	123.2	16.5	238.4
<b>3</b>	125.2	11.7	113.0	13.6	221.8
<b>4</b>	125.3	20.0	117.9	17.7	222.6
<b>5</b>	124.0	19.5	104.3	20.4	231.2
<i>mean</i>	<b>125.7</b>	<b>17.1</b>	<b>114.4</b>	<b>17.6</b>	<b>229.5</b>
<b>Staggered operators</b>					
<i>expected</i>	116.8	14.7	142.2	14.7	230.0
<b>6</b>	116.1	18.5	123.7	15.9	223.1
<b>7</b>	119.3	16.5	130.9	18.0	231.1

### §2.3.9 Conclusions

The study described above supports the idea that CI binding to  $O_3$  operators greatly stabilizes looping of  $\lambda$  DNA fragments. Overwhelmingly specific binding was exhibited by 50 nM CI protein to the  $\lambda$  operator sites. The intrinsic order of this binding,  $O_{LI} > O_{RI} > O_{L3} > O_{L2} > O_{R2} > O_{R3}$ , [74, 75] changes to  $O_{RI} \sim O_{LI} \sim O_{R2} \sim O_{L2} > O_{L3} > O_{R3}$  when cooperative interactions are considered, and this cooperative ranking was reflected in measurements of the positions measured for CI particles on unlooped DNA that shifted slightly upon mutation of the  $O_{L3}$  but not the  $O_{R3}$  operators. Measuring protein particle volumes with AFM probes having small spring constants reduced the compression of protein particles reported by others to give an accurate calibration that facilitated the analysis of CI oligomers securing DNA loops. In

<sup>18</sup> This table was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10, P 494-501) by © 2009 Bentham Science Publishers

addition, the strong affinity of the polyamine-coated mica for DNA preserved the looped-unlooped equilibrium of the DNA-protein complexes. Volume measurements of these protein particles showed that DNA loops were stabilized most frequently by CI oligomers of 10-12, less frequently by oligomers of 6-8 and occasionally by oligomers of 14-16 that likely include non-specifically bound CI. This underscores the important role of the  $O_3$  binding sites in loop stabilization. Finally, rare observations of dimers bound to adjacent operators, and adjacent CI octamers securing specific loops suggest that the tripartite binding sites in the operator regions enhance the targeting of CI to promote efficient looping and transcriptional repression at low protein concentrations.<sup>19</sup>

This work was published [76] in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) and a reprint of the paper is reported in Appendix A. Results of this work is also included by a book chapter (appendix B).

---

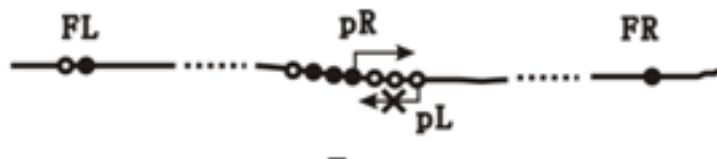
<sup>19</sup> This paragraph was first published in *Current Pharmaceutical Biotechnology* (2009, Vol. 10,P 494-501) by © 2009 Bentham Science Publishers

## **Chapter 3**

# **AFM and TPM Study of DNA Wrapping and Looping of Phage 186**

### § 3.1 Background and introduction

Coliphage 186 is a counterpoint to inducible phage  $\lambda$  because the two phages are not evolutionally related but developed similar life cycles in evolution [31, 42]. The hypothesis that 186 repressor (or 186 CI) can assemble into wheels of 14 monomers around which DNA partially or fully wraps may explain many *in vivo* experimental results including how 186 CI regulates both positively and negatively its own transcription [41]. A schematic drawing of the 186 major control region is showed in fig. 3.1. According to this model, the 186 CI wheel will first bind to the strongest binding site pR at low concentration to turn off all the lytic transcriptions. Repression of pR will permit transcription from pL (which leads to 186 CI) by inhibiting transcriptional interference between pR and pL. However, pL transcription cannot easily occur if the CI wheel is bound at pR. This is because pL is only about 60 bp from pR and the wheel contains seven dimers, one of which can occupy pL and repress it [31, 43]. This problem can be alleviated by the two flanking site FL and FR. When the CI concentration is low, these two flanking site can compete with pL by interacting with the 186 wheel bound at pR inducing a loop in the DNA. In this case, pL will be left unoccupied and free for RNAP to bind [31].



*Figure 3.1:* Three groups of binding sites are involved in the regulation of the genetic switch between transcription of the lysogenic promoter (pL, production of 186 CI repressor) and that of the lytic promoter (pR).

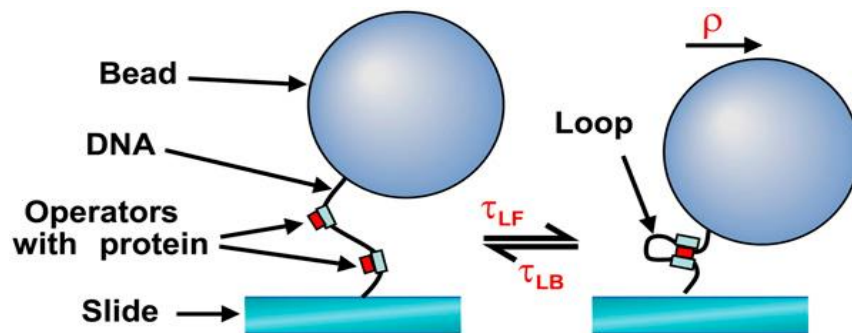
In response to a severe DNA damage, LexA, a repressor, will be removed so that DNA polymerase will be produced to repair the damage. However, the removal of LexA will also release the transcription of Tum, a phage protein. Since Tum protein is an antirepressor that can prevent 186 CI to bind, the pR will be derepressed and the prophage will irreversibly undertake a lytic response [77].

The efficient switching from the lysogenic state to lytic state requires a well regulated CI concentration so that a little amount of Tum protein can remove all possible CI binding on pR region. In prophage 186, when the CI protein concentration is too high, FL and FR will be occupied by different CI particles. In this case, FL and FR can no longer compete with pL for the wheel bound at pR, so pL will be occupied by a free dimer in the wheel at pR. As a consequence, transcription of CI will be turned off and CI concentration will be kept at a level such that not only the lysogenic state can be maintained but also the phage can efficiently switch to a lytic response if needed [31].

However, there was no direct evidence supporting the existence of wheel-like particles of 186 CI before. Furthermore, analytical ultracentrifugation-sedimentation experiments showed that 186 CI monomer can form dimers, tetramers to octamers but not 14mer [42]. Therefore, the structure of the 186 CI-DNA nucleoprotein complexes needs to be characterized by AFM and tethered particle microscopy (TPM) in order to, then, understand the mechanism of the 186 epigenetic switch. For a brief description of AFM, please go to § 2.1, page 13.

Tethered particle motion (TPM) was first described in 1991 [78] (Fig. 3.2). In this technique, microbeads are tethered to the surface by polymer chains such as DNA.

Therefore, their Brownian motion is limited by the tether. Once the protein or other factors interact with the DNA and shorten its length by looping or wrapping, the Brownian motion range of the beads will become smaller. Using optical microscopy, and tracking the motion of the microbeads, the formation-breakdown of DNA loops and wraps can be observed [25, 79].



*Figure 3.2:* Schematic drawing of TPM [27]. Micro beads are tethered on the surface by DNA. Once protein interacts with the DNA by looping or wrapping, the tether length will be shorted and the Brownian motion range of beads will decrease. Therefore, DNA shortening can be measured by observation of beads Brownian motion with a microscope.

## §3.2 Material and method

### §3.2.1 AFM sample preparation.

1584 bp-long DNA fragments were produced by cutting plasmids derived from pBluescript containing wild type 186 operators (FL, FR, pR, pL) with two restriction enzymes: NgoMIV and XmaI (New England BioLabs). The digestion product was isolated and purified (QIAGEN gel purification kit). The position of the midpoint of each operator from one end was: 178 bp/56.7 nm (FL), 484 bp/154.9 nm (baricenter



of pR. In particular, 463 bp/148.2 nm (pR1), 484 bp/154.9 nm (pR2), 505 bp/161.6 nm (pR 3)), 567 bp/181.4 nm (pL) and 857 bp/274.2 nm (FR).

The following forward and reverse primers were used to amplify various DNA fragments as follows: 5'-TTACCGGAGAAGGAGAAGCA-3' and 5'-ATTAATG CAGCTGGCACGAC-3' (524 bp-long DNA containing only FL), and Biotin5'-CTTTCTTGCAGCCTTTACGG-3' and 5'-TTTACAAATGCTTCTCCTTCTCC-3' (528 bp-long DNA containing just pR and pL).

Wild-type 186 CI repressor was prepared and purified as described previously [80]. The protein was diluted to the desired final concentration (5 nM, 50 and 100 nM) in the presence of 1 nM DNA in a buffer containing 50 mM HEPES, 150 mM NaCl and 0.1 mM EDTA (pH 7.0). All steps were conducted at  $T_{\text{room}}$ . The mixture was incubated for 20 min. The biotin-labelled DNA fragment was incubated in a mixture containing also 1  $\mu\text{g/ml}$  streptavidin. Shortly before deposition, a 10  $\mu\text{l}$  drop of 0.01  $\mu\text{g/ml}$  poly-L-ornithine (1 kDa MW, Sigma-Aldrich, St. Louis, MO) was incubated on freshly cleaved mica for one minute. The poly-L-ornithine-coated mica was then washed with 0.4 ml HPLC water and dried with compressed air. Then, 10  $\mu\text{l}$  of the solution containing DNA and protein were deposited on the poly-L-ornithine-coated mica and incubated for one minute. The droplet was rinsed with 0.4 ml HPLC water and dried gently with compressed air. The sample was left overnight in a desiccator before imaging.

Images were acquired with a NanoScope MultiMode AFM microscope (Digital Instrument, Santa Barbara, CA) operated in tapping mode using uncoated, etched silicon tips (MirkoMasch, San Jose, CA). The oscillation amplitude was 50-60 mV

with a resonance frequency of 75 kHz (NSC18, MirkoMasch, San Jose, CA). Areas of  $1 \times 1 \mu\text{m}^2$  were scanned at a rate of 1.2 Hz and with a resolution of  $512 \times 512$  pixels. After filtering images to remove scan line offsets and bowing, DNA molecules were interactively traced with NeuronJ [81], a plug-in function for ImageJ [57].

### § 3.2.2 TPM sample preparation:

The following primers: 5' TCC AGA GGC GCC GGG GGG TTC GTG CAC ACA G and 5'TGGTAACCTAGGTAAACAAATAGGGGTTCGCGCAC were used to amplify by PCR the 186 region contained in pBluescript. pDL611[25] and the PCR product were then digested with EcoR1 and Pst1 in order to insert by ligation the 186 region from pBluescript into pDL611. The preparation of pDL 611 fragment contains 186 relevant fragment was done by Chiara Zurla in our group. The final 1898 bp-long wt or mutated TPM tether was obtained by PCR using this modified plasmid and the following 5' end biotin and digoxigenin-labeled oligos:

Biotin-5'-CGCAATTAATGTGAGTTAGCTCACTCATTAGGCACCCCAGGC-3' and dig-5'GCATTGCTTATCAATTTGTTGCAACGAACAGGTCACTATCAGTC-3'

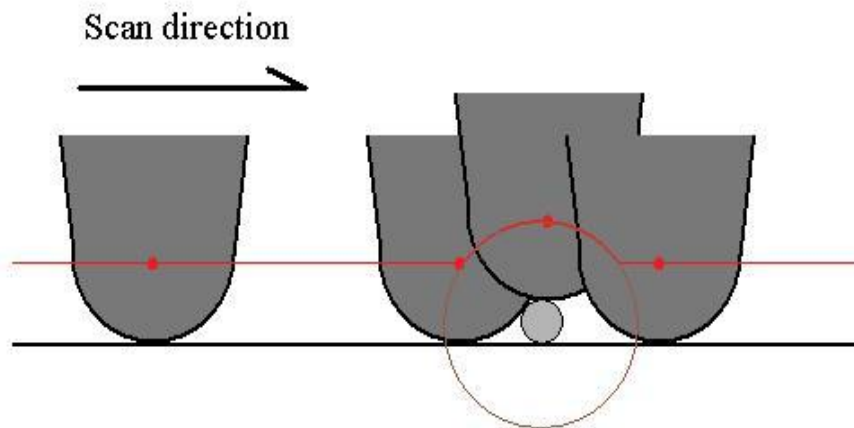
The FL- or FR- DNA fragments contained mutated FL or FR operators to prevent CI binding. In  $\Delta$ pR DNA the region containing the pR binding sites was replaced with an equally long, but unrelated DNA.

The TPM microchamber and experiment were prepared and run as previously described [82-84]. In brief, the glass surface of a microscope flowchamber was coated with biotin-BSA and incubated with streptavidin. DNA tethers were labeled with anti-

digoxigenin-coated beads with a diameter of 0.48  $\mu\text{m}$  (Indicia Diagnostics, Oullins, France). Interaction of the 186 CI protein with DNA was monitored as a reduction in the amplitude of the Brownian motion of the bead as previously described [79, 82, 85, 86].

### §3.2.3 Measurement of wheel diameter

When the SFM probe crosses over an object on the surface, it will be raised up by the object. The trajectory of the probe is decided by the curvature of the object and the probe (Fig. 3.3). The manufacturer only tells that the radius of the tip is less than 10 nm, therefore the probe size needs to be estimated with other method before experiments.



*Figure 3.3:* AFM tip scan through a DNA. The dark gray blob represents an AFM tip. When the tip scanning through a DNA (light gray circle), the movement of the tip will be recorded by AFM (red curve). This curve is a mixture of tip effect and the diameter of DNA cross section.

If one looks at the bottom of the tip as a sphere, when the probe is scanning cross rod like DNA, the trajectory of the sphere center will be a mixture of straight lines

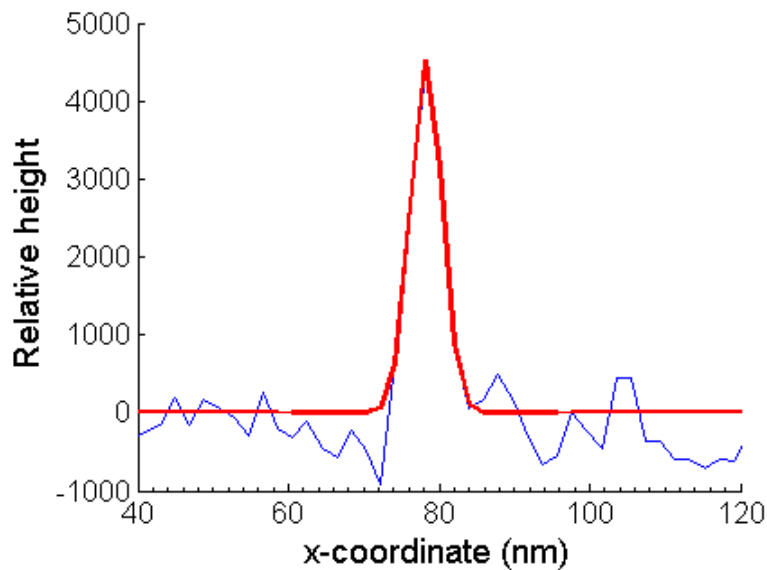
and a circle as showed in figure 3.3. The radius of the red circle is the sum of the sphere radius and the radius of the DNA. The height of the trajectory is equal to the DNA diameter.

Let the radius of the tip be  $R$  and the radius of the DNA be  $r$ , then the relationship between DNA half-height-width and the radius of the tip and DNA can be derived from figure 3.3 with a method similar to Miller's [87]:

$$(R+r)^2 - R^2 = (W/2)^2$$

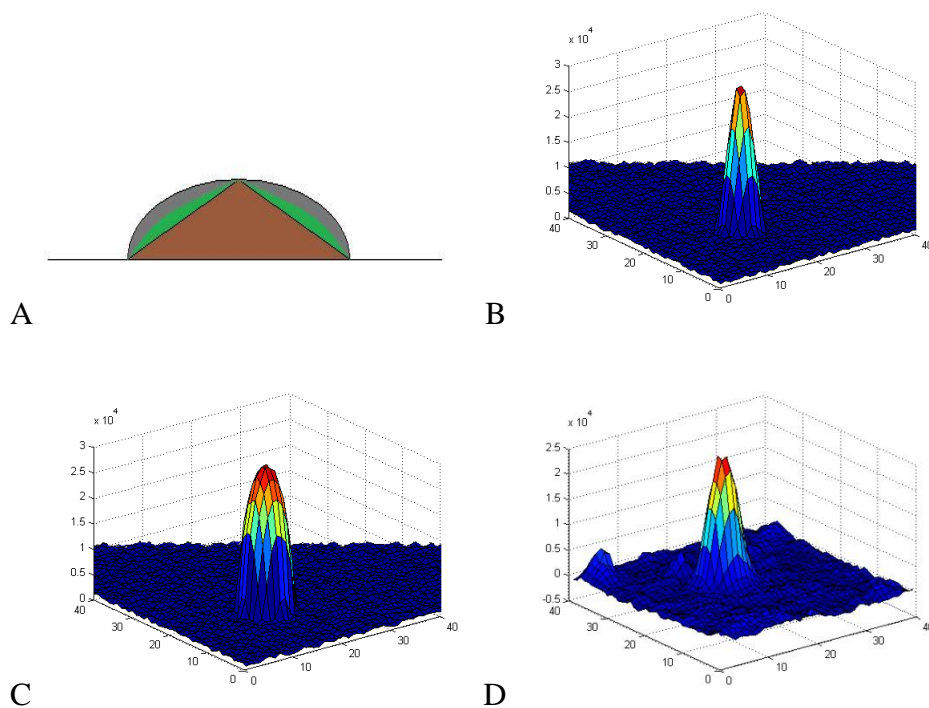
Where  $W$  is half-height-width of the peak of a DNA.

The half-height-width is 5.06 nm which is calculated by fitting the image data of a DNA cross section (fig. 3.4) with a Gaussian function. Using this equation, the radius of a typical AFM probe can be estimated as: 2.7 nm.



*Figure 3.4:* A cross section of the image of a DNA molecule obtained by AFM. The width of the center peak is much larger than the DNA diameter because of the effect of the tip. The half-height-width is 5.06 nm which is calculated by fitting the data with a Gaussian function.

The effect of the AFM tip is not only decided by the tip size but also by the shape of the particle itself. Using the calculated diameter of the AFM tip, two different types of particles are simulated. Figure 3.5A shows the cross section of a particle with a half ellipsoid shape (showed in gray) and a conic shape particle (brown). Comparing the 3-D AFM-revealed topography of real protein particles (fig 3.5D) with the two simulated particles, the shape of real particles can be found in between ellipsoid and conic shape (showed in green). Therefore, the simulated particle of two shapes can serve as upper and lower boundary for the estimation of real particles.



*Figure 3.5:* (A) Cross section of different shape particles: half ellipsoid (gray) and conic shape (brown). The real particle is expected in between of the two shapes (green) (B) virtual 3-D topography of a conic shape particle scanned by AFM tip. (C) virtual 3-D topography of a half ellipsoid particle scanned by AFM tip. (D) AFM topography of a real particle of 186 repressor.

The diameter of CTD wheel is 10.2 nm according to X-ray crystallography [41] (fig. 1.4). If the NTDs were added to the CTD wheel, the diameter should be estimated as 15 nm. Therefore, 7.1 nm radius conic and half-ellipsoid particles were simulated and scanned by a fake AFM tip. The simulated 3-D topology shows that the cross section at 17% of particle height of virtual scanned topography has the same diameter compare to original particle for conic shape particles. For half ellipsoid particles, this cross section appeared at 45% of particle height. Therefore, the cross section at 31% particle height (average of 17% and 45%) was believed to best present the real size of particles in experiment. Since the average particle height measured from AFM is 3.5 nm, the cross section should be 1.1 nm high. In order to get the error of particle diameter estimation, this 1.1 nm threshold is applied on conic particle and half ellipsoid particle respectively. The cross section diameter is 10.8 nm for the conic particle and 16.1 nm for the half ellipsoid. Given the particle diameter is 14.2 nm, the error of the diameter measurement should be about  $\pm 3$  nm.

### **§ 3.3. Result and discussion**

#### **§ 3.3.1 Confirmation of basic model**

##### **§ 3.3.1.1 The 186 repressor wheel and its assembling.**

The 186 bacteriophage repressor, 186 CI, binds to DNA as a dimer, and it was suggested to assemble into oligomers of dimer, tetramer and octamer in solution [88]. A more recent crystallographic study showed that the CTD of 186 CI assembles into a wheel of seven dimers (hereafter referred to as the 186 heptamer) [89] (Fig. 1.4, 3.6). This led to the hypothesis that the whole protein, including the NTD DNA-binding

domain, may too form wheel-shaped heptamers. However, the existence of 186 heptamer was not supported by the study of sedimentation equilibrium [80] although it can provide a good explanation of 186 genetic switch [90]. Therefore, AFM was used to image 186 CI free, as well as bound to 1584 bp-long DNA fragments to characterize its shape and dimension *in vitro*. The image data were analyzed with matlab program discussed in §4.3.4.

The diameter of 186 CTD 14mer wheel and the length of one 186 NTD can be measured from protein data bank structure (10.2 nm for 14mer and 2.4 nm for one NTD, Fig. 3.6). Therefore, the diameter of a whole 186 wheel can be estimated around 15 nm.

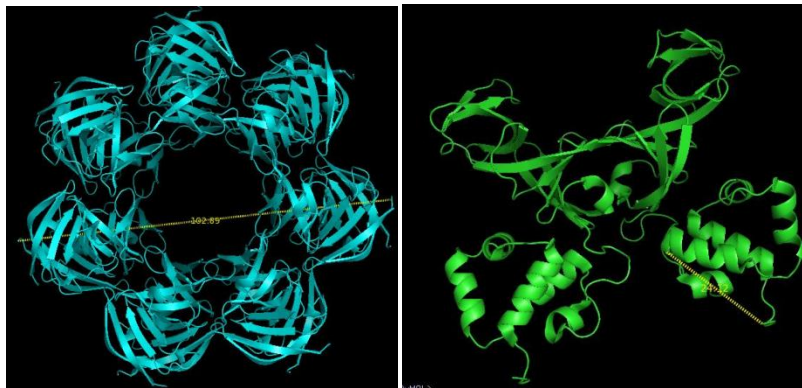
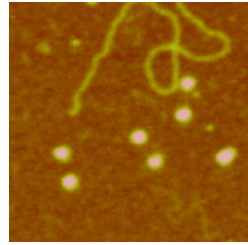
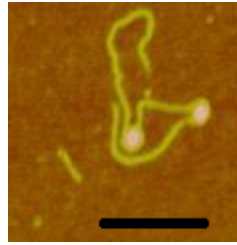


Figure 3.6: X-ray crystal structure of 14mer wheel of 186 CTD (left, PDB ID: 2FKD) and of 186 dimer (right, PDB ID: 2FJR). The CTDs of 186 repressor can interact with each other and form wheels contain seven dimers each.

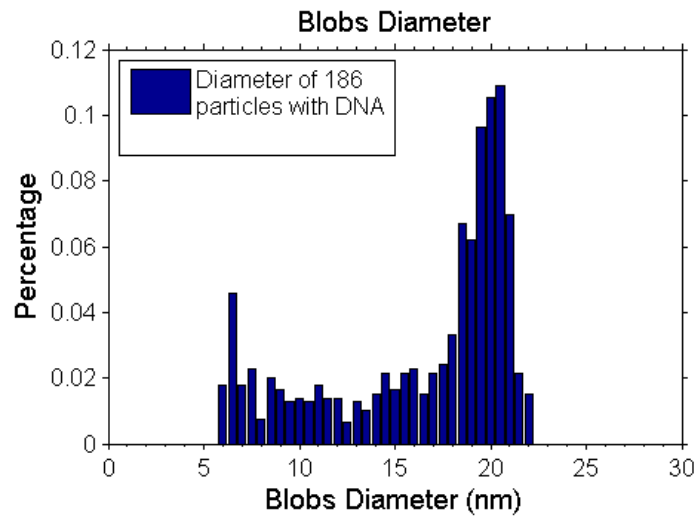
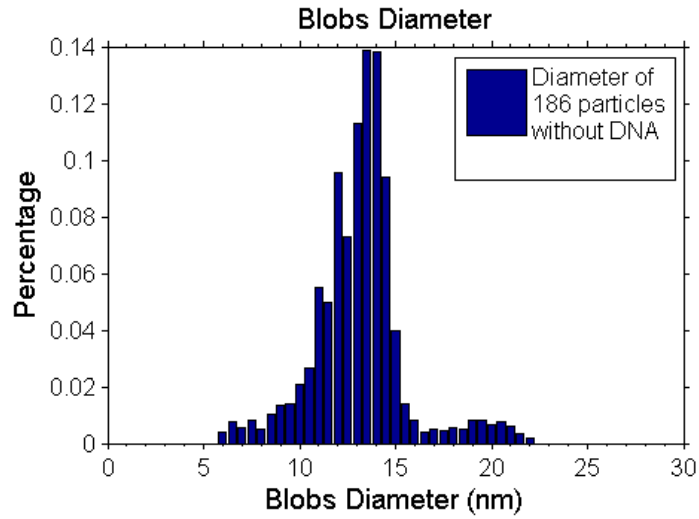
The results, summarized in figure 3.7 strongly support the idea that the protein oligomerizes to form wheel-shaped heptamers. Furthermore, comparison of diameter with or without DNA disclosed that assembling of 186 heptamer needs facility of DNA.



Unbound particles



Bound particles



*Figure 3.7:* First row: left, mutated 186 repressor with wild type DNA; right, wild type 186 repressor with wild type DNA, most of particles binding onto the DNA. Second row, the histogram of particle diameter of mutated 186 repressor. Last row, the histogram of particle diameter of wild type 186 repressor and wild type DNA.



The left image in the first row of figure 3.7 was obtained with 186 repressors carrying a mutation on NTD. This mutated protein cannot bind to DNA but are still able to interact with each other because protein-protein interaction relies on CTD only [42, 91]. The histogram of diameter was made from 6632 particles. The main peak of the histogram located at  $13.3 \pm 2.0$  nm. This peak is obviously smaller than 186 heptamer wheels and agrees with the study of sedimentation equilibrium [80] that 186 CI exist in solution mainly in dimer-tetramer-octamer but not higher oligomers.

However, the histogram contains a tiny tail on the right side of main peak. The center of this tail can be fitted out at  $19.4 \pm 2.4$  nm, which can be possibly addressed to 186 heptamer. On the other hand, if wild type 186 CI was incubated with its DNA (Fig. 3.7, upper right), the main peak of the diameter histogram (987 particles) shifts significantly to  $19.8 \pm 1.7$  nm and most of the particles were found on DNA. This result clearly showed that interaction between protein and DNA can significantly help the protein assembling to high order oligomers.

Furthermore, the volume of the big wheels imaged by AFM was measured and compared to a calibration curve previously obtained [76] (Figure 3.8). This volume analysis is consistent with the idea that the wheels may be composed of seven dimers. Finally, since such wheels are very abundant in the images obtained using only 50 nM CI, which is a much lower than the 1100 nM estimated for the lysogen, it is likely that 186 CI associates into a heptamer at an early stage after infection and that this state of assembly is robust through the host cell division.

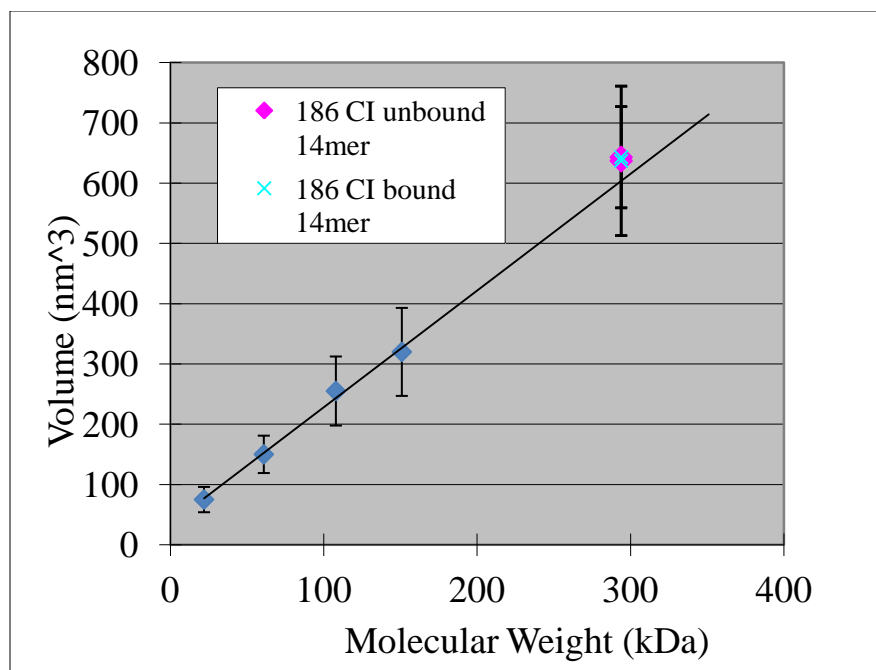


Figure 3.8: AFM measurement of CI volume. The particle volume of wheel-like particles measured by AFM (pink diamond and cyan cross) falls on the calibration curve of volume vs. molecular weight obtained using from left to right : lambda CI monomer (25 kD), lambda CI dimer(50 kD), nucleosome (108 kD) and lac repressor(150 kD) (blue diamond) [76].

In conclusion, the wheel-like particles of 186 repressors are observed both on the surface and the DNA directly. The volume and size of parts of the bind and unbound 186 particles are measured larger than an octamer and close to a 186 heptamer wheel under physiological concentration. Since the ability of 186 repressor to form a 186 heptamer wheel was approved by X-ray crystal structure and comparing to other high order multimers, the wheel-like 186 heptamer has some kind of advantage because the extra protein-protein interaction inside the wheel, the research strongly support that 186 repressor interact with DNA as a wheel-like 14mer particle under physiological condition .

The shift of histogram peak disclosed that the interaction between 186 repressor and DNA can help protein assembling into high order oligomers. There are too possible pathways for this kind of facility. The DNA can either shift the oligomerization equilibrium to the right side by grabbing 186 heptamer wheels from solution or assemble heptamer wheels around the specific binding sites directly. No matter what mechanism is preferred, this experimental result perfectly connects the gap between the study of sedimentation equilibrium [80] and 186 mechanism [90].

#### **§ 3.3.1.2 CI regulatory mechanism**

A 186 heptamer may bind cooperatively to multiple operators [89, 91, 92], giving rise to physiologically relevant nucleoprotein complexes with different structure and conformation, and with different impact on the 186 transcriptional regulatory network. Indeed, the fact that lysogeny maintenance requires repression of pR and tight control of transcription from pL, and that pR and pL face one another, suggests that different nucleoprotein species may be in equilibrium in different repressor concentration regimes, so that the probability of pL being unoccupied decreases with increasing CI concentration [92]. Figure 3.9 shows the possible species and equilibria that have been suggested, together with AFM images confirming the existence of these complexes.

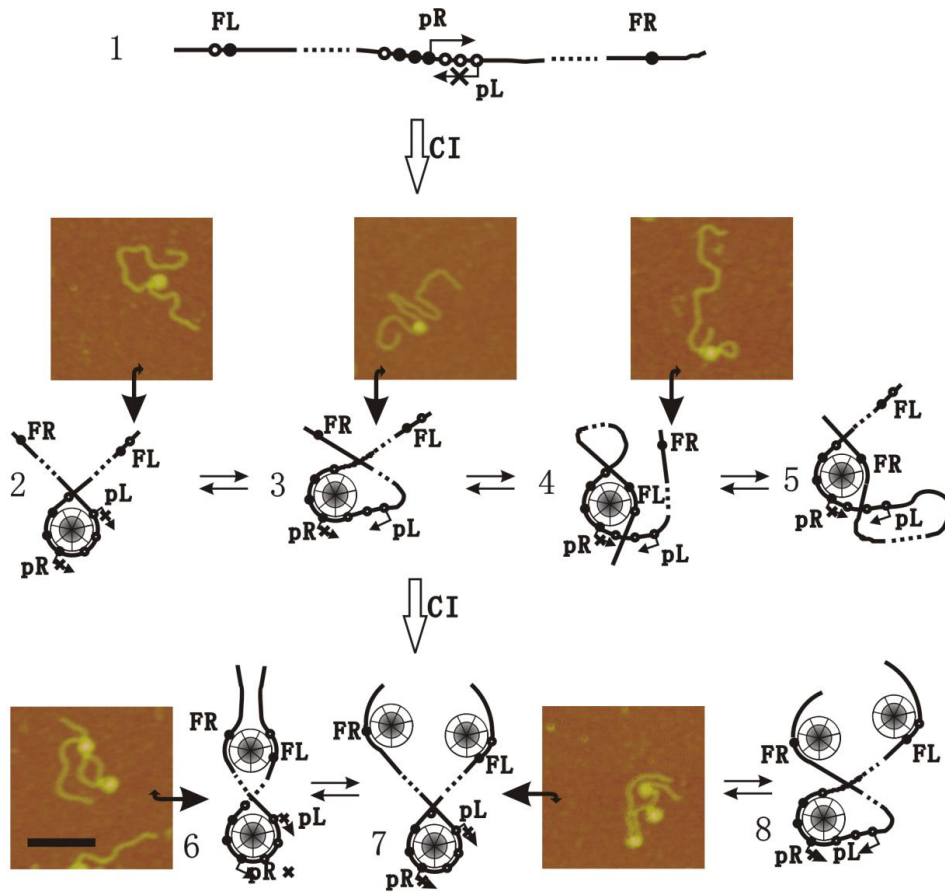
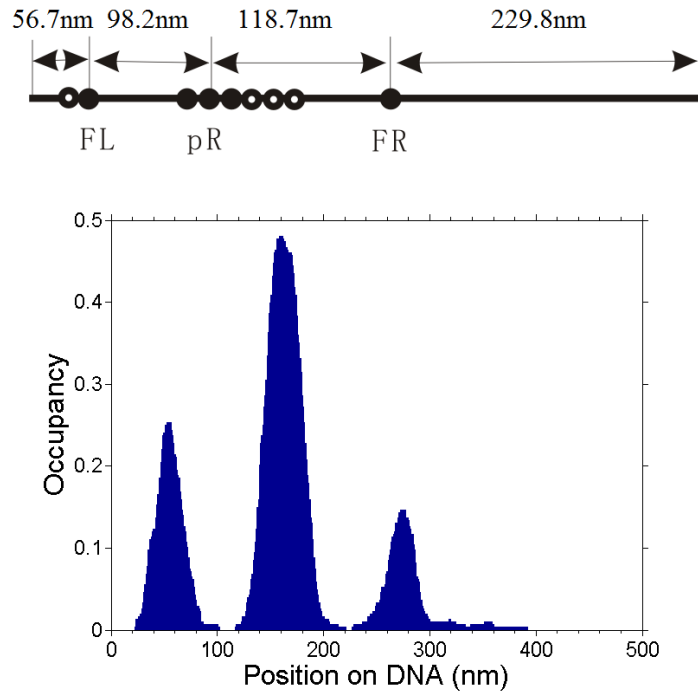


Figure 3.9: Schematic representation of the linear, wt, 1584 bp-long DNA fragment used for AFM imaging in the absence of repressor (1). The full dots represent specific binding sites for the 186 repressor, while the empty dots identify pseudo sites. Schematic representation of the nucleoprotein complexes (2-8) which could co-exist in equilibrium with the AFM images that support their occurrence. As the concentration of repressor increases, complexes with more than one wheel bound to DNA (6-8) may become more probable.

Understanding the 186 regulatory mechanism requires characterization of the specific interaction of the 186 wheel with the operators FL, pR, and FR and quantification of the probability of occurrence of each species. Thus a statistical analysis of the AFM images acquired was performed. Figure 3.10 shows that the

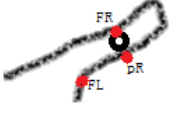
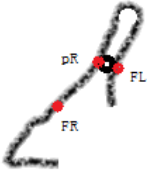
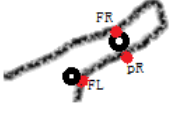
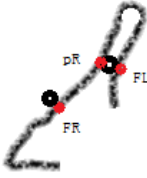
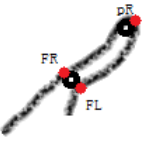
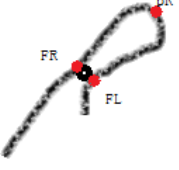
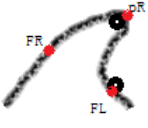
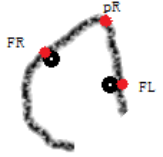
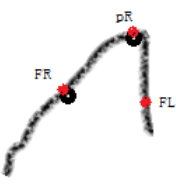
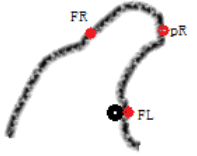
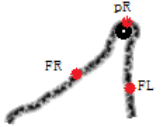
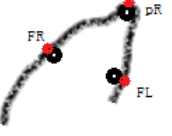
occupancy of the operators ranks as follows:  $pR > FL > FR$ , independently of the DNA conformation that the protein mediated.



*Figure 3.10:* Frequency distribution histogram of the measured location of 186 wheels along the DNA molecules measured by AFM. Left-to-right, the peaks agree well with the expected position for the FL, pR and FR sites.

Table 3.1 reports the distribution of the nucleoprotein complexes found. The images reveal that the 186 wheel may interact with DNA either by wrapping or by looping it.

Table 3. 1: Statistics on the interaction between 186 CI and wt DNA

Type	Figure	Number	%	Type	Figure	Number	%
FR-pR loop		7	2.2	FR-pR loop		11	3.5
FR-pR loop w/ FL		5	1.6	FL-pR loop w/ FR		2	0.6
FL-FR loop w/ pR		69	21.9	FR-FL loop		4	1.3
FL + pR		66	21.0	FR + FL		3	1.0
FR + pR		20	6.3				
FL only		5	1.6	pR only		88	27.9
3 particles		27	8.6	No particle		8	2.5
total		315	100				

Condensed table:

Shortening type	Shortening (bp)	Percentage
Big loop between FL and FR	678	23.2%
Small loop between pR and FL (or FR)	307 or 371	8.0%
3 particles wrapping (fully or partially)	Less than 600	8.6%
2 particles (fully or partially)	Less than 400	28.2%
1 particles (fully or partially)	200 or less	29.5%

When there is no CI protein in solution, pL is always repressed by the strong promoter pR by transcriptional interference. This is because, as explained in § 1.3.1., RNAP which may bind at pL will be quickly removed by the RNAP from the frequently activated strong pR promoter (Fig. 3.9. case 1). When CI concentration is low, CI will first bind on the strong promoter pR and turn off the transcriptional interference of pL by repressing pR. However, the seven dimers within a wheel-like particle can cover not only three binding sites of pR but also pL region simultaneously, by way of DNA wrapping. Since pL is both the promoter for transcription of CI and a binding site for the same protein, regulation of CI concentration will depend on which nucleoprotein complexes are near, or involve, pL (Fig. 3.9). At first one may think that the vicinity of pR would lead to constant repression of pL, however, the two flanking sites FL and FR attenuate such repression. Either of these two sites can interact with the particle on pR and loop the DNA. In this case, the wheel would no longer occupy pL (Fig.3.9, case 2&5). This would favor transcription, and production of more CI protein. When CI concentration is high, both flanking sites can be occupied by other wheel-like particles. Therefore, the competition between flanking sites and pL is dampened, pL will be mostly occupied by the particle sitting on pR and the production of CI protein will be repressed (Fig. 3.9, case 6, 7&8). This mechanism provides an explanation for how the 186 prophage

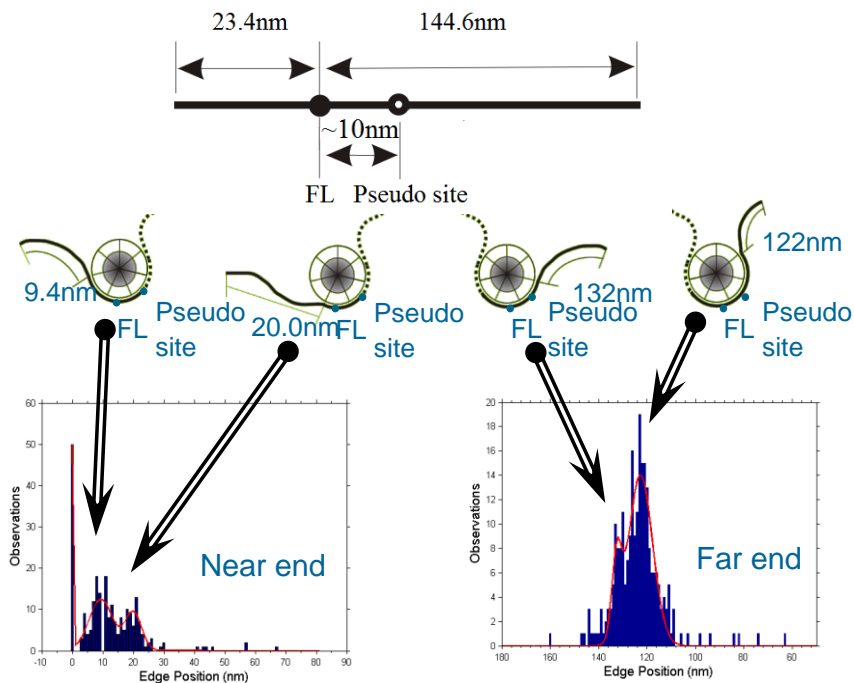
can regulate CI concentration to a level that allows maintenance of lysogeny, and keeps the ability to switch to a lytic response efficiently with a little amount of Tum protein [77].

### § 3.3.2 Pseudo sites on FL

The DNase I footprinting experiment shows the region that DNA interaction with FL is relatively bigger than others[93]. The wheel higher affinity for FL than for FR is also revealed in figure 3.10. Therefore, a weaker binding site (or a pseudo site) and its cooperativity between FL is prospected. In agreement with previous DNase digestions [93], closer analysis of the complexes at FL, performed on 524 bp/167.7 nm-long DNA fragments containing only this operator, revealed the presence of a pseudo site on the side away from pR (Fig. 3.11). The distance from each end of the DNA to the point of contact with the wheel was measured. The distribution of the lengths of free DNA measured on each end of the bound wheel is shown in figure 3.11. Given the position of FL in the synthesized DNA fragment (Fig 3.11, top), these histograms show that FL and an adjacent pseudo site in the direction away from pR were always occupied. Each distribution shows two peaks separated by about 10 nm. This corresponds to the footprint of one dimer in the wheel since it is close to one seventh of the perimeter of the 186 heptamer. The left histogram shows that the free DNA on the left of the bound wheel was, in average, either 9.4 or 20.0 nm long. Since FL was centered in this DNA fragment 25 nm from the end in the direction of pR (left end in the diagram in Fig 3.11), the peak values indicate that one dimer of the wheel binds at FL, leaving approximately 20 nm of free DNA to the left. However, the next 10 nm of this free DNA may bind dynamically to the next dimer in the wheel. On the



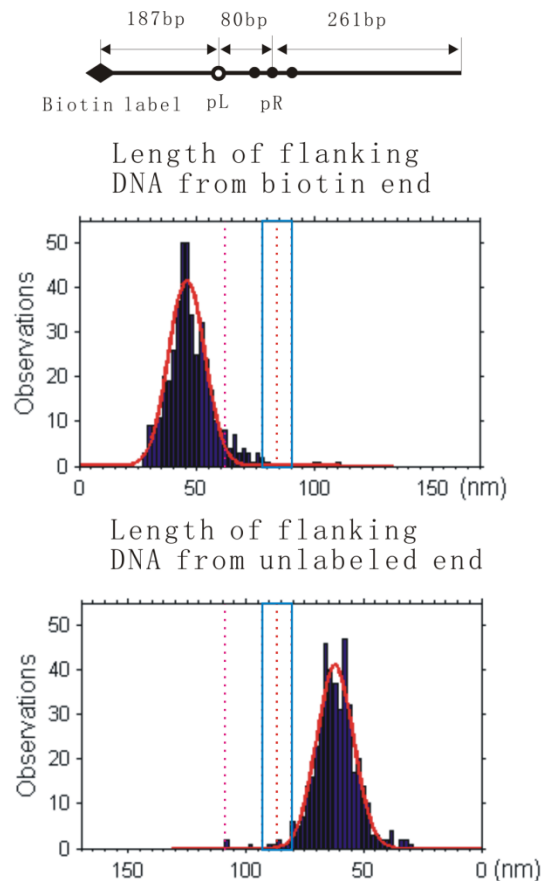
other hand, the right histogram in figure 3.11 shows that the free DNA on the right of the bound wheel was, in average, either 122 or 132 nm long. FL was centered 150.6 nm from the end of the DNA fragment away from pR (right end in the diagram in Fig 3.11). Thus, the peak values indicate that two dimers of the wheel bind both FL and an adjacent pseudo site, leaving approximately 132 nm of free DNA to the right. Ten more nm of this free DNA may bind dynamically to yet the next dimer in the wheel leaving 122 nm free. On the basis of these observations it is suggested that a pseudo site for binding of the 186 repressor exists next to FL on the side away from pR. Note also that DNA binding to successive dimers around the wheel leads to its wrapping by DNA.



*Figure 3.11:* 186 wheel positioning at FL. A short DNA fragment containing only FL was incubated with 186 CI and imaged by AFM. The distance from each end of the fragment to the point of attachment to the wheel was measured and histogrammed. *Top:* schematic representation of the DNA fragment used showing FL and its distance from each end of the fragment. The solid arrow shows the direction to the pR and FR sites. *Bottom left.* Distribution of the lengths of free DNA, before binding point, measured from the end nearest to pR. *Bottom right.* Distribution of the lengths of free DNA, before binding point, measured from the end far from pR. Each peak in these histograms is assigned to a DNA wrapping conformation shown in the associated cartoon.

### § 3.3.3 Asymmetric DNA wrapping on pR region

Asymmetric DNA wrapping on the 186 wheel was also observed in 528 bp/179 nm-long DNA fragments that contained only pR (Fig 3.12). Here, the wheel is not centered on pR because it most often occupies a pseudo site, containing pL, as well. This is consistent with the idea that the protein bound at pR will repress pL leading to 186 CI negative autoregulation, unless competition from distal sites frees the repressor promoter [90].



*Figure 3.12: Top:* Schematic representation of the DNA fragment used. The biotin-labelled DNA was incubated with both the 186 repressor and 1  $\mu\text{g/ml}$  streptavidin to identify the end of the DNA fragment close to pL. *Center:* Distribution of measured DNA lengths from the streptavidin labeled end to the point of contact between DNA and the protein wheel. *Bottom:* Distribution of lengths from the other end. The (purple) dash lines indicate the center of pR and pL. The solid lines indicate the region spanned by the three pR operators. The DNA between the peaks in each histogram is occupied by the protein.

### **§ 3.3.4 DNA wrapping/unwrapping**

AFM imaging of 1584 bp-long fragments of wt 186 DNA containing all binding sites showed that the degree of wrapping of DNA around the wheel depends on the operator. The 186 wheel bound at pR is most often found to be fully wrapped by DNA (Fig 3.9, species 2, 3, 6 and 7), while at FL and FR may be more often only partially wrapped such that the DNA going in does not cross over the DNA coming out of the wheel (Fig 3.9, species 7 and Table 3.1). However, the wheel may also mediate a loop between either FL or FR and pR (Fig 3.9, species 4 and Table 3.1). Furthermore, in the presence of a wheel already wrapped at pR, a second wheel may bridge FL and FR (Fig 3.9, species 6 and Table 3.1).

### **§ 3.3.5 TPM study of DNA wrapping and looping**

In AFM imaging, the DNA and protein are deposited onto a poly-ornythin coated mica surface and washed with HPLC water. During this process, DNA-protein complexes may be washed away or may dissociate. Therefore, TPM experiments were carried out to provide complementary information on the interaction between 186 CI and its DNA. Furthermore, TPM experiment can provide information of dynamic looping and wrapping compare to solidly fixed AFM sample.

In TPM experiments, micro beads are tethered to the surface of a microscope flow-chamber by single DNA molecules. Therefore, the Brownian motion range of the beads is limited by the tether length. Before a real experiment is run, a calibration curve of the average x-y displacement ( $\langle \rho_{\perp} \rangle$ ) as a function of DNA tether length was made to address the DNA shortening. Table 3.2 shows the calibration data obtained

with five DNA segments: 186 wild type (1898 bp), 944 fragment (1555 bp), 1051 fragments (225 bp, 1064 bp and 2974 bp). The bead diameter is 479 nm. The measured  $\langle\rho\rangle$  and DNA length are fitted by the equation obtained from Monte Carlo simulation [25, 49] (fig. 3.13). Using this curve, DNA shortening due to looping and wrapping can be studied quantitatively and information of looping/wrapping dynamics can be revealed.

*Table 3.2. Calibration data of TPM experiment.*

DNA length (bp)	Average of $\langle\rho\rangle$ (nm)	STD of $\langle\rho\rangle$ ( $\pm$ nm)
225	117.66	5.7
1064	210.94	7
1555	247.34	6.7
1898	270.22	3.7
2974	322.42	8.2

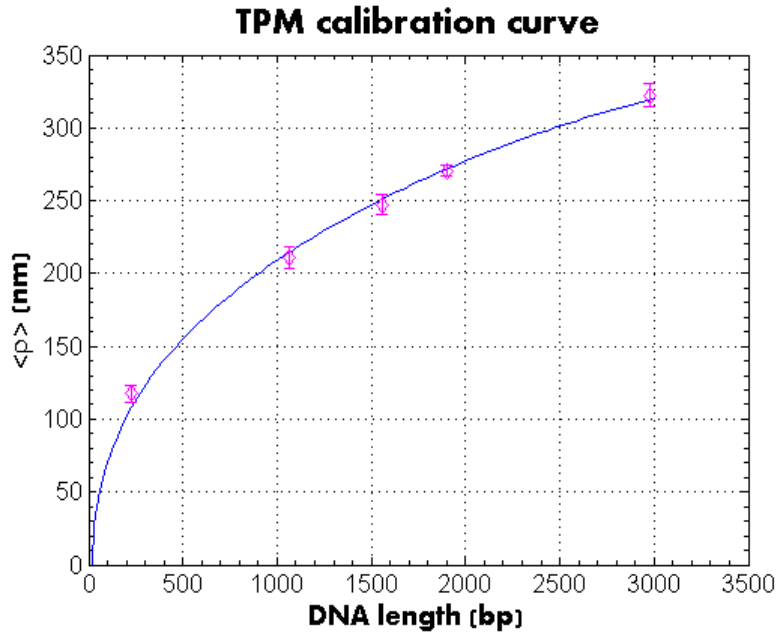
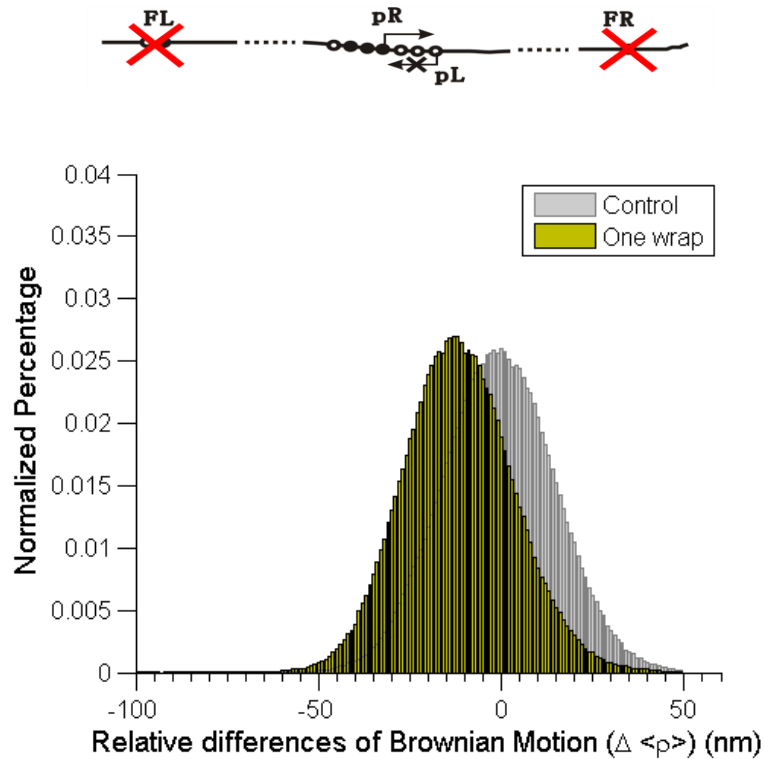


Figure 3.13: TPM calibration curve. Measuring the Brownian motion of particles with well known tether length provides the relationship between tether length and the x-y range of Brownian motion  $\langle \rho \rangle$ . Fitting measured  $\langle \rho \rangle$  with simulation model (*J. Phys. Chem. B*, 2006, 110, 17260-17267) provides a calibration curve correlating  $\langle \rho \rangle$  and tether contour length.

The fully wrapped conformation at pR was observed also by TPM using 1898 bp-long FL<sup>-</sup>pRpLFR<sup>-</sup> DNA tethers. Comparing to control data, addition of repressor in the microchamber caused an immediate and stable decrease of the TPM signal,  $\rho_{\perp}$ , by 12.2 nm (Fig 3.14) which corresponds, according to a calibration curve obtained in identical buffer conditions (Fig 3.13), to a shortening of the DNA tether of 210 bp. This is the decrease expected for a full wrapping event assuming that each 186 dimer binds 10 nm of DNA and that a heptamer will therefore wrap approximately 70 nm or 210 bp of DNA. This assumption is justified by the structural information available (see above) and by the AFM study on the DNA fragment containing only FL described above.



*Figure 3.14:* Frequency distribution of TPM data for 1898 bp-long 186 DNA tethers containing the wt binding sites as well as mutated sites. When only pR is present,  $\langle \rho \rangle$  decreases by 12.2 nm. This corresponds to approx. a 210 bp shortening in DNA tether which is consistent with a fully wrap at pR. Here, as well as in all following measurements [CI] = 50 nM and DNA tethers were 1898 bp in length.

Interestingly, TPM assays performed on 1898 bp-long DNA tethers containing only the FL site (FL. $\Delta$ pRpL.FR- DNA) showed a similarly stable shortening of about 11.3 nm (Fig 3.15). In this case too, the TPM traces recorded did not show transitions between the wrapped and unwrapped conformations as shown by the representative traces (Fig 3.16), their associated frequency distribution histograms, and by the frequency distribution of the average TPM signal for each of the beads analyzed for the FL. $\Delta$ pRpL.FR- DNA tethers in the absence and in the presence of 50 nM 186 CI (Fig 3.15).

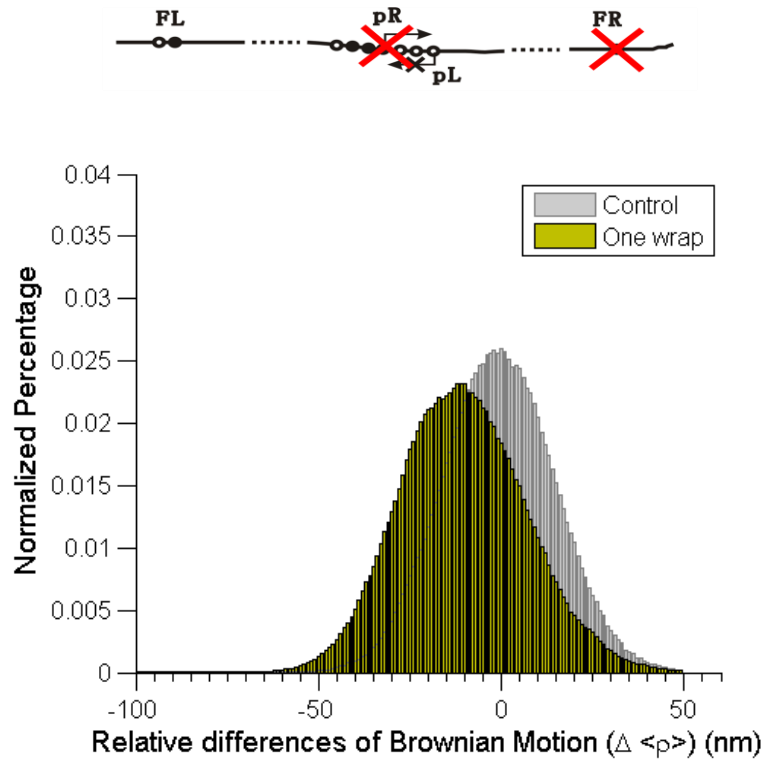
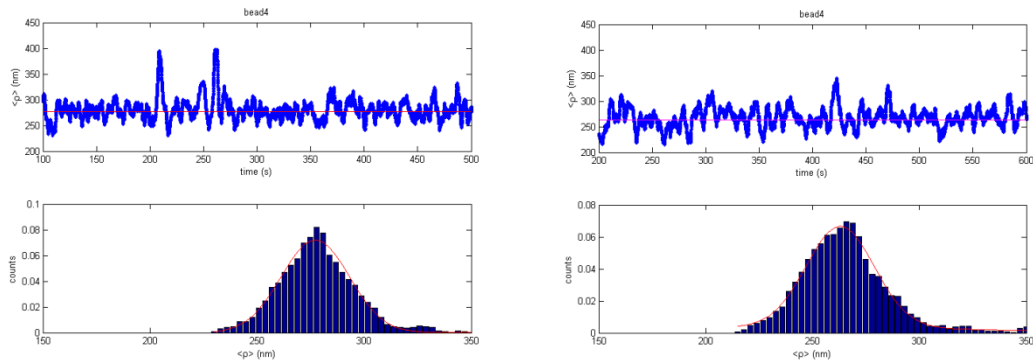


Figure 3.15: When only FL is present,  $\langle \rho \rangle$  decreases by 11.3 nm. This shortening is close to that of the DNA fragments containing pR site only and could also correspond to a full wrapping event at FL.

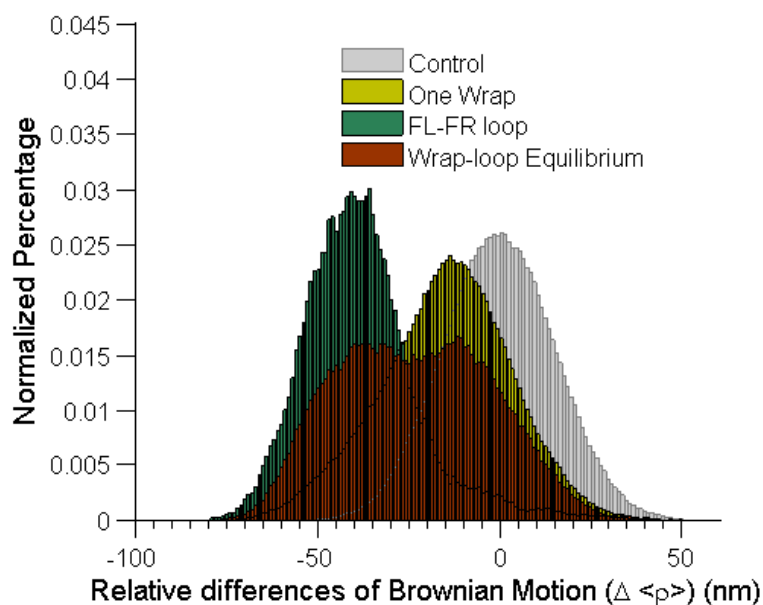


Without protein

[Cl]=50 nM

Figure 3.16: TPM trajectory of a representative FL- pR FR- DNA tether. The histogram of  $\langle \rho \rangle$  contains only one peak after adding 186 Cl. The trajectory does not have any transition on  $\langle \rho \rangle$ .

Wild type DNA (Fig. 3.17) shows two main peaks consistent with a conformational state where DNA fully wraps around one wheel, and one where a second wheel mediates a loop between FR and FL (species 6 in Fig.3.9). Most often these states are stable for the duration of the measurements, but transitions may be observed between the wrapped and looped configurations. Notice, however, that the distribution is broad and probably includes all the species observed by AFM imaging.



*Fig 3.17:* Wild type DNA in the presence of protein shows a peak consistent with a conformational state where DNA fully wraps around one wheel, and a peak where a second wheel mediates a loop between FR and FL (species 6 in Fig.3.9). Most often these states are stable for the duration of the measurements, but transitions may be observed between the wrapped and looped configurations (bimodal histogram). Notice, however, that the distribution is broad and probably includes all the species observed by AFM imaging.

### §3.3.6 DNA looping

Although wrapping seems to be preferred (Tab 3.1), AFM images revealed the presence of nucleoprotein complexes including wheel-mediated DNA looping (Fig 3.9, species 4 and 6). These complexes were classified and their relative weight was



measured for wt DNA (FL<sup>+</sup> pR<sup>+</sup> FR<sup>+</sup>), as well as for FL<sup>+</sup> pR<sup>+</sup> FR<sup>-</sup>, where the FR site was mutated, and for FL<sup>+</sup> ΔpR FR<sup>+</sup>, where the pR sequence was replaced with a sequence of equal length that did not bind 186 CI. The results of this statistical analysis are reported in Tables 3.1-5. In all cases, DNA wrapping around the repressor is more common than repressor mediated looping.

Table 3.3: Statistics on the interaction between 186 CI and FL+ delta pR FR+

Only one particle					
FL	23	27.7%	FR	12	14.4%
Nonspecific	26	31.3%			
Two particles					
FL & FR	2	2.4%	Two nonspecific	2	2.4%
FR & nonspecific	1	1.2%	FL & nonspecific	4	4.8%
Three particles					
FL, FR and one nonspecific	1	1.2%	FL and two nonspecific	1	1.2%
One loop					
FL-FR loop	4	4.8%	Two nonspec. loop	4	4.8%
FR-nonspecific loop	1	1.2%	FL-nonspec. loop	1	1.2
One loop and one particle			Total		
FL-nonspecific loop + one nonspecific particle	1	1.2%	83		

Table 3.4: Statistical result of interaction on FL+pRFR- mutation.

Binding Location	Number of Molecules	Percentage of Molecules
No proteins bound	99	9.25%
Only pR site	673	62.90%
Only FL site	18	1.68%
Two particles FL and pR sites	224	20.93%
One particle Loop with FL and pR	45	4.21%
Nonspecific Binding	11	1.03%
TOTAL:	1070	100.00%

Table 3.5: Statistical result of interaction on FL+ pR+FR-mutation.

No particle	9.25%		Only pR	62.90%
Nonspecific	1.03%			
Total	10.28%			
Only Fl	1.68%		Two particles FL and pR sites	20.93%
			One particle Loop with FL and pR	4.21%
			Total	25.14%

Tables 3.4&3.5 show a statistical analysis of AFM images of the DNA fragment carrying only the FL and pR sites. According to the Boltzmann distribution, the ratio between different states,  $S$ , in equilibrium depends only on the free energy of each

state. If the CI wheel binds to pR and FL independently, the free energy of the state where both sites are occupied ( $\Delta G_{pR,FL}$ ) should be the sum of free energy changes associated with the formation of each of the other two states: the state with only one wheel bound at pR ( $\Delta G_{pR}$ ) and the state with only one wheel at FL ( $\Delta G_{FL}$ ). Therefore, the population of four states ( $S_1$ : no protein;  $S_2$ : only pR occupied;  $S_3$ : only FL occupied;  $S_4$ : pR and FL both occupied) will be related as follows:

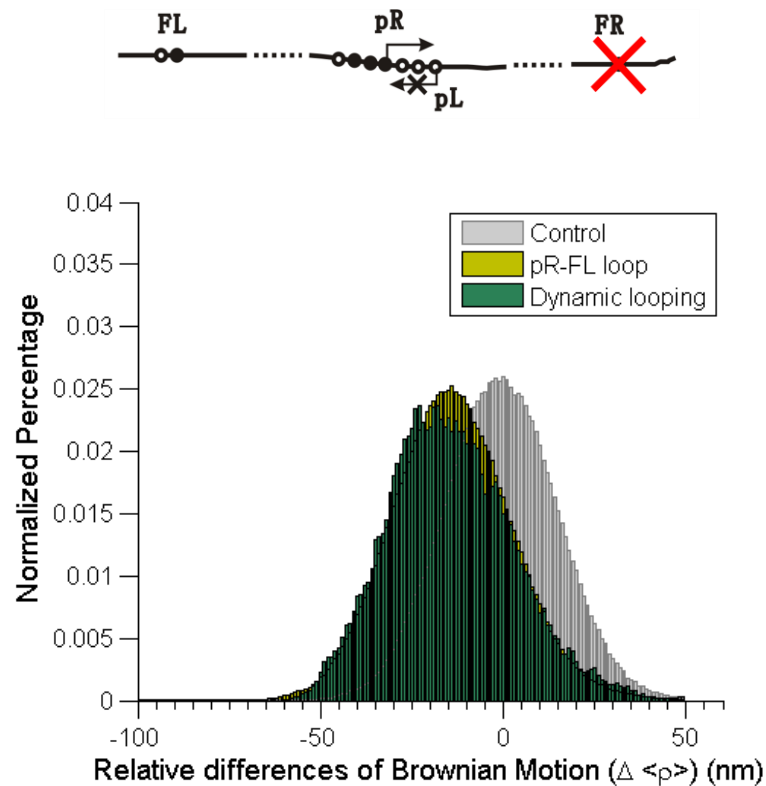
$$S_1/S_2 = S_3/S_4$$

However,  $S_1/S_2$  calculated from table 3.5 is 0.16 and  $S_3/S_4$  is 0.07. Since  $S_1/S_2$  is more than two times bigger than  $S_3/S_4$ , cooperativity may exist between FL and pR.

In solution, 186 repressor-mediated looping versus wrapping was investigated by TPM. After addition of repressor to wt 186 DNA, most of the tethers adopted either one of two conformations, characterized by an average decrease in  $\langle \rho_{\perp} \rangle$  of 14.5 nm (most probable) and 37.0 nm, each, which correspond to a shortening of the DNA tether of approx 250 bp and 580 bp, respectively (Fig 3.17). The 250 bp shortening is greater than the one associated with a full wrapping event. Thus, it could result from a wrapping event at the strong pR sites and a partial wrapping at one of the flanking sites as well as from a looping event between pR and either FL or FR. In this respect, notice that the histogram is quite broad. The 580 bp shortening may be interpreted as due to the wrapping of the DNA around three wheels bound one to each operator (FL, pR and FR) or to the formation of a loop between FL and FR, since the distance between the centers of these two operators is 678 bp. The difference between 678 bp and 580 bp can be explained by experimental error and the diameter of 186 wheel-

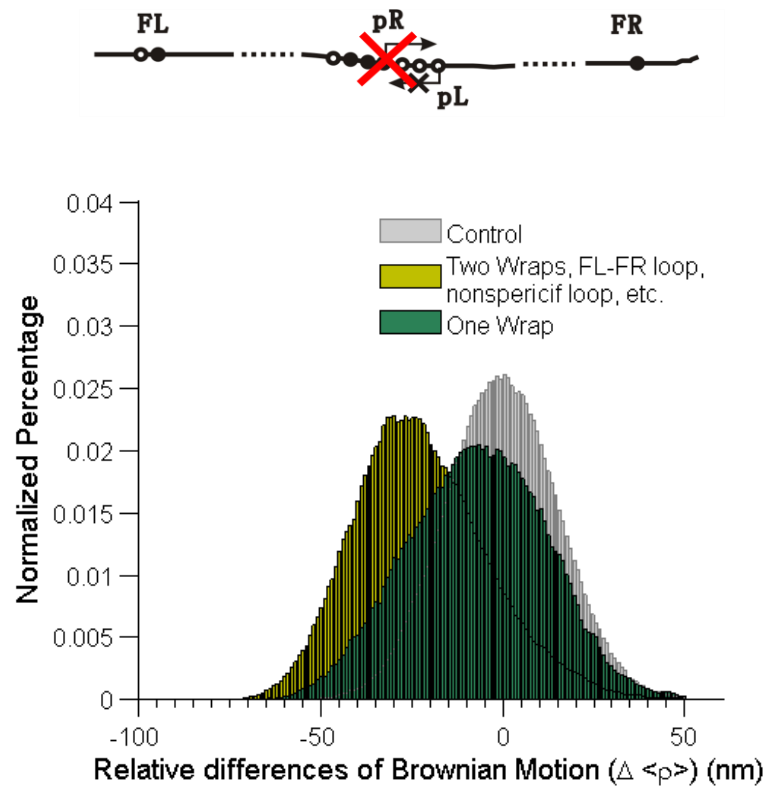
like particle (20 nm, which may be looked as 60 bp long DNA tether). Notice that in this looped state, a second wheel may be bound at pR, but would not cause a detectable TPM signal. Out of 31 molecules that were analyzed, only 5 displayed just one or two transitions between the two states in 20 min of observation, but never back to the free DNA state. Their frequency histogram was, therefore, bimodal. Although TPM measurements did not show all the nucleoprotein complexes revealed by AFM, one should notice that the TPM histograms are quite broad, and it is possible that several nucleoprotein complexes, including the loop between pR and one of the flanking sites, coexist in equilibrium, without being clearly resolved by TPM.

TPM measurements performed on DNA tethers containing only FL and pR (Fig 3.18), showed a 14.5 nm decrease in  $\langle \rho_{\perp} \rangle$ , corresponding to 245 bp shortening of the DNA tether. This shortening, as already discussed for the wt case, may be interpreted as due to a full wrapping event, probably at pR (will take around 210 bp). Even if FL contains less binding sites than pR (fig. 3.1), a particle bound at FL may still be partially wrapped by DNA. This would explain why this shortening observed with this fragment is bigger than the one observed for the fragment containing only pR site. The broad TPM frequency distribution histogram may also be consistent with a loop which was dynamically forming and breaking between FL and pR. This loop would consume some 300 bp of DNA if the two binding sites came in direct contact, but the wheel would reduce the observed shortening. Indeed, three of 44 FL<sup>+</sup>pRpLFR<sup>-</sup> DNA tethers display two peaks, one at 18.9 nm and the other at 0, respectively, and can be explained by the transition between the looped and the unlooped DNA at FL and pR.



*Figure 3.18:* In the absence of FR, many DNA tethers are stably shortened by  $\sim 245$  bp ( $\sim 14.5$  nm in  $\langle \rho \rangle$ ), which is consistent with a loop between FL and pR (including the size of the 186 wheel). Some of the tethers display brief transitions back to the unlooped or partially wrapped state (shoulder at 0 nm). The broad distribution of measured reductions in  $\langle \rho \rangle$  may result from tethers where the DNA wraps around the 186 repressor (supported by AFM, see Table 3) either at pR or at FL.

TPM of FL+  $\Delta$ pR FR+ DNA was also performed (Fig 3.19). These molecules are not expected to bind the 186 wheel at pR. DNA tethers which displayed just one peak after addition of repressor could be separated into two groups. One group of molecules showed an average decrease in  $\langle \rho_{\perp} \rangle$  of 24.9 nm, corresponding to 410 bp shortening of DNA tether.

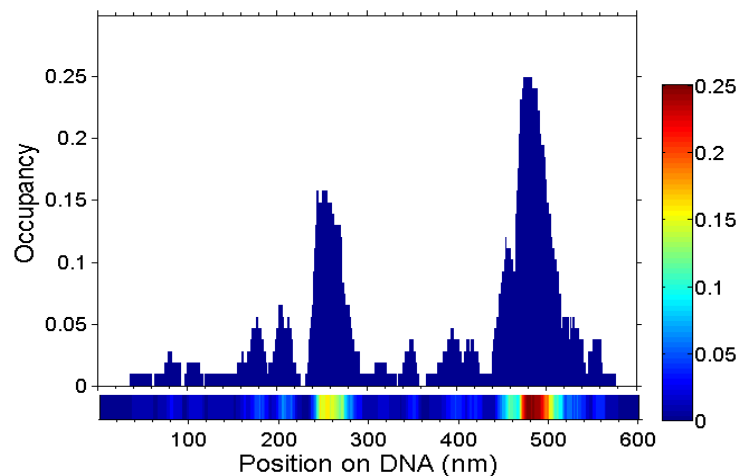


*Figure 3.19:* When the binding sites at pR are deleted, the tether shortening observed cluster into two groups: one consistent with one wrapping event and another which could include both two wrapping events and a FL-FR loop.

The 410 bp shortening is unexpected because there is no known pair of binding sites which can cause this shortening. Since the ratio between 186 monomer to DNA is 50:1, one wheel needs 14 monomers to form, and there is a complex equilibrium between several protein oligomerization states, one DNA may in average only have 2-3 wheels. If there is not pR, FL and FR may always be occupied and prevent loop formation by just one wheel bound both at FL and FR. Therefore, this shortening may come from two full or partial wrapping on FL and FR respectively. The other group of data shows an average 7.0 nm decrease of  $\langle \rho_{\perp} \rangle$ , which, considering the standard deviation of the data, can be due to a single wheel fully wrapped at FR or FL. Once again, TPM seems to reveal fewer nucleoprotein complexes than AFM. In particular, the loop between the two flanking sites was not distinctly detected in the TPM

measurements performed on this mutated 186 DNA fragment, and the proportion between one wrapped and two wrapped wheels is not the same as in the AFM images despite the similar DNA/repressor concentration ratio in the two types of measurements.

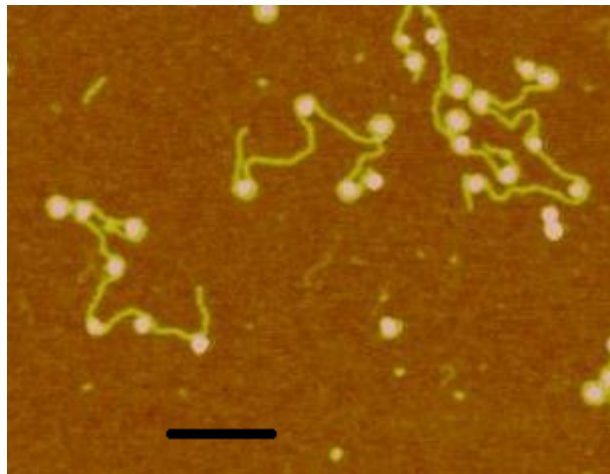
The overall interpretation of all these observations should not neglect to consider the possible role of nonspecific binding. An occupancy analysis, performed on the AFM images of the FL+  $\Delta$ pR FR+ DNA (Fig 3.20), revealed several weaker binding sites, which may play a role in shaping the equilibria between the nucleoprotein complexes involving FL, pR and FR. Indeed, DNA loops between a specific and a nonspecific site were observed by AFM in the absence of pR (Table 3.3). Therefore, the histograms of TPM signals may be broadened also by transient interactions with nonspecific sites which may have the physiological role of facilitating and/or stabilizing specific interactions that regulate the 186 bacteriophage genetic switch.



*Fig 3.20:* Frequency distribution histogram of the location occupied by the 186 repressor wheel on FL+  $\Delta$ pR FR+ DNA in the presence of 50 nM 186 CI, as detected by AFM imaging. The two major peaks belong to the specific sites FR and FL. The small peaks indicate other locations where the wheel was found. At these weak binding sites, the protein either wrapped DNA or bridged the site to FR/L via looping.

### § 3.3.7 Other CI binding forms and non-specific binding

The 186 repressor can bind non-specifically, just as many prokaryotic repressors and probably most transcriptional factors. This ability is clear from the analysis of AFM images of the beads-on-a-string fiber that 186 DNA forms in the presence of 300 nM repressor (Fig 3.21). Non specific binding is eliminated when using 186 CI mutant. AFM imaging also showed there is some kind of non specific interaction between wild type protein and non-related DNA ( $\lambda$ ) or the FL- delta pR-pL FR-DNA.



*Figure 3.21:* AFM image of 186 wt DNA in the presence of 300 nM 186 repressor. The way that DNA wraps on the 186 wheels resembles strongly that in which DNA wraps histones in chromatin. The study of the interaction between DNA and 186repressor might serve as a model of how DNA wrap and flutter on such kind of particles. Scale bar: 100 nm.



## **Chapter 4**

# **Automated DNA Segmentation and Protein Recognition from AFM Images**

#### § 4.1. Background

AFM can visualize protein-DNA complexes by scanning a solid surface where these are adsorbed. Although the AFM lack the ability to identify the atoms and chemical bonds of bio-molecules, this technique is widely used because convenience sample preparation and nanometer resolution. For example, AFM can visualize the formation and changing of the DNA loop associate with RSC and study its ability to modify DNA structure [94]; imaging RNAP transcribing ds-DNA in solution can be used to measure properties such as transcription rate and DNA dissociation [95]. Furthermore, by measuring the curvature and end-to-end distance of DNA deposited onto mica surface with AFM was already used to study the stiffness of DNA molecules under different condition[96].

In most of studies, DNA images obtained by AFM need to be transformed into skeleton by tracing process before measuring. The most direct way of tracing is to point out the DNA skeleton from image point-by-point. This time consuming process can be improved by interactively tracing computer algorithm [59, 61, 97]. In those algorithms, a set of “seed points” are provided by user experience with a mouse. The program then successively connects these points with traces that best fit the DNA skeleton according to local cost function of each pixel around[97]. These semiautomatic tracing methods greatly improved the efficiency and accuracy of DNA tracing [97]. However, there are two drawbacks to this kind of method. First, since the DNA skeleton need to be outlined by the experimenter, this process is still very time consuming, especially when a large data set is needed for statistical analysis. Second, because the selection of the points of the DNA skeleton is made by hand and

subjective, the operator bias may affect the data. Thus, an automated segmentation program is useful to improve the efficiency and minimize artifacts.

The thinning procedure derived from the work of Brugal and Chassery [98] is one of the mostly used automated tracing methods [99-101]. It first transforms the image into a bi-color map with a threshold. The next process, iteratively removes pixels from the edge of DNA segments, if the removal of the pixel does not sever the segment. This process will repeat until no more pixels can be taken out. This procedure is relatively efficient and leaves behind DNA skeletons only one pixel wide [100]. Then, the computer can easily trace the one-pixel wide skeleton from one end to the other. Finally, sets of pixel coordinates representing DNA traces are generated for later analysis.

Although fully automated tracing algorithms are very efficient and reproducible, the heterogeneity of the sample often prevents their implementation [102]. Bound proteins and the image noise can both affect the accuracy of the DNA skeleton identification. In particular, long DNA fragments often follow a complex contour with several cross-over points. This requires significant user inputs to be identified, and reduce the efficiency of these algorithms [97].

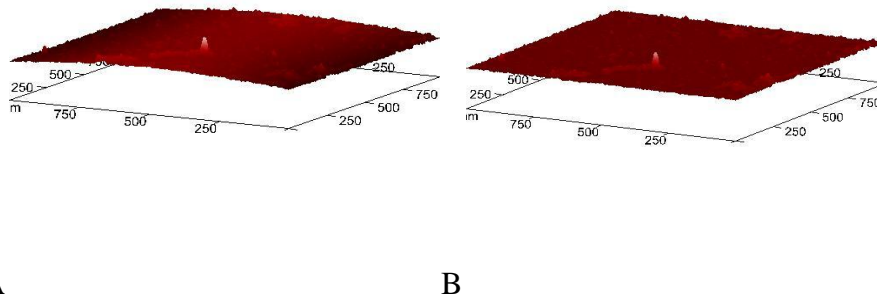
Given this challenge, a group of matlab programs were developed to improve the efficiency of automated analysis of DNA-protein AFM images. The program can automatically recognize short DNA segments and protein particles, measure the DNA molecules length, and find the position of bound particles. The program can also automatically calculate the particle height, diameter and volume. Even complexes with no simple shape may be analyzed, using a variation of the program where

complicated contours may be rebuilt from the tracing of different segments by the user. Finally, the program is easy to modify and constitute a convenient toolbox for AFM image analysis.

## §4.2. Method and algorithm

### §4.2.1 Filtering

Although AFM can provide a good signal to noise ratio compared to other techniques, the images acquired by AFM cannot be fed directly to a program. Because the response of the piezoelectric motors is not perfectly linear, the background in the AFM images is not always flat (fig. 4.1).



*Figure 4.1:* The background of AFM images may not flat. A), raw data of a test image. The middle of the image is higher than the edge. B), after 3rd order polynomial flattening, the image become flat.

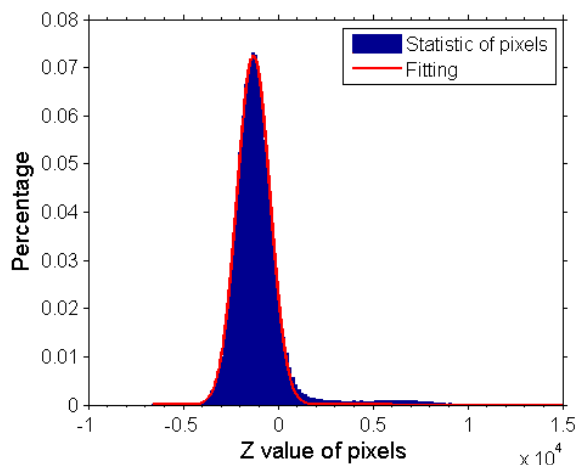
Fortunately, software available with the AFM instrument provides a flattening algorithm. This allows fitting the surface with a third order polynomial function. Subtracting it from the background of the image provides a flat background.

#### § 4.2.2 Threshold and segmentation

Two methods can be used to segment images: one is based on the difference in gray level; the other is based on the discontinuity in grey levels between foreground and background. Because the discontinuity that marks the edge of DNA fragments is rounded by the AFM tip during imaging, DNA can only be differentiated from the background based on the difference in grey level between foreground and background.

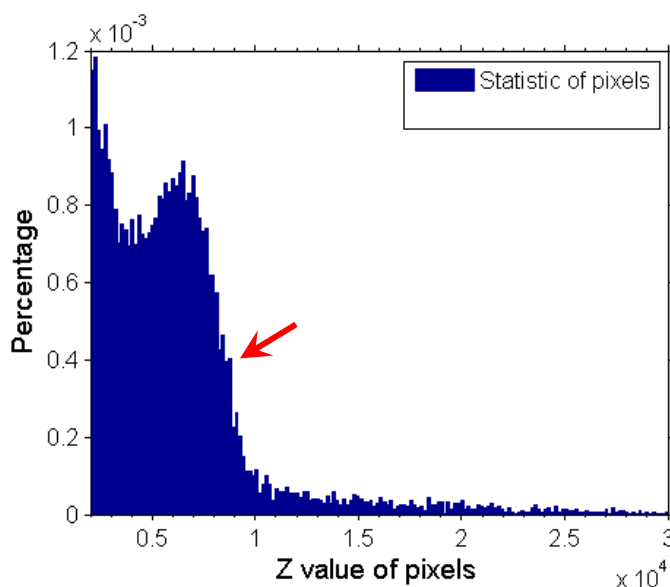
In this method, a certain threshold of gray level needs to be decided to recognize DNA and proteins from the whole image. All the image pixels with a gray level below the threshold are set to zero. Normally this threshold is calculated through an adjustable ratio between background level and the height of the DNA.

To minimize the possibility that different DNA segments cross over, the DNA concentration used was limited to the nM range. Since over 90% of the pixels are background in any given AFM image, the background level can be approximated by the mean value of all the pixels in the image (Fig. 4.2).



*Figure 4.2:* Frequency distribution of the z value of all pixels in a 512x512 AFM image of DNA and protein particles deposited on polyornithine coated mica. Only very few of the pixels belong to DNA or protein particles. These give rise to the tail on the right side which has a big Z value.

Two methods are used to extract the DNA height from the image. The first method is to fit the sudden drop in height in the histogram tail. Figure 4.3 is the zoomed view of the right side of the tail in figure 4.2. The significant decrease of the distribution can be used to calculate DNA height.

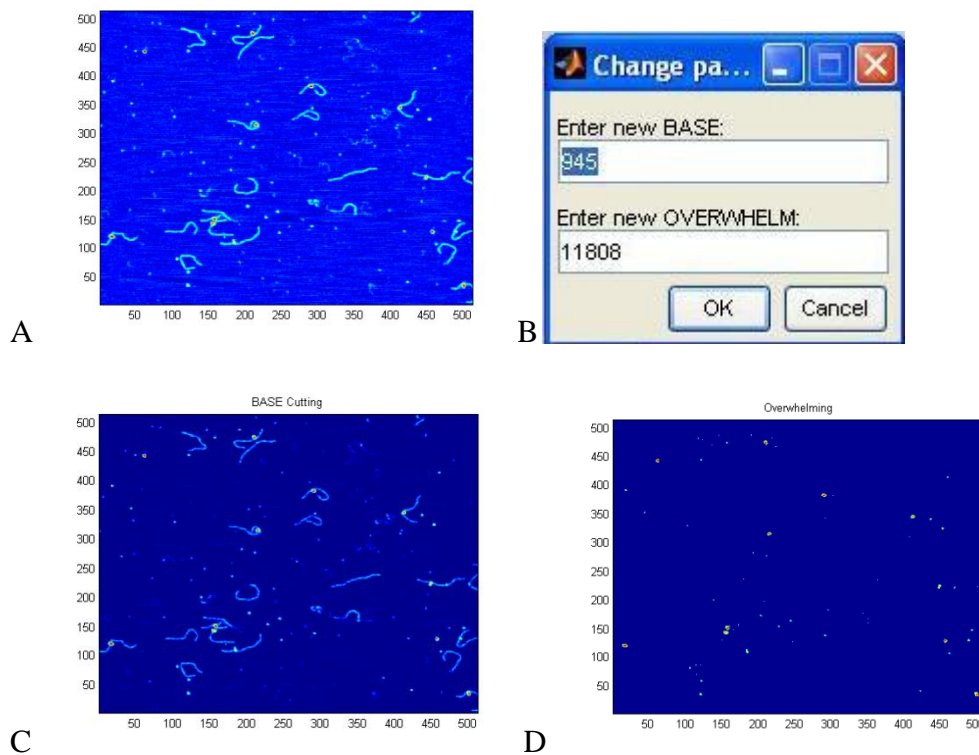


*Figure 4.3:* Zoomed view of the tail of fig. 4.2. Because there are much less pixels belong to protein blobs (which is higher than DNA height) than DNA, there is a significant decrease (red arrow) of number of pixels at the height of DNA. This sudden drop can be used to identify the height of DNA in AFM image.

This sudden drop can be fitted with a straight line. The DNA height is then calculated by the intercept of the straight line and x axis. This algorithm is very accurate in yielding the DNA height in the AFM images. However, if the image quality is not very good, the tracing program might not go to completion because the program has a hard time finding a value for the DNA height. Therefore, a second “back-up” algorithm was developed. This backup method uses the mean maximum value of each row as DNA height. Because DNA molecules are long, nearly every row of image data contains parts of at least one DNA molecule. On the contrary, the

protein blobs are rare and globular and are found in just a few rows of image data. Therefore, the maximum value of each row will mostly represents the height of DNA and only rarely the height of protein blobs (fig. 4.4 a).

Therefore, the average of row maxima will likely be close to DNA height. Although this method is not very accurate, it is robust because it does not use any fitting. If the fitting method fails to give a DNA height, this algorithm will be activated so that a value for the height of DNA height may always be obtained.



*Figure 4.4:* An original AFM image. (A); question dialog of parameter modification where “BASE” represents the calculated background and “OVERWHELM” represents the DNA height (B); background contributions to the image are eliminated by setting all the pixels below background to zero (C); All pixels below DNA height were set to zero. Only protein particles were left on the image (D).

After calculating background value and DNA height, images and a question dialog will appear as showed in figure 4.4. Fig 4.4 A shows the original AFM image after flattening. Figure 4.4C is the effect of cutting background off. In figure 4.4C, all the pixels with a value below BASE are set to zero. In figure 4.4D, all the pixels below OVERWHELM are set to zero, which means there are only protein blobs left on the image. DNA and background disappear. The program also provides a question dialog for the user to change the BASE and OVERWHELM parameters.

Once the BASE and OVERWHELM values are decided, the program will calculated the threshold value as follows:

$$\text{THRESHOLD} = \text{BASE} + (\text{OVERWHELM} - \text{BASE}) \times 0.3 \quad [\text{Eq. 4.1}]$$

The 0.3 value is an experimental value set in the program which may be changed depending on the application. Then, the program will transfer the real image into a binary map by setting all the pixels above threshold to one and those below threshold to zero.

### **§4.2.3 Thinning and selection of the DNA skeleton**

A thinning process was used to abstract the DNA skeleton from the image. This process narrows the DNA trace by taking away pixels from the edge of the DNA. The pixels were removed from four directions respectively. If the removal of one pixel will break the segment into two parts, this pixel will be preserved as a critical pixel. Figure 4.5 shows examples of pixels that will be removed/kept. Iterative steps are performed until all the pixels left are critical and the skeleton is obtained.



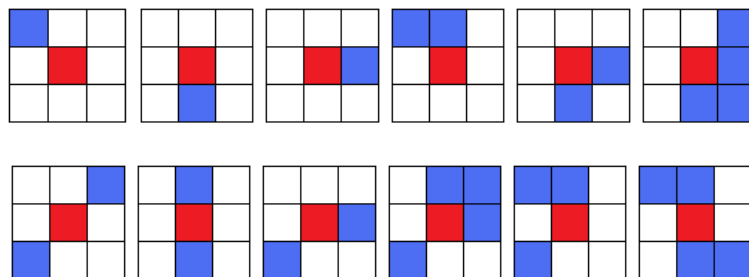


Figure 4.5: Identification of critical pixels. The red pixels in the upper row will be removed during thinning. The red pixels in the lower row are considered critical pixels and preserved because the removal will break the segment into two parts.

Figure 4.6 shows how pixels are removed from the left edge of the image. In order to guarantee that the skeleton follows the axis of the DNA molecule, pixels need to be removed from all four directions sequentially.

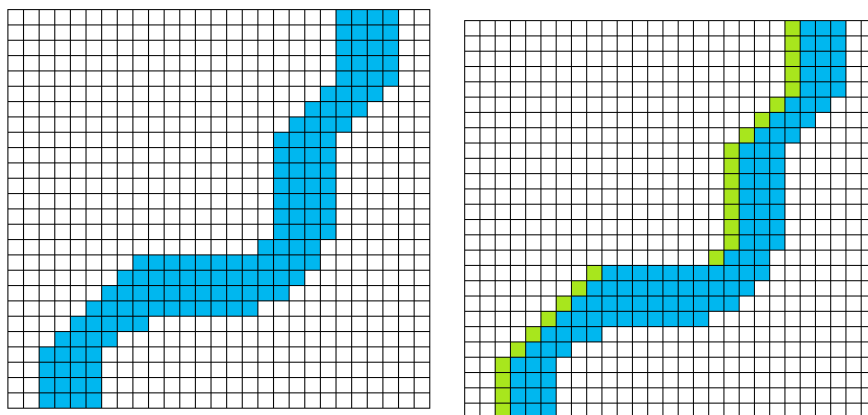


Figure 4.6: One process of thinning. Left: the original binary map of DNA segment. Right: after one step of thinning, the pixels on the left edge (green) are removed from the map.

In order to do this, the image was rotated 90 degree after each round of elimination. In each elimination cycle, the pixels are always removed from the left. In this way, the program code is simplified and efficiency of execution is improved.

Figure 4.7 shows the working flow of the thinning process.

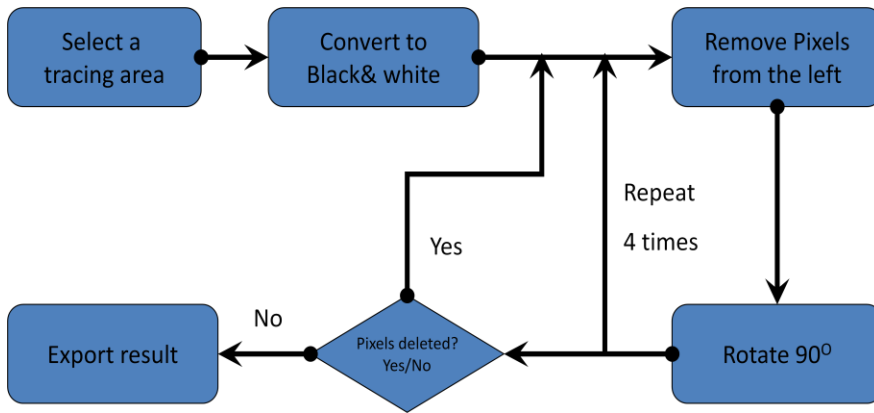


Figure 4.7: Working flow of the thinning process. Instead of removing pixels from all four directions (left, up, right and down), the pixels are removed only from the left and the image is rotated 90 degrees after each removal. After four rounds of pixel removal, the image will be set back to the original orientation. This procedure can decrease the complexity of coding and increase the code efficiency.

After several cycles of pixel elimination, the DNA skeletons are extracted from the image successfully. Figure 4.8 shows a whole cycle of pixel elimination.

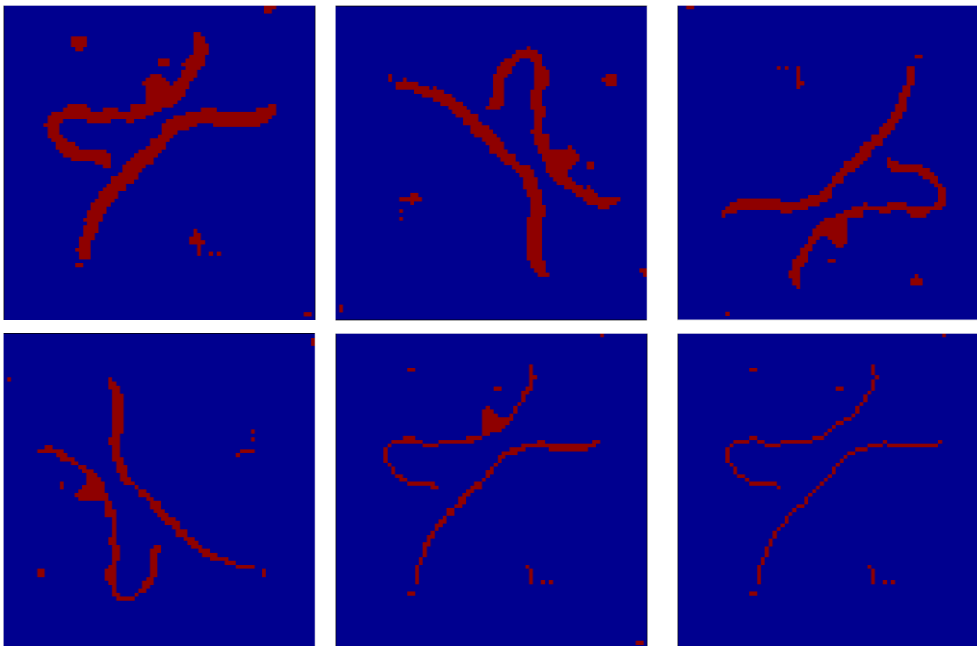
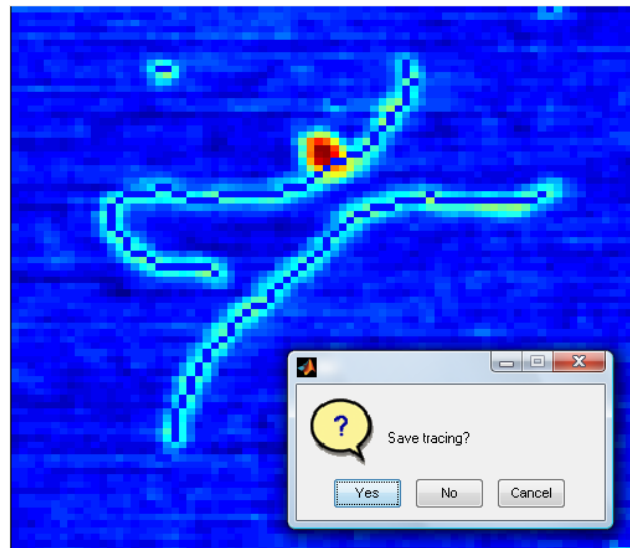


Figure 4.8: The process of thinning. From up left, center, right to lower left and center, pixels are removed from the binary map and the DNA image become thinner and thinner. Image rotates 90 degree between every two images. Lower right is the final image after thinning. Only DNA skeletons are left.

In some cases, different segments along a DNA molecule may cross over. Currently, the program lacks the ability to follow the DNA contour through these intersection points. Thus, it simply removes the pixel inside the intersection and breaks the segment into several parts. These broken segments can be reconnected by the user later.

This thinning process will remove pixels from all four directions of the fragment until the skeleton is only one pixel in width. Therefore, some pixels may be removed from both ends of the fragment. Since the DNA traces were broadened by the tip as described in §3.2.3, the two effects are likely to compensate each other and no action was taken to try to correct for them.

After thinning process, the binary map is converted into narrow DNA skeletons (Fig. 4.9). Then the program reads the x-y coordinates of each pixel in the DNA skeletons.



*Figure 4.9:* DNA skeletons were generated by thinning. Blue pixels in the middle of DNA images represent the one-pixel-width skeleton of the DNA. After thinning, the coordinates of DNA skeleton can be easily found and saved in txt files.

#### § 4.2.4 DNA length estimation

The length estimator is also a sub function of the program. In many applications, the resolution is limited by AFM tips and short scale kinks and bending may not be reflected in the image. This introduced an underestimation of DNA contour length [99]. On the other hand, the pixelization of the image may result in a shift of the skeleton with respect to the central axis of individual DNA molecules. Therefore, the DNA contour length is often overestimated by commonly used methods such as Freeman estimation[103].

In this study, five different DNA length estimators were used and compared on a set of simulated DNA molecules (Tab. 4.1).

Table 4.1: Measured DNA contour length with different estimators

Simulated 1500 bp DNA, 520 nm long.

<i>Method</i>	<i>Contour length(nm)</i>	<i>Standard deviation(nm)</i>	<i>Error (nm)</i>
Freeman	531	5.573	21
MPO	525.1	27.79	15.1
Kulpa	504.1	5.193	5.9
Corner chain	504.2	4.634	5.8
Step two	502.9	4.284	7.1

Most of these estimators are  $(n_e, n_o, n_c)$ -based estimators. When the next pixel only has one coordinate (x or y) different from the previous pixel, the segment between two pixels is looked as even. If both coordinates (x and y) are different from

the previous pixel in the DNA skeleton, the segment is looked as odd. If the moving from one pixel to the next there is an odd to even or even to odd transition, the segment is treated as a corner.  $(n_e, n_o, n_c)$  represents the number of even, odd and corner segments in one DNA segment.

The freeman estimator was introduced by Freeman in 1970 [104]. It calculates the distance between neighbor pixels and adds them together. Therefore, the total length is given by:

$$L_F=1.0n_c+1.414n_o \quad [\text{Eq. 4.2}]$$

The MPO estimator [105] was proved to be very accurate for straight segments. The formula used by the MPO estimator is:

$$L_{MPO} = \sqrt{(n_e + n_o)^2 + n_e^2} \quad [\text{Eq. 4.3}]$$

The Kulpa estimator derives from the Freeman estimator and includes derived coefficients for the even and odd pixels to minimize the error [106].

$$L_K=0.948n_c+1.343n_o \quad [\text{Eq. 4.4}]$$

The corner chain estimator includes the effect of corner [107] and the formula is:

$$L_C=0.980n_e+1.406n_o-0.091n_c \quad [\text{Eq. 4.5}]$$

The last estimator is called “step two” estimator. In this estimator, the distances between every two successive pixels are calculated and put together. Although this estimator only uses half of the coordinate on the DNA skeleton, it is a fast and easy one to estimate the DNA contour length from skeleton and keep a similar accuracy.

Results of different estimators are showed in table 4.1. The Freeman and MPO estimator over estimated the DNA length by 3-4%. The MPO estimator has the

largest standard deviation. The other three estimators have similar errors. All five estimators are included into a subfunction of the program and users can choose any of them by activating appropriate codes. Users are free to use any other estimators by inserting their codes or replace the whole subfunction. The current software uses the “step two” estimator because of its advantages in coding and testing.

### **§ 4.3. Application and programming**

#### **§ 4.3.1 DNA tracing**

The tracing program asks the user to select one or multiple files. After this file selection, the program will read data with a sub function named “readimage.m”. This sub function can read AFM image data and convert them into a matlab matrix.

The only requirement for this sub function is that the returned data must be a double precision  $n \times n$  matrix. Therefore, if the user wants to work with other types of data or image, this sub function can be easily replaced by a customized one.

Tracing results will be saved into txt files which will be named as “\*tr.txt” where “\*” is the original filename of the data file.

An example of such a data file is showed below:

```
654 8177
-1 1
32 56
31 57
. . . . .
38 77
39 76
-1 0
```

The first line of the data contains the background (BASE) value and the calculated DNA height (OVERWHELM) value. Every segment starts with [-1 trace ID] where trace ID is a positive number for the program to identify every segment in

each image. The following data are the x and y coordinates of every pixel in the trace. The last line [-1 0] indicates the end of one segment.

The matlab code of the tracing program can be found in appendix E (tracing.m).

### § 4.3.2 Masking and interactive modification

After segmentation, the coordinates of the DNA skeletons were saved. It is useful to visualize the DNA traces together with the AFM images. Therefore, a program was made to do this and allow the user interactively delete unwanted traces or connect unexpectedly broken traces. This program also provides a function that allows the user to select a part or the whole segments and measure its length.

The interface of the masking routine is showed in fig 4.10. The DNA skeleton is visualized in red and superimposed (masked) on the original AFM image. This masking routine provides to functions: delete and connect.

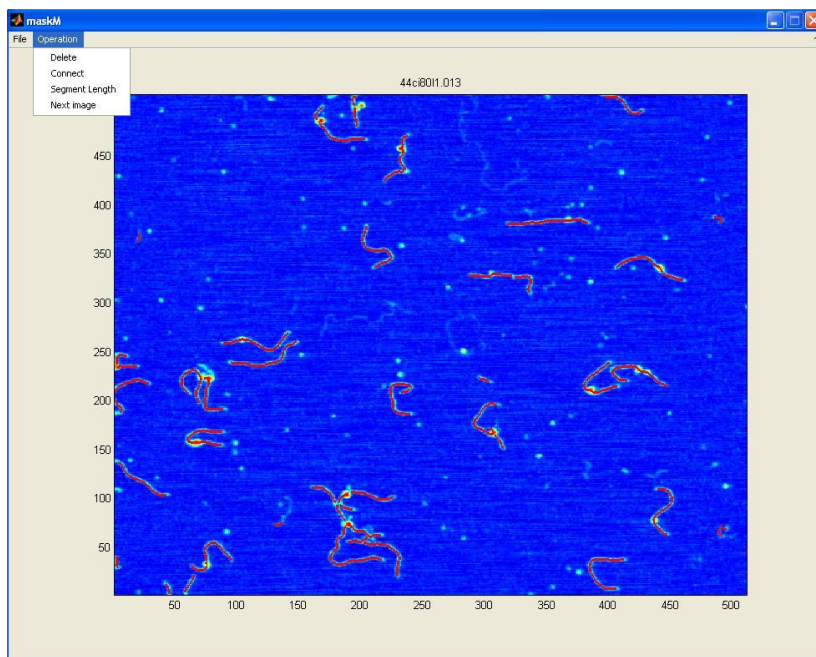
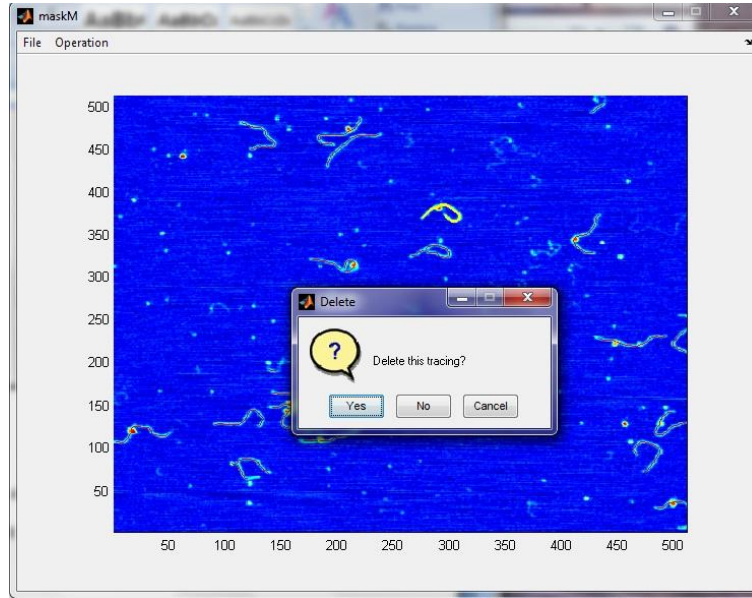


Figure 4.10: Interface of masking program. This program allows users to review DNA traces obtained by the tracing program. Furthermore, users can delete bad traces or connect broken traces together with the program. The program also allows users to select a DNA trace or a part of it and measure the contour length.

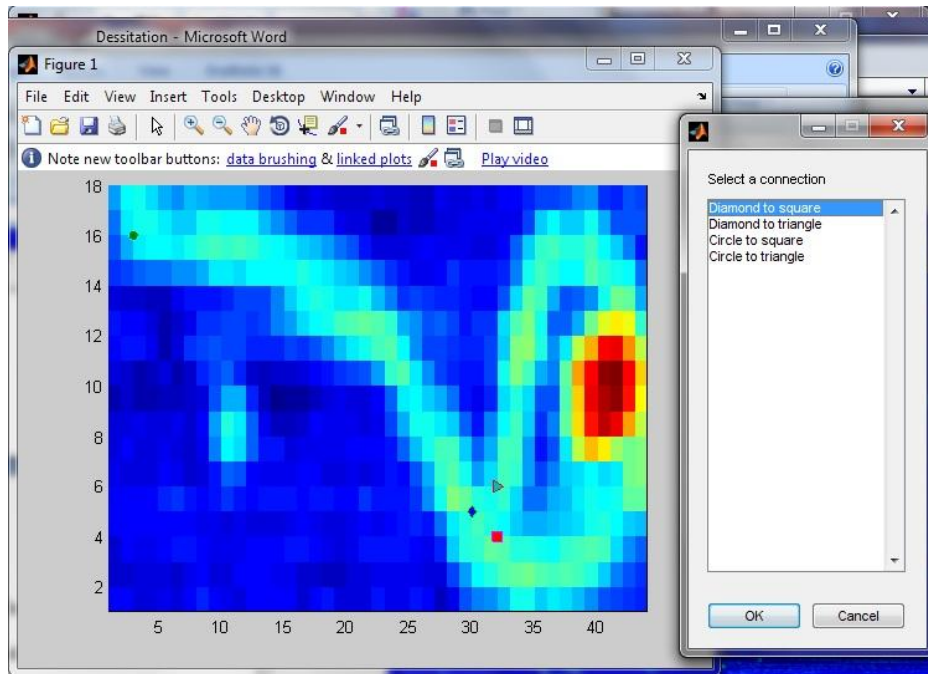
Once the delete function was chosen, matlab will ask the user to select a trace with the mouse. Then the selected trace highlighted in yellow and a dialog window will ask if the user really wants to delete the trace (Fig. 4.11). If the user chooses ‘Yes’, the selected trace will be removed from the data.



*Figure 4.11:*. Delete sub-function. A DNA trace was selected by a click of the mouse. Then the user can delete the trace by clicking “Yes”.

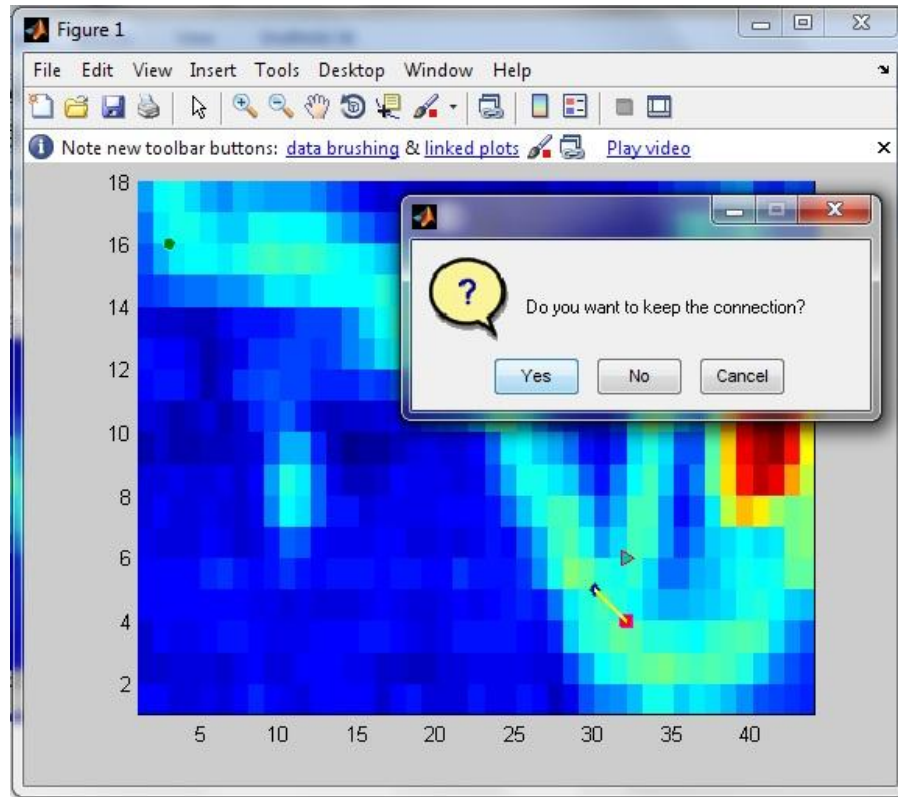
When connect function is chosen, matlab will ask the user to choose two segments with the mouse. Then the program zooms in the trace region and the ends of two segments are highlighted differently (Fig. 4.12). Then the user can choose a connection in the list dialog.





*Figure 4.12:* Connect sub-function. The user first selects two traces by clicking the mouse. Then the four ends of two DNA skeletons will be labeled by a circle, diamond, triangle and square respectively. The user can choose the way that two traces to be connected.

After selection, matlab will connect the two ends with a yellow line (Fig. 4.13). If the user is satisfied with the connection, the matlab routine will connect the two traces together and the user can move on to the next operation or image.



*Figure 4.13:* The two DNA tracing are connected by a yellow line. The user can select “Yes” if he agree with the connection and wants to save it. Otherwise, the user can click “No” and redo the connection.

The mask program also provides a convenient way to get the length of a whole segment or one part of it. Once a segment is selected, the program will calculate the length of the segment in both nanometers and basepairs (fig. 4.14). The mask routine also allows the user to calculate the length of a segment between two points (fig. 4.15).

The matlab code of the mask program can be found in appendix E (maskM.m and maskM.fig).

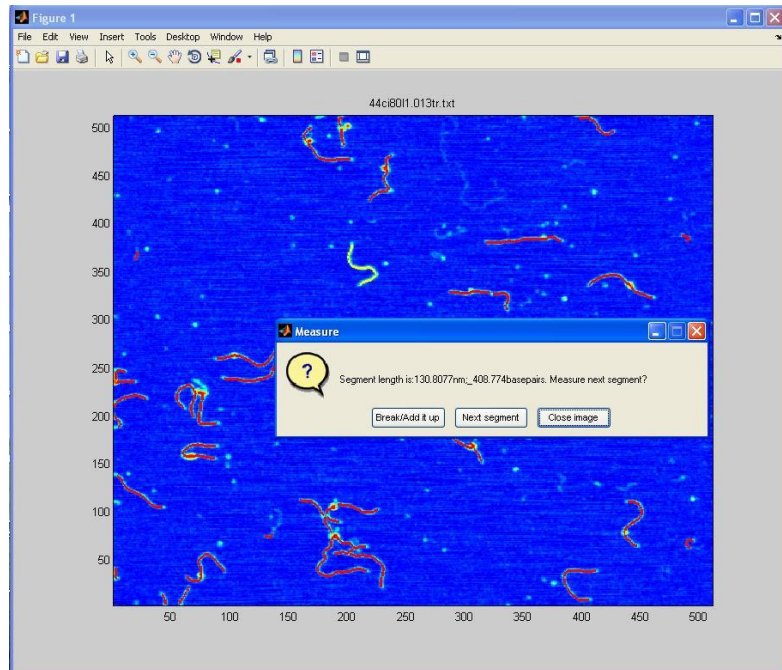


Figure 4.14: User interface of a segment length measurement. A DNA trace was selected and the contour length was displayed on the dialog.

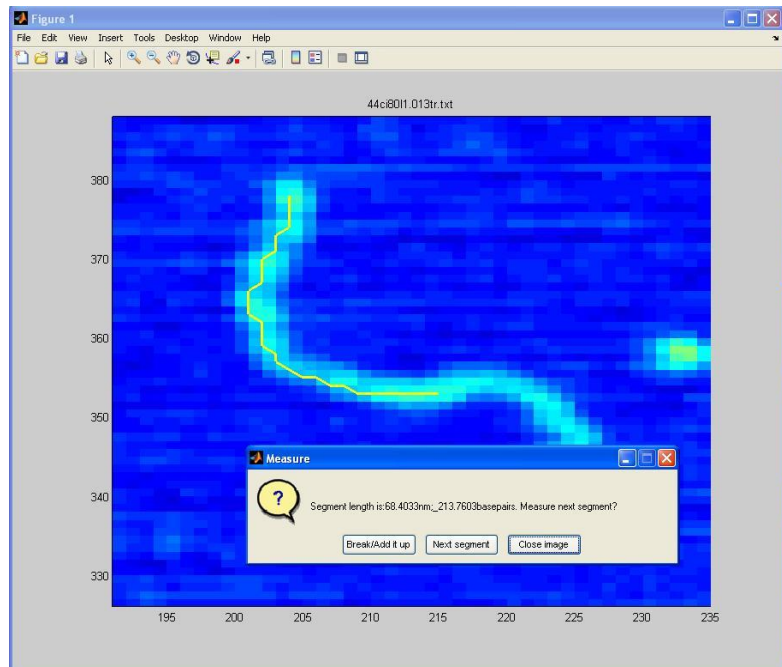
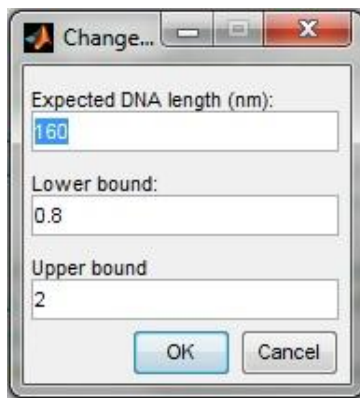


Figure 4.15: Calculating the length of a part in the DNA segment. The user can select a part of one DNA trace by clicking on start and end points. The program will then calculate the DNA length between the two points and display it in the dialog.

### § 4.3.3 DNA contour length

Negative controls are an important part of most experiments. A typical negative control in a DNA AFM study is the measurement of the DNA contour length. Other times this is the object of the study itself because it reveals DNA conformational changes [58]. Therefore, a matlab routine was developed to measure the contour length from DNA skeleton obtained by tracing program.

The program asks the user to input three parameters in a question dialog (Fig. 4.16). The expected length is the expected DNA length calculated knowing the number of basepairs. Because most PCR products contain lots of short fragments or broken DNA segments, AFM samples often contains DNA fragments much shorter than what expected. Therefore, the user can establish the minimum acceptable length. Any DNA segment shorter than this lower bound will be thrown away. Occasionally, there will be few extremely long molecules (traces). They may come from broken plasmids. Although such long traces are very rare, they will offset the program calculation of histogram bin size. Therefore, the user can remove those extremely long traces with an upper bound.



*Figure 4.16:* Dialog box for the measurement of the DNA contour length. The lower bound is calculated by multiplying the expected DNA length by the lower bound number provided by the user. DNA traces shorter than this lower bound will not be considered. The upper bound is calculated in a similar way and DNA traces longer than the upper bound will be disregarded.

Then, the program will ask to select one or multiple trace files and will calculate the contour length of all the traces in the selected files. The result will be displayed as a histogram and kept in a matlab array (Fig. 4.17, 4.18) for further analysis.

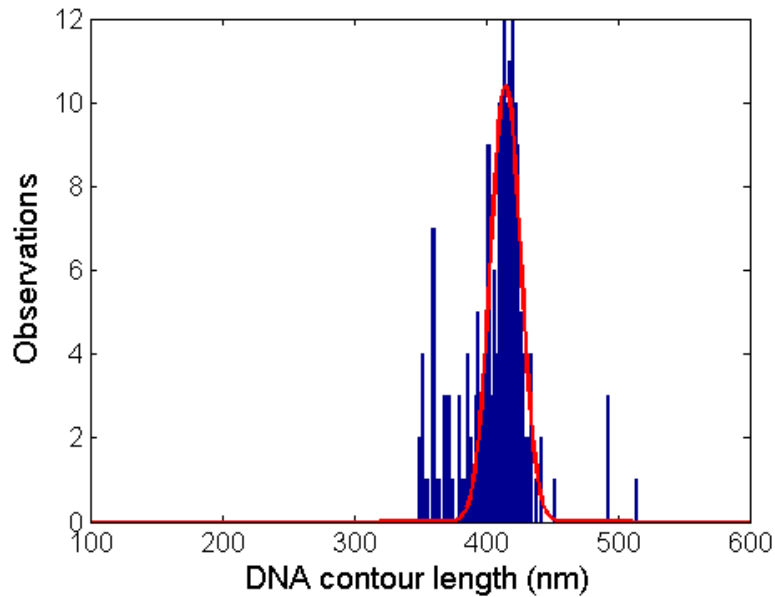


Figure 4.17: DNA contour length of 1394 bp DNA segments. Gaussian fitting shows the DNA length is  $415 \pm 16$  nm.

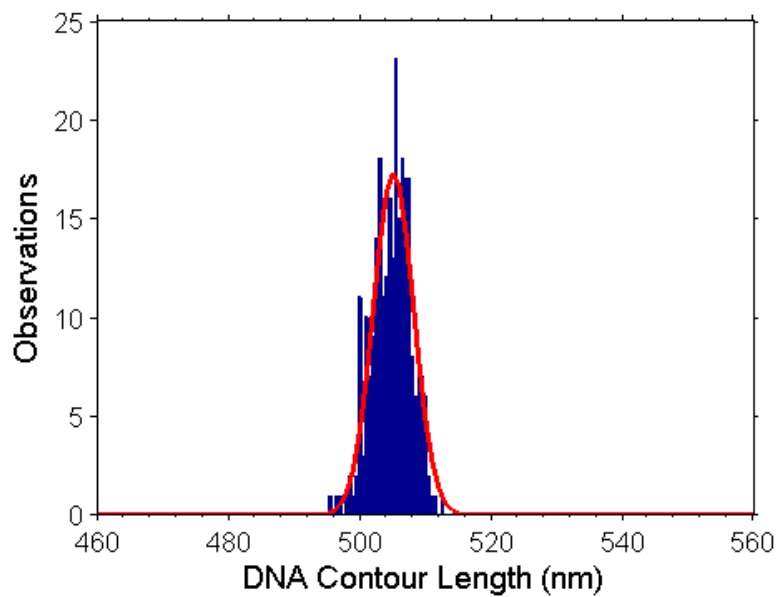


Figure 4.18: Contour length of simulated 1500 bp-long simulated DNA. The peak centered at  $505 \pm 4.3$  nm.

Two types of images were traced by the program to test it. Figure 4.17 is the histogram of the measurement on AFM images of 1394 bp-long DNA deposited on mica surface. Figure 4.18 is the histogram of the measurement on images of 1500 bp-long simulated DNA. The contour length and standard deviation were obtained by fitting the histogram with a Gaussian curve. The final results are summarized in table 4.2.

*Table 4.2:* Comparison of different tracing methods on DNA images acquired by AFM

DNA segments	Original	Neuron J tracing	Automated tracing program
1394 bp real DNA	$1394 \times 0.32 = 446$ nm *	$420.4 \pm 9.1$ nm, from 224 molecules	$415 \pm 16$ nm, from 182 molecules
1500 bp simulated DNA, 300 molecules	$1500 \times 0.34 = 520$ nm	$502.5 \pm 5.6$ nm	$505 \pm 4.3$ nm

\*: The 0.32 nm/bp comes from the tracing of 1584-long, enzyme cut 186 DNA traced with Neuron J.

The matlab code of segment length measurement can be found in appendix E (lengthC.m).

#### **§4.3.4 Automated measurement of particles on the surface or on a DNA molecule**

Protein particle size and volume are important properties that can be assessed by AFM [108-111]. The volume and size of particles sitting on the mica surface or binding to a DNA can be used to determine the protein molecular weight [111], enzyme dimerization [112] and non specific protein-DNA interactions [76].

Because of that, a matlab routine was written to identify and analyze the particles sitting on the surface or binding to a DNA.

Figure 4.19 shows the interface panel of this particle analysis program. There are twelve parameters in this panel which can be either changed or used as a default.

Following are the definitions of those parameters:

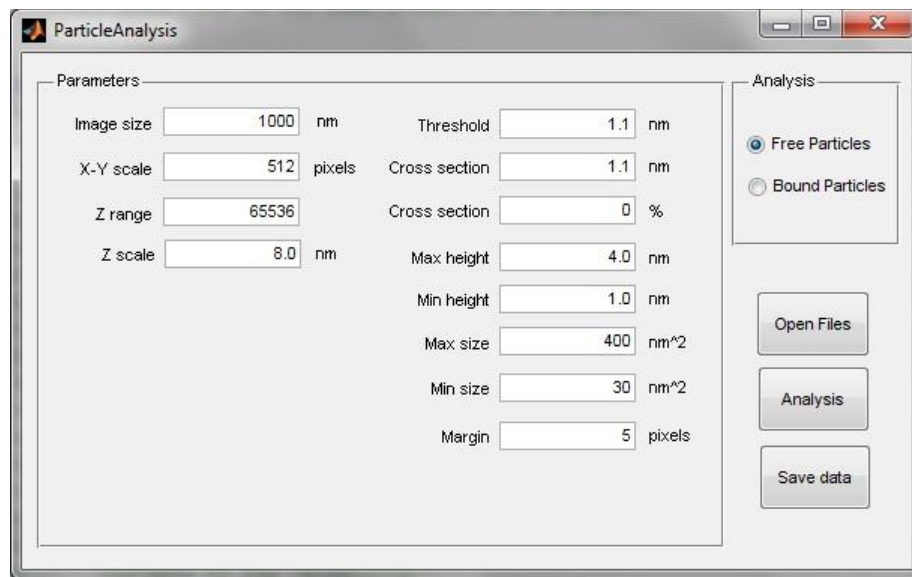


Figure 4.19: User interface panel of the particle analysis program. Users can select to analyze free particles on the surface or only look at the particles binding to a DNA molecule. The values on the left are parameters of imaging. The values on the right are parameters that will be used for analysis purposes.

“Image size” is the size of one AFM image. In our experiments, this value is often equal to 1000 nm which means one image covers 1000 nm×1000 nm of the sample.

“X-Y scale” is the number of pixels on each line or row of image. In our experiments, it is 512.

“Z range” is the range of values used to quantize the height of each pixel. In our experiments, this number is equal to 65536 ( $2^{16}$ ).

“Z scale” is the scale factor of AFM imaging. In our experiment, this value is 8.0 nm.

“Threshold” is the value the program uses to separate the particles from the background.

“Cross section” is the height of a selected cross section. It can be given as the real height or a percentage of the maximum height. If the percentage value is equal to zero, the program will automatically use the real height. The particle diameter will be calculated from the cross section given by this parameter.

“Max height” is the maximum height of the particles that should be considered.

“Min height” is the minimum height of the particles that should be considered.

“Max size” is the maximum area that a particle will cover. If a particle covers more than this size, it will be interpreted as an aggregate of two or more particles and will be thrown away by the program.

“Min size” is the minimum area size below which the particle will be discarded.

“Margin” is the parameter that is used to exclude particles are too close to the image edge.

The user can also choose if to analyze free particles or particles bound to DNA with an interface panel. If the user selects bound particles, the program will automatically disregard particles which do not contact DNA.

After setting these parameters, the user can open one or multiple files by clicking the “Open Files” button and click “Analysis” button to start the analysis.

The program will display the final result in figures as showed in figure 4.20. In the upper left panel, the particles identified by the program were labeled in red. The



lower left panel shows the original AFM figure. The right upper figure is the histogram of the particle diameters. The middle lower panel is the histogram of particle heights. The lower right panel is the histogram of particle volume.

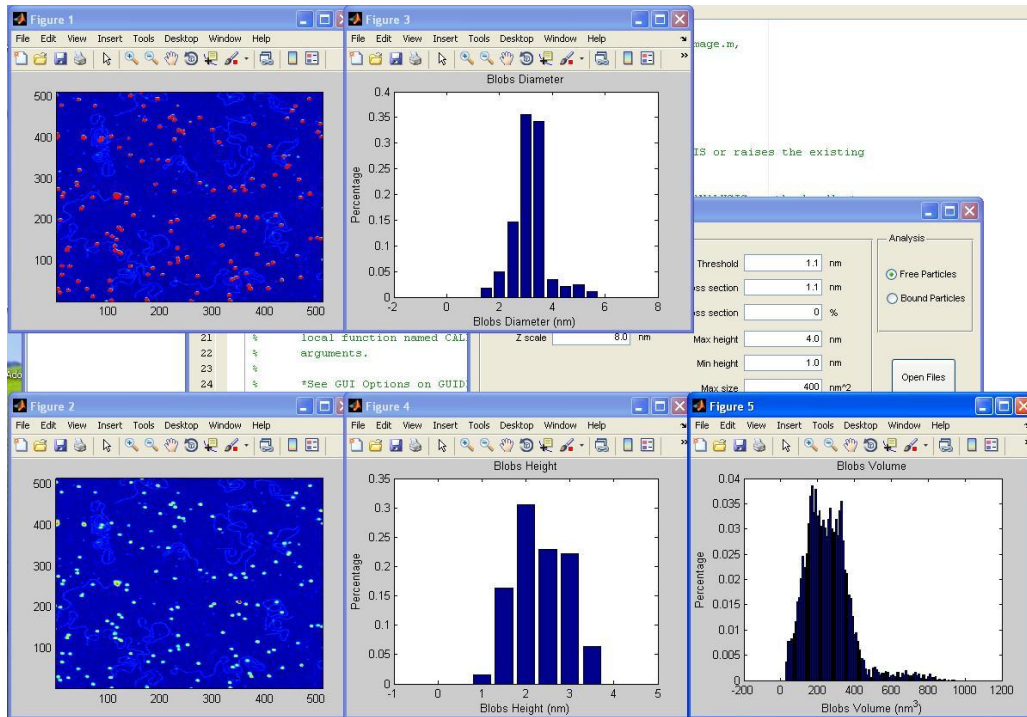


Figure 4.20: Output of particle analysis program. Top left: red blobs are particles identified by the program. Lower left: original AFM image. Top right: the histogram of particle diameter. Lower middle: the histogram of particle height. Lower right: the histogram of particle volume.

The user can also save the data by clicking the “Save Data” button. The program will then save all the data and parameters in an excel file as showed in figure 4.21.

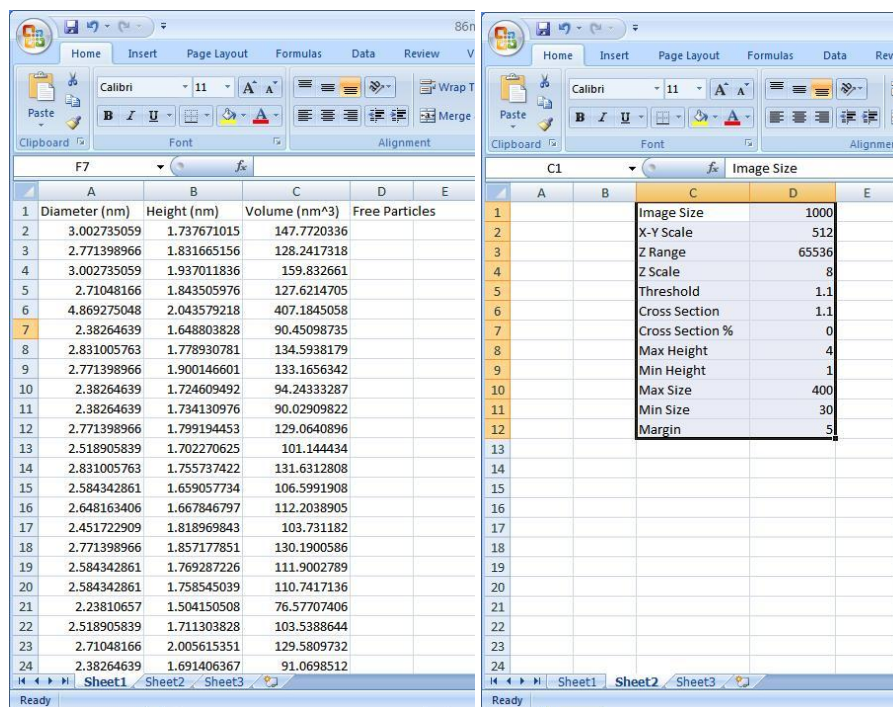


Figure 4.21: Saved excel data of particle analysis. Left figure is the data in columns. The right figure is the working sheet of parameters.

Matlab code of particle measurement can be found in appendix E (ParticleAnalysis.m, ParticleAnalysis.fig).

#### § 4.3.5 Protein-DNA interactions

In some studies, DNA binding proteins such as repressors or RNAP are incubated with DNA before they are deposited onto the mica. In such experiments, DNA may bind, wrap or even loop on these protein particles [65, 76]. Currently, there is no automatic recognition tool for identifying protein-DNA complexes.

Starting from my tracing program, a program that can group DNA skeletons with particles that contact them was developed. The program starts from one trace and groups all the other traces and particles that contact the first trace either directly or

through another particle. Looking at these groups is helpful to characterize the interaction between proteins and DNA segments. After that, the traces and particles can be put into different statistics according to their interaction (binding, looping or wrapping).

Figure 4.22 represents one example of automatic protein-DNA complex analysis. Here images of short fragments containing binding site for lambda repressor were analyzed by the program. The program analyzed over 200 molecules in about 20 minutes, and gave a histogram of the position at which the protein particle contacts the DNA. The result gives the expected position in a much faster time than it would have been possible through a manual analysis.

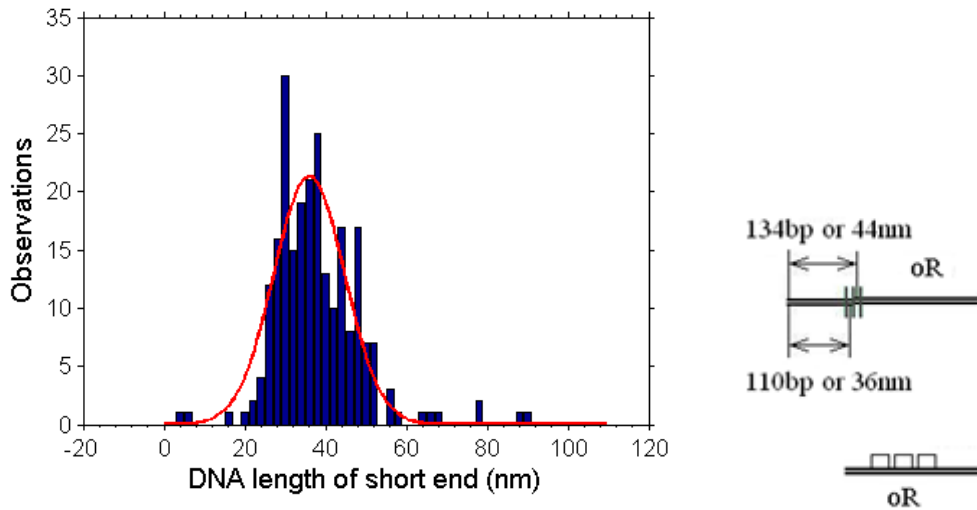


Figure 4.22: Result of protein binding position analysis from over 200 molecules. The histogram represents the DNA length from one end to the particle. The expected value is 34 nm according to the DNA sequence. The fitting result centered at 35 nm.

Matlab code can be found in appendix E (GroupAnalysis.m).

#### **§ 4.3.6 Data conversion**

Since a lot of previous work was done with NeuronJ, a matlab routine was made to convert auto-tracing data to NeuronJ data format. This Matlab code can be found in appendix E (ConvertJ.m).

#### **§ 4.4. Discussion**

AFM is a very powerful technique in the study of biomacromolecules such as protein and DNA. But it is often very time consuming to analyze the images quantitatively. A large number of observations are needed to support a hypothesis or a conclusion. Yet, manual analysis is too slow. Here a toolbox of image analysis programs was developed based on matlab that automate a good part of the analysis and increase considerably its efficiency. This will be of great help to our lab and hopefully to many others.

# References:

1. Ptashne, M.a.G., A., *Genes and Signals*. 2002, New York: Cold Spring Harbor Laboratory.
2. Bird, A., *Perceptions of epigenetics*. *Nature*, 2007. **447**: p. 3.
3. Amasino, R., *Vernalization, Competence, and the Epigenetic Memory of Winter*. *The Plant Cell*, 2004. **16**: p. 8.
4. Marcus E Pembrey, L.O.B., Gunnar Kaati, Soren Edvinsson, Kate Northstone, Michael Sjostrom, Jean Golding and the ALSPAC Study Team, *Sex-specific, male-line transgenerational responses in humans*. *European Journal of Human Genetics*, 2006. **14**: p. 8.
5. Verstrepen, O.J.R.a.K.J., *Timescales of Genetic and Epigenetic Inheritance*. *Cell*, 2007. **128**: p. 14.
6. Chandler, V.L., *Paramutation: From Maize to Mice*. *Cell*, 2007. **128**: p. 5.
7. Zachary A Kaminsky, T.T., Sun-Chone Wang, Carolyn Ptak, Gabriel H T Oh, Albert H C Wong, Laura A Feldcamp, Carl Virtanen, Jonas Halfvarson, Curt Tysk, Allan F McRae, Peter M Visscher, Grant W Montgomery, Irving I Gottesman, Nicholas G Martin and Art Petronis, *DNA methylation profiles in monozygotic and dizygotic twins*. *Genetics*, 2009. **41**: p. 6.
8. Reik, W., *Stability and flexibility of epigenetic gene regulation in mammalian development*. *Nature*, 2007. **447**: p. 8.
9. Ptashne, M., *On the use of the word "epigenetic"*. *Current Biology*, 2007. **17**(7): p. 4.
10. Ian B. Dodd, M.A.M., Kim Sneppen, and Genevieve Thon, *Theoretical Analysis of Epigenetic Cell Memory by Nucleosome Modification*. *Cell*, 2007. **129**: p. 10.
11. Wassenegger, M., *The Role of the RNAi Machinery in Heterochromatin Formation*. *Cell*, 2005. **122**: p. 4.
12. Ingela Djupedal, K.E., *Epigenetics: heterochromatin meets RNAi*. *Cell Research*, 2009. **19**: p. 14.
13. Edmunds, A.Y.a.W.J., *Epigenetic inheritance and prions*. *Journal of Evolutionary Biology*, 1998. **11**: p. 2.
14. William M. Rideout III, K.E., Rudolf Jaenisch, *Nuclear Cloning and Epigenetic Reprogramming of the Genome*. *Science*, 2001. **293**: p. 6.
15. Paul A. De Sousa, T.K., Linda Harkness, Lorraine E. Young, Simon K. Walker, and Ian Wilmut, *Evaluation of Gestational Deficiencies in Cloned Sheep Fetuses and Placentae*. *Biology of Reproduction*, 2001. **65**: p. 8.
16. Fatima Santos, V.Z., Miodrag Stojkovic, Antoine Peters, Thomas Jenuwein, Eckhard Wolf, Wolf Reik, and Wendy Dean, *Epigenetic Marking Correlates with Developmental Potential in Cloned Bovine Preimplantation Embryos*. *Current Biology*, 2003. **13**: p. 6.
17. J.H.M. Knoll, R.D.N., R.E. Magenis, J.M. Graham Jr., M. Lanlande, S.A. Latt, *Angelman and Prader-Willi Syndromes Share a Common Chromosome 15 Deletion but Differ in Parental Origin of the Deletion*. *American Journal of Medical Genetics*, 1989. **32**: p. 6.
18. Jacqueline R Engel, A.S., Antonita Harper, Michael J Higgins, Mitsuo Oshimura, Wolf Reik, Paul N Schofield, Eamonn R Maher, *Epigenotype-phenotype correlations in Beckwith-wiedemann syndrom*. *Journal of Medical Genetics*, 2000. **37**: p. 6.
19. Jack B. Bishop, K.L.W., Richard A. Sloane, *Genetic toxicities of human teratogens*. *Mutation Research*, 1997. **396**: p. 35.

20. Esteller, M., *Epigenetics in Cancer*. The New England Journal of Medicine, 2008. **358**: p. 12.
21. Ptashne, M., *A genetic switch : phage lambda revisited*. 3rd ed. 2004, Cold Spring Harbor, N.Y.: Cold Spring Harbor Laboratory Press. xiv, 154 p.
22. Johnson, A.D., et al., *lambda Repressor and cro--components of an efficient molecular switch*. Nature, 1981. **294**(5838): p. 217-23.
23. Ptashne, M., *Repressors*. Trends in Biochemical Sciences, 1984. **9**(4): p. 142-145.
24. Oppenheim, A.B., et al., *Switches in bacteriophage lambda development*. Annu Rev Genet, 2005. **39**: p. 409-29.
25. Zurla, C., et al., *Novel tethered particle motion analysis of CI protein-mediated DNA looping in the regulation of bacteriophage lambda*. Journal of Physics-Condensed Matter, 2006. **18**(14): p. S225-S234.
26. Lia, G., et al., *Supercoiling and denaturation in Gal repressor/heat unstable nucleoid protein (HU)-mediated DNA looping*. Proceedings of the National Academy of Sciences of the United States of America, 2003. **100**(20): p. 11373-11377.
27. Beausang, J.F., et al., *DNA looping kinetics analyzed using diffusive hidden Markov model*. Biophysical Journal, 2007. **92**(8): p. L64-L66.
28. Adhya, S. and M. Gottesman, *Promoter Occlusion - Transcription through a Promoter May Inhibit Its Activity*. Cell, 1982. **29**(3): p. 939-944.
29. Shearwin, K.E., B.P. Callen, and J.B. Egan, *Transcriptional interference - a crash course*. Trends in Genetics, 2005. **21**(6): p. 339-345.
30. Mazo, A., et al., *Transcriptional interference: an unexpected layer of complexity in gene regulation*. Journal of Cell Science, 2007. **120**(16): p. 2755-2761.
31. Dodd, I.B., K.E. Shearwin, and K. Sneppen, *Modelling transcriptional interference and DNA looping in gene regulation*. Journal of Molecular Biology, 2007. **369**(5): p. 1200-1213.
32. Rocco, V., B. Demassy, and A. Nicolas, *The Saccharomyces-Cerevisiae Arg4 Initiator of Meiotic Gene Conversion and Its Associated Double-Strand DNA Breaks Can Be Inhibited by Transcriptional Interference*. Proceedings of the National Academy of Sciences of the United States of America, 1992. **89**(24): p. 12068-12072.
33. Eszterhas, S.K., et al., *Transcriptional interference by independently regulated genes occurs in any relative arrangement of the genes and is influenced by chromosomal integration position*. Molecular and Cellular Biology, 2002. **22**(2): p. 469-479.
34. Corbin, V. and T. Maniatis, *Role of Transcriptional Interference in the Drosophila-Melanogaster Adh Promoter Switch*. Nature, 1989. **337**(6204): p. 279-282.
35. Minuzzo, M., et al., *Interference of transcriptional activation by the antineoplastic drug ecteinascidin-743*. Proceedings of the National Academy of Sciences of the United States of America, 2000. **97**(12): p. 6780-6784.
36. Lenasi, T., X. Contreras, and B.M. Peterlin, *Transcriptional interference antagonizes proviral gene expression to promote HIV latency*. Cell Host & Microbe, 2008. **4**(2): p. 123-133.
37. Eggermont, J. and N.J. Proudfoot, *Poly(a) Signals and Transcriptional Pause Sites Combine to Prevent Interference between Rna Polymerase-Ii Promoters*. Embo Journal, 1993. **12**(6): p. 2539-2548.
38. Martens, J.A., L. Laprade, and F. Winston, *Intergenic transcription is required to repress the Saccharomyces cerevisiae SER3 gene*. Nature, 2004. **429**(6991): p. 571-574.
39. Wang, P.X., et al., *Demonstration that the TyrR protein and RNA polymerase complex formed at the divergent P3 promoter inhibits binding of RNA polymerase to the major*

- promoter, P1, of the aroP gene of Escherichia coli.* Journal of Bacteriology, 1998. **180**(20): p. 5466-5472.
40. Ackermann, H.W., *Frequency of morphological phage descriptions in the year 2000.* Archives of Virology, 2001. **146**(5): p. 843-857.
  41. Pinkett, H.W., et al., *The structural basis of cooperative regulation at an alternate genetic switch.* Molecular Cell, 2006. **21**(5): p. 605-615.
  42. Shearwin, K.E., I.B. Dodd, and J.B. Egan, *The helix-turn-helix motif of the coliphage 186 immunity repressor binds to two distinct recognition sequences.* Journal of Biological Chemistry, 2002. **277**(5): p. 3186-3194.
  43. Dodd, I.B. and J.B. Egan, *DNA binding by the coliphage 186 repressor protein CI.* Journal of Biological Chemistry, 1996. **271**(19): p. 11532-11540.
  44. Medvedev, Z.A., M.N. Medvedeva, and H.M. Crowne, *Age-Related-Changes of the Pattern of Non-Histone Proteins in Active and Condensed Fractions of Mouse-Liver Chromatin and Hepato-Carcinoma.* Experientia, 1984. **40**(11): p. 1282-1284.
  45. Thakur, M.K., *Age-Related-Changes in the Structure and Function of Chromatin - a Review.* Mechanisms of Ageing and Development, 1984. **27**(3): p. 263-286.
  46. Wolffe, A.P. and D. Guschin, *Chromatin structural features and targets that regulate transcription.* Journal of Structural Biology, 2000. **129**(2-3): p. 102-122.
  47. Bird, A.P. and A.P. Wolffe, *Methylation-induced repression - Belts, braces, and chromatin.* Cell, 1999. **99**(5): p. 451-454.
  48. Tachiwana, H., et al., *Structures of human nucleosomes containing major histone H3 variants.* Acta Crystallogr D Biol Crystallogr, 2011. **67**(Pt 6): p. 578-83.
  49. Zurla, C., et al., *Direct demonstration and quantification of long-range DNA looping by the lambda bacteriophage repressor.* Nucleic Acids Res, 2009. **37**(9): p. 2789-95.
  50. Bakk, A. and R. Metzler, *In vivo non-specific binding of lambda CI and Cro repressors is significant.* Febs Letters, 2004. **563**(1-3): p. 66-68.
  51. Bakk, A. and R. Metzler, *Nonspecific binding of the OR repressors CI and Cro of bacteriophage lambda.* Journal of Theoretical Biology, 2004. **231**(4): p. 525-33.
  52. Senear, D.F. and R. Batey, *Comparison of Operator-Specific and Nonspecific DNA-Binding of the Lambda-CI Repressor - [Kc1] and Ph Effects.* Biochemistry, 1991. **30**(27): p. 6677-6688.
  53. Giessibl, F.J., *Advances in atomic force microscopy.* Reviews of Modern Physics, 2003. **75**(3): p. 949-983.
  54. Hinterdorfer, P. and Y.F. Dufrene, *Detection and localization of single molecular recognition events using atomic force microscopy.* Nature Methods, 2006. **3**(5): p. 347-355.
  55. Zurla, C., et al., *Direct demonstration and quantification of long-range DNA looping by the {lambda} bacteriophage repressor.* Nucleic Acids Res, 2009.
  56. Meijering, E., et al., *Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images.* Cytometry Part A, 2004. **58A**(2): p. 167-176.
  57. Rasband, W.S., *Image J.* 2008.
  58. Rivetti, C. and S. Codeluppi, *Accurate length determination of DNA molecules visualized by atomic force microscopy: evidence for a partial B- to A-form transition on mica.* Ultramicroscopy, 2001. **87**(1-2): p. 55-66.
  59. Marek, J., et al., *Interactive measurement and characterization of DNA molecules by analysis of AFM images.* Cytometry Part A, 2005. **63A**(2): p. 87-93.

60. Ficarra, E., et al., *Automated DNA fragments recognition and sizing through AFM image processing*. Information Technology in Biomedicine, IEEE Transactions on, 2005. **9**(4): p. 508-517.
61. Rivetti, C., M. Guthold, and C. Bustamante, *Scanning Force Microscopy of DNA Deposited onto Mica: Equilibration versus Kinetic Trapping Studied by Statistical Polymer Chain Analysis*. Journal of Molecular Biology, 1996. **264**(5): p. 919-932.
62. Podesta, A., et al., *Positively charged surfaces increase the flexibility of DNA*. Biophysical Journal, 2005. **89**(4): p. 2558-2563.
63. Claudio Rivetti, S.C., *Accurate length determination of DNA molecules visualized by atomic force microscopy: evidence for a partial B- to A-form transition on mica*. Ultramicroscopy, 2001. **87**: p. 12.
64. Koblan, K.S. and G.K. Ackers, *Site-specific enthalpic regulation of DNA transcription at bacteriophage lambda OR*. Biochemistry, 1992. **31**(1): p. 57-65.
65. Yang, Y., et al., *Determination of protein-DNA binding constants and specificities from statistical analyses of single molecules: MutS-DNA interactions*. Nucleic Acids Res, 2005. **33**(13): p. 4322-34.
66. Schneider, S.W., et al., *Molecular weights of individual proteins correlate with molecular volumes measured by atomic force microscopy*. Pflugers Arch, 1998. **435**(3): p. 362-7.
67. Neaves, K.J., et al., *Atomic force microscopy of the EcoKI Type I DNA restriction enzyme bound to DNA shows enzyme dimerization and DNA looping*. Nucleic Acids Res, 2009. **37**(6): p. 2053-63.
68. Ratcliff, G.C. and D.A. Erie, *A novel single-molecule study to determine protein--protein association constants*. J Am Chem Soc, 2001. **123**(24): p. 5632-5.
69. Brenowitz, M.H.a.M., *Comparison of the DNA Association Kinetics of the Lac Repressor Tetramer, Its Dimeric Mutant Lac Iadi and the Native Dimeric Gal Repressor*. The Journal of Biological Chemistry, 1997. **272**(August 29): p. 5.
70. Burz, D.S., et al., *Self-assembly of bacteriophage lambda cl repressor: effects of single-site mutations on the monomer-dimer equilibrium*. Biochemistry, 1994. **33**(28): p. 8399-405.
71. Maniatis, T. and M. Ptashne, *Multiple repressor binding at the operators in bacteriophage lambda*. Proc Natl Acad Sci U S A, 1973. **70**(5): p. 1531-5.
72. Dodd, I.B., et al., *Octamerization of lambda CI repressor is needed for effective repression of P-RM and efficient switching from lysogeny*. Genes & Development, 2001. **15**(22): p. 3013-3022.
73. Dodd, I.B., et al., *Cooperativity in long-range gene regulation by the lambda CI repressor*. Genes Dev, 2004. **18**(3): p. 344-54.
74. Koblan, K.S. and G.K. Ackers, *Site-Specific Enthalpic Regulation Of Dna-Transcription At Bacteriophage-Lambda Or*. Biochemistry, 1992. **31**(1): p. 57-65.
75. Senear, D.F., et al., *Energetics Of Cooperative Protein Dna Interactions - Comparison Between Quantitative Deoxyribonuclease Footprint Titration And Filter Binding*. Biochemistry, 1986. **25**(23): p. 7344-7354.
76. Wang, H., Finzi, L., Lewis, D. and Dunlap, D., *AFM studies of the CI oligomers that secure DNA loops*. J. Pharmaceutical Biotechnology, 2009. **10**: p. 494-501.
77. Shearwin, K.E., A.M. Brumby, and J.B. Egan, *The Tum protein of coliphage 186 is an antirepressor*. Journal of Biological Chemistry, 1998. **273**(10): p. 5708-5715.
78. Schafer, D.A., et al., *Transcription by Single Molecules of Rna-Polymerase Observed by Light-Microscopy*. Nature, 1991. **352**(6334): p. 444-448.



79. Nelson, P.C., et al., *Tethered particle motion as a diagnostic of DNA tether length*. Journal of Physical Chemistry B, 2006. **110**(34): p. 17260-17267.
80. Shearwin, K.E. and J.B. Egan, *Purification and self-association equilibria of the lysis-lysogeny switch proteins of coliphage 186*. J Biol Chem, 1996. **271**(19): p. 11525-31.
81. E. Meijering, M.J., J. C. Sarria, P. Steiner, H. Hirling, and M. Unser, *Design and Validation of a Tool for Neurite Tracing and Analysis in Fluorescence Microscopy Images*. Cytometry A, 2004. **58**: p. 9.
82. Dunlap, D., et al., *Probing DNA topology with Tethered particle Motion*, in *Methods in Molecular biology: Single-Molecule Analysis: Methods and Protocols*, E.J.G. Peterman and G. Wuite, Editors, Humana Press.
83. Finzi, L. and D. Dunlap, *Single-molecule studies of DNA architectural changes induced by regulatory proteins*. Methods Enzymol, 2003. **370**: p. 369-78.
84. Zurla, C., Manzo, C, Dunlap, DD, Lewis, DEA, Adhya, S, Finzi, L, *Direct Demonstration and Quantification of Long-Range DNA looping by the Lambda Bacteriophage Repressor*. Nucleic Acids Res, 2009. **37**: p. 2789-2795.
85. Finzi, L. and D.D. Dunlap, *Single-molecule approaches to structure, kinetics and thermodynamics of transcriptional regulatory nucleoprotein complexes*". J. Biol. Chem., 2010. **285**: p. 18973-18978.
86. Manzo, C. and L. Finzi, *Quantitative analysis of DNA looping kinetics from tethered particle motion experiments*, in *Methods In Enzymology: Molecule Tools, Part B: Super-Resolution, Particle Tracking, Multiparameter, and Force Based Methods*, N.G. Walter, Editor. 2010, Academic Press, Elsevier. p. 199-220.
87. Miller, R., J. Vesenska, and E. Henderson, *Tip Reconstruction for the Atomic Force Microscope*. SIAM Journal on Applied Mathematics, 1995. **55**(5): p. 1362-1371.
88. Egan, K.E.S.a.J.B., *Purification and self-association equilibria of the lysis-lysogeny switch proteins of coliphage 186*. The Journal of Biological Chemistry, 1996. **271**(May 10): p. 7.
89. Pinkett, H.W., et al., *The structural basis of cooperative regulation at an alternate genetic switch*. Mol Cell, 2006. **21**(5): p. 605-15.
90. Ian B. Dodd, K.E.S.a.K.S., *Modelling transcriptional interference and DNA looping in gene regulation*. J. Mol. Biol., 2007. **369**: p. 14.
91. Egan, I.B.D.a.J.B., *Action at a distance in CI repressor regulation of the bacteriophage 186 genetic switch*. Molecular Microbiology, 2002. **45**(3): p. 14.
92. Dodd, I.B., K.B. Shearwin, and K. Sneppen, *Modelling transcriptional interference and DNA looping in gene regulation*. Journal Of Molecular Biology, 2007. **369**(5): p. 1200-1213.
93. Egan, I.B.D.a.J.B., *DNA binding by the coliphage 186 repressor protein CI*. The Journal of Biological Chemistry, 1996. **271**(May 10): p. 9.
94. Lia, G., et al., *Direct Observation of DNA Distortion by the RSC Complex*. Molecular Cell, 2006. **21**(3): p. 417-425.
95. Kasas, S., et al., *Escherichia coli RNA Polymerase Activity Observed Using Atomic Force Microscopy*†. Biochemistry, 1997. **36**(3): p. 461-468.
96. Podestà, A., et al., *Positively Charged Surfaces Increase the Flexibility of DNA*. Biophysical Journal, 2005. **89**(4): p. 2558-2563.
97. Barrett, W.A. and E.N. Mortensen, *Interactive live-wire boundary extraction*. Medical Image Analysis, 1997. **1**(4): p. 331-341.
98. Brugal, G. and J.M. Chassery, *[A new image-processing system designed for densitometry and pattern analysis of microscopic specimen. Application to the*

- automated recognition and counting of cells in the various phases of the mitotic cycle (author's transl)]*. Histochemistry, 1977. **52**(3): p. 241-58.
99. Sanchez-Sevilla, A., et al., *Accuracy of AFM measurements of the contour length of DNA fragments adsorbed on mica in air and in aqueous buffer*. Ultramicroscopy, 2002. **92**(3-4): p. 151-158.
  100. Spisz, T.S., et al., *Automated sizing of DNA fragments in atomic force microscope images*. Med Biol Eng Comput, 1998. **36**(6): p. 667-72.
  101. Ficarra, E., et al., *Automated DNA fragments recognition and sizing through AFM image processing*. IEEE Trans Inf Technol Biomed, 2005. **9**(4): p. 508-17.
  102. Rivetti, C., *DNA contour length measurements as a tool for the structural analysis of DNA and nucleoprotein complexes*. Methods Mol Biol, 2011. **749**: p. 235-54.
  103. Rivetti, C., *A simple and optimized length estimator for digitized DNA contours*. Cytometry Part A, 2009. **75A**(10): p. 854-861.
  104. Lipkin, B.S. and A. Rosenfeld. *Picture processing and psychopictorics*. New York: Academic Press.
  105. Dorst, L. and A.W.M. Smeulders, *Length estimators for digitized contours*. Computer Vision, Graphics, and Image Processing, 1987. **40**(3): p. 311-333.
  106. Kulpa, Z., *Area and perimeter measurement of blobs in discrete binary pictures*. Computer Graphics and Image Processing, 1977. **6**(5): p. 434-451.
  107. Vossepoel, A.M. and A.W.M. Smeulders, *Vector code probability and metrication error in the representation of straight lines of finite length*. Computer Graphics and Image Processing, 1982. **20**(4): p. 347-364.
  108. Rio, D.C., et al., *Analysis of P element transposase protein-DNA interactions during the early stages of transposition*. Journal of Biological Chemistry, 2007. **282**(39): p. 29002-29012.
  109. Minh, P.N.L., et al., *Insights into the architecture and stoichiometry of Escherichia coli PepA•DNA complexes involved in transcriptional control and site-specific DNA recombination by atomic force microscopy*. Nucleic Acids Research, 2009. **37**(5): p. 1463-1476.
  110. Yang, Y., H. Wang, and D.A. Erie, *Quantitative characterization of biomolecular assemblies and interactions using atomic force microscopy*. Methods, 2003. **29**(2): p. 175-187.
  111. Erie, D.A. and G.C. Ratcliff, *A novel single-molecule study to determine protein-protein association constants*. Journal of the American Chemical Society, 2001. **123**(24): p. 5632-5635.
  112. Henderson, R.M., et al., *Atomic force microscopy of the EcoKI Type I DNA restriction enzyme bound to DNA shows enzyme dimerization and DNA looping*. Nucleic Acids Research, 2009. **37**(6): p. 2053-2063.

# Appendices

AFM Studies of  $\lambda$  Repressor Oligomers Securing DNA LoopsHaowei Wang<sup>1</sup>, Laura Finzi<sup>1</sup>, Dale E. A. Lewis<sup>2</sup> and David Dunlap<sup>3,\*</sup><sup>1</sup>Physics Department, Emory University, Atlanta, GA, USA; <sup>2</sup>Laboratory of Molecular Biology, NCI, NIH, Bethesda, MD, USA and <sup>3</sup>Department of Cell Biology, Emory University School of Medicine, Atlanta, GA, USA

**Abstract:** Large, cooperative assemblies of proteins that wrap and/or loop genomic DNA may “epigenetically” shift configurational equilibria that determine developmental pathways. Such is the case of the  $\lambda$  bacteriophage which may exhibit virulent (lytic) or quiescent (lysogenic) growth. The lysogenic state of  $\lambda$  prophages is maintained by the  $\lambda$  repressor (CI), which binds to tripartite operator sites in each of the  $O_L$  and  $O_R$  control regions located about 2.3 kbp apart on the phage genome and represses lytic promoters. Dodd and collaborators have suggested that an initial loop formed by interaction between CI bound at  $O_R$  and  $O_L$  provides the proper scaffold for additional CI binding to attenuate the  $P_{RM}$  promoter and avoid over production of CI. Recently, the looping equilibrium as a function of CI concentration was measured using tethered particle motion analysis, but the oligomerization of CI in looped states could not be determined. Scanning force microscopy has now been used to probe these details directly. An equilibrium distribution of looped and unlooped molecules confined to a plane was found to be commensurate to that for tethered molecules in solution, and the occupancies of specific operator sites for several looped and unlooped conformations were determined. Some loops appeared to be scaled by oligomers of 6-8, most by oligomers of 10-12, and a few by oligomers of 14-16.

**Keywords:** Lambda repressor, DNA looping, Atomic force microscopy.

## INTRODUCTION

From viruses to humans, transcription is regulated by proteins that bind the DNA. It is becoming increasingly clear that, in most cases, genes are controlled by large, cooperative assemblages of proteins that wrap and loop the DNA. These protein-induced configurational changes often represent real “epigenetic switches” in which shifting the equilibrium towards one configuration versus the other commits the system to one developmental pathway instead of another. Such is the case of the  $\lambda$  bacteriophage and, it is suspected, of most temperate bacteriophages which may adopt a quiescent lifestyle (the lysogenic growth) or a virulent lifestyle (the lytic growth). After infection, repressor protein often binds to multipartite operators and mediate cooperative, long-range interactions which repress the lytic genes maintaining a stable lysogenic state, until adverse environmental conditions (DNA damage, poisoning, starvation, etc) induce a cascade of events that leads to repressor dissociation from the double helix and efficient switch to lysis. The  $\lambda$  epigenetic switch is not only a paradigm of transcriptional regulation, but is also at the basis of our understanding of phage lysogeny [1].

The lysogenic state of  $\lambda$  prophages is maintained by the  $\lambda$  repressor, or CI protein [2]. During lysogeny, dimers of CI bind to the  $O_L$  and  $O_R$  control regions, located about 2.3 kbp apart on the phage genome and repress the  $P_L$  and  $P_R$  promoters for the lytic genes. Each control region contains three binding sites for CI,  $O_{L1}$ ,  $O_{L2}$ ,  $O_{L3}$  and  $O_{R1}$ ,  $O_{R2}$ ,  $O_{R3}$  [3-5]. CI binds to these operators with an intrinsic affinity  $O_{L1} >$

$O_{R1} > O_{L3} > O_{L2} > O_{R2} > O_{R3}$  [6, 7]. By studying the  $O_L$  and  $O_R$  regions separately and in isolation from the rest of the  $\lambda$  chromosome [8], it was found that pairs of dimers interact when bound to adjacent or nearby operators, forming tetramers. These cooperative interactions improve the specificity and strength of CI binding to  $O_{R1}$  and  $O_{R2}$ , and  $O_{L1}$  and  $O_{L2}$ , respectively, so that the binding affinity ranking becomes  $O_{R1} > O_{L1} > O_{R2} > O_{L2} > O_{L3} > O_{R3}$ . Biochemical and genetic studies have identified the contacts between amino acids in the C-terminal domain that mediate these interactions. These contacts have been confirmed from the crystal structure of the isolated CTD tetramer [9] and are thought to contribute significantly to the stability of lysogeny. Occupancy of  $O_{R2}$  by CI also activates transcription of the CI gene from the  $P_{RM}$  promoter, constituting a positive auto-regulatory mechanism [10-12] to generate the amount of CI required for repression of the lytic genes, as described above. At very high concentrations, CI was observed to also bind to  $O_{R3}$  and repress its own transcription from  $P_{RM}$  [13]. This negative auto-regulation has been suggested to be important to prevent excessive accumulation of repressor to facilitate efficient switching to lysis when necessary. However, such negative regulation did not seem possible at physiological concentrations and the role of both  $O_{L3}$  and  $O_{R3}$  remained controversial.

Recently, it was suggested that CI molecules bind cooperatively not only to adjacent sites, but also to sites separated by over 2000 bp in the  $\lambda$  genome, inducing a regulatory loop in the phage DNA. This led to the hypothesis that the loop is first formed by interaction between two tetramers bound at  $O_{R1}$ - $O_{R2}$  and  $O_{L1}$ - $O_{L2}$ , respectively. This octamer-mediated loop brings  $O_{L3}$  and  $O_{R3}$  into juxtaposition and favors their occupancy by CI dimers which can interact “head-to-head” in a long-range cooperative fashion, and lead to a CI oc-

\*Address correspondence to this author at the Emory University School of Medicine, Dept. of Cell Biology, Whitehead Bldg. Rm 475, Atlanta, Georgia 30322, USA; Tel: (404)727-3951; Fax: 404-727-6256; E-mail: ddunlap@emory.edu

tamer+tetramer-mediated loop. According to this hypothesis, the loop provides the right scaffold for CI binding to weak  $O_R3$  at lysogenic concentrations [3, 14, 15] and effective repression of  $P_{RM}$ .

To evaluate this hypothesis, the stoichiometry of CI securing the regulatory loop in wild-type lambda DNA is a fundamental piece of information that has not been previously reported. We used atomic force microscopy, AFM, to image the CI-mediated loop and characterize the looping probability and the stoichiometry of the protein closure. Solutions of DNA and repressor were deposited on a flat, positively charged surface, rinsed and dried, and imaged using scanning force microscopy. In the resulting topographs, discrete bumps corresponding almost exclusively to specifically bound protein were found on both looped and unlooped DNA molecules. A looping equilibrium commensurate with that measured using tethered particle motion was measured as well as the volumes of individual protein particles securing loops between  $O_L$  and  $O_R$ . Virtually no CI tetramers were found to secure DNA loops. Instead higher order oligomers of 6-8 and especially 10-12 accounted for the majority of the CI particles associated with DNA loops. The data are consistent with the model in which multipartite operators collect CI dimers that multimerize to stabilize loops formed through random encounters between  $O_L$  and  $O_R$ .

## MATERIALS AND METHODS

1555 bp DNA fragments were produced by PCR amplification of segments of plasmids pDL944 and pDL965 using 5'-CGCAATTAATGTGAGTTAGCTCACTCATTAGGCA CCCAGGC-3' and 5'-GCATTGCTTATCAATTGTTGC AACGAACAGGTCACTATCAGTC-3' as forward and reverse primers. These fragments contained respectively wild-type or mutant lambda operator regions ( $O_L$  and  $O_R$ ) and including the associated promoters  $P_L$ ,  $P_{RM}$  and  $P_R$ . The distance between the midpoints of operator sites  $O_L3$  and  $O_R3$  was 393 bp. pDL965 contains CC to AT mutations in  $O_L3$  and  $O_R3$ , which abrogate CI binding (Lewis *et al.* manuscript in preparation and [16]). PCR using the same plasmid templates was also used to generate 505 or 392 bp DNA fragments that contained only one group of binding sites ( $O_R$  or  $O_L$ ).

Another 732 bp DNA fragment containing two high affinity lac operators  $O_d$  (5'-TGTGAGCGCTCACA-3') and  $O_I$  (5'-AATTGTGAGCGGATAACAATT-3') [17, 18] separated by 70 bp was provided by Opher Gileadi (Quantomix Ltd, Rehovot, Israel). It was produced by PCR using the plasmid *pOid-OI* from the Müller-Hill laboratory as a template and 5'-GCCACCTCTGACTTAAGCGTCG-3' and 5'-TTGAGGGGACGTCGACAGTATC-3' as forward and reverse primers.

The wild-type CI protein (7.25  $\mu\text{g}/\mu\text{l}$ ) was purified from pEA305 in the laboratory of Sankar Adhya. 20 nM CI and 2 to 4 nM DNA were gently mixed in a buffer containing 50 mM HEPES, 150 mM NaCl and 0.1 mM EDTA (pH 7.0) and incubated at RT for 10 min. Shortly before deposition, a 10  $\mu\text{l}$  drop of 0.1  $\mu\text{g}/\text{ml}$  poly-L-ornithine (1 kDa MW, product #P5666, Sigma-Aldrich, St. Louis, MO) was incubated on freshly cleaved mica for one minute at RT. The poly-L-ornithine-coated mica was then washed with 0.4 ml HPLC

water and dried with compressed air. Then 5  $\mu\text{l}$  of the solution containing DNA and protein was quickly diluted with 40  $\mu\text{l}$  of buffer, and a 10  $\mu\text{l}$  droplet of this solution was deposited on the poly-L-ornithine-coated mica and incubated for one minute at RT. The droplet was rinsed away with 0.4 ml HPLC water and dried gently with compressed air. The sample was left overnight in a dessicator at RT before imaging.

Images were acquired with a NanoScope MultiMode AFM microscope (Digital Instrument, Santa Barbara, CA) operated in tapping mode using a 50-60 mV oscillation amplitude of uncoated, etched silicon tips with a resonance frequency of 75 kHz (NSC18, MirkoMasch, San Jose, CA). Areas of  $1 \times 1 \mu\text{m}^2$  were scanned at a rate of 1.2 Hz and a resolution of  $512 \times 512$  pixels.

After filtering images to remove scan line offsets and bowing, DNA molecules were interactively traced with NeuronJ [19], a plug-in function for ImageJ [20]. To measure the volume of protein particles, a basal threshold was established above (typically 0.08 nm) the background. The mean value of all pixels below this threshold was calculated and used as the base for the measurement. The volumes of isolated protein particles were determined as the sum of the pixel heights above the base within the area of the particle protruding above the basal threshold. For DNA-bound protein particles, a second "DNA" threshold was chosen just above the DNA. The volume of protein particles was determined as the sum of the pixel heights above the base within the area of the particle protruding above the "DNA" threshold.

## RESULTS AND DISCUSSION

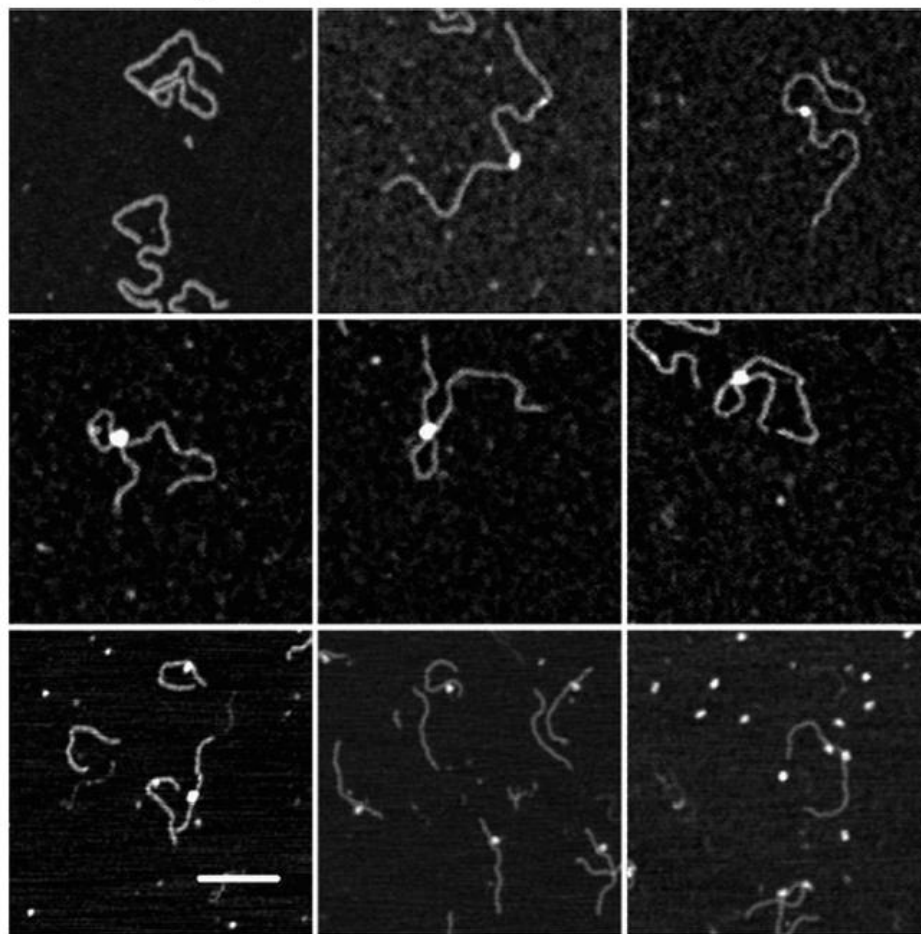
Dried, 1555 bp-long DNA molecules (Fig. (1), upper and middle rows) containing both  $O_L$  and  $O_R$  averaged 510 nm in length in scanning force micrographs. The measured pitch of the DNA on poly-L-ornithine was therefore 0.327 nm/bp which is quite close to that of the B-form structure [21].

### Specific Binding to Operator Sites

To assay the specificity of CI binding, the positions of CI particles were measured along unlooped DNA. Fig. (2) shows schematic diagrams of the molecules used, along with the positions of the right and left operator regions. The positions of the center of bound CI particles were measured and frequency distributions are shown for DNA containing both wild-type operator regions (Fig. (1), upper center and left; Fig. (2), middle-left). There was almost no non-specific binding with the vast majority of particles near the  $O_L$  and  $O_R$  regions located 118 or 265 nm from one end of each molecule.

### Weak Affinity for the $O_R3$ Operator Site

The peak at  $O_L$  was noticeably broader than that corresponding to  $O_R$ . It is well accepted that CI dimers on adjacent operator sites may bind cooperatively; two dimers could occupy either  $O1$  and  $O2$  or  $O2$  and  $O3$ . Given the experimentally determined affinities of the operator sites [22], this is likely to have occurred at  $O_L$  but not  $O_R$ , because the affinity of CI dimers for  $O_L3$  is greater than that for  $O_R3$ . This interpretation was supported by experiments using DNA



**Fig. (1).** AFM images of CI and DNA: (upper left) 1555 bp DNA containing  $O_L$  and  $O_R$ , (upper middle and right) CI protein bound to 1555 bp DNA, (middle row) CI-mediated loops in 1555 bp DNA, (bottom left) CI bound to DNA containing  $O_L$  (wild-type), (bottom center) CI protein bound to DNA containing  $O_L$  ( $O3^-$ ), (bottom right) *lac* repressor bound to  $O_L$  and  $O_I$  containing DNA. The white bar represents 100 nm.

with mutations in the third binding sites ( $O3^-$ ) that abrogated the binding of CI dimers to  $O_L3$  and  $O_R3$ . As in the case of the wild-type DNA, CI binding to the  $O_R$  region of  $O3^-$  DNA produced a narrow peak at 119 nm (Fig. (2), bottom-left). However, with respect to this peak, CI binding to the  $O_L$  region of  $O3^-$  DNA shifted to give a narrow peak at 275 nm, in which the cooperative binding of CI to  $O_L2$  and  $O_L3$  seemed to have disappeared.

To further demonstrate the weak affinity for the  $O_R3$  site, experiments were done with short fragments containing either  $O_R$  or  $O_L$  (Fig. (1), bottom left and center). In histograms of particle locations on the wild-type  $O_L$  containing fragment, there are two peaks separated by 9.5 nm (Fig. (2), middle-center). This distance is slightly larger than the value expected for cooperatively bound dimers bridging either sites  $O_L1$  and  $O_L2$  or  $O_L2$  and  $O_L3$  (20 bp or 6.7 nm). However, the peak located at 47 nm, which corresponds to the  $O_L3$  site,

disappeared for DNA with the  $O_L3^-$  mutation (Fig. (3), bottom-center) while the peak at  $O_R$  (32 nm) remained unchanged (Fig. (2), compare middle-right and bottom-right). The simplest interpretation is that no significant binding to  $O_R3$  occurred with or without mutation while  $O_L3$  binding was observed only for the wild-type operator.

#### Multiple Operators May Recruit Dimers

Among the hundreds of molecules in the recorded topographs, there were a few DNA molecules with small protein particles bound in adjacent positions that were commensurate with the distance between the  $O_I$  and  $O_3$  operator sites (Fig. (3)). Based on the calibration that was performed and is described below, these particles with a mean volume of 174 nm<sup>3</sup> were identified as CI oligomers of 2-4 monomers. According to the DNA construct, the center-to-center distance from  $O_L1$  to  $O_L3$  is 44 bp which corresponds to 14.7 nm and

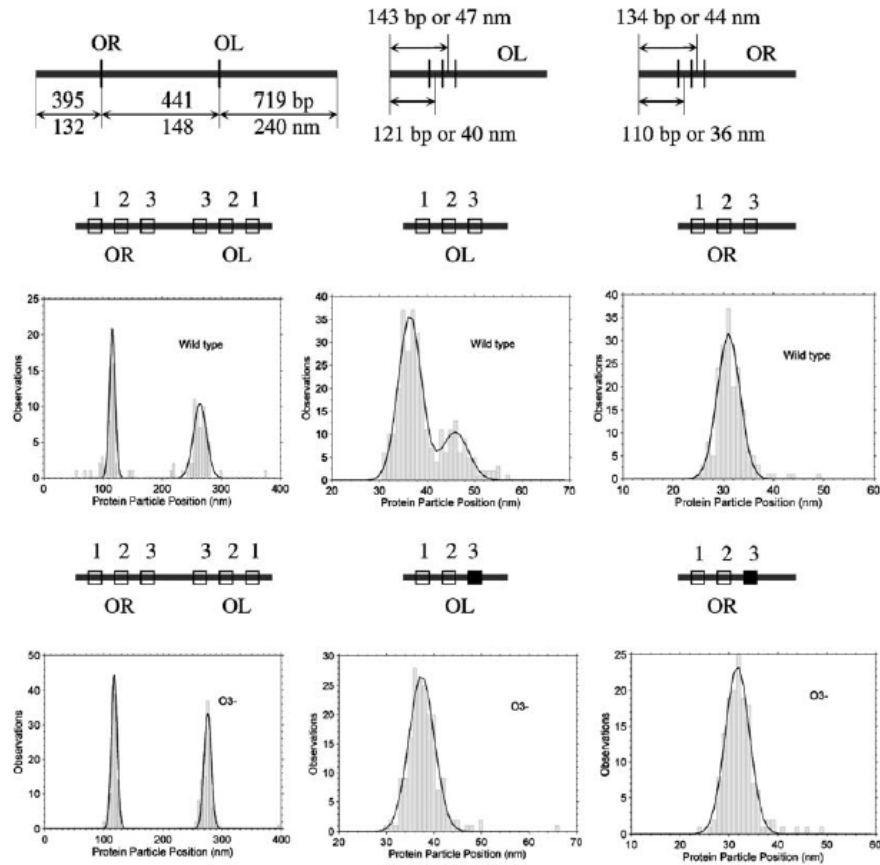


Fig. (2). AFM measurements of the positions of CI particles bound to DNA. Schematic diagrams of the DNA constructs with wild-type (open square) or O3- (black square) operators in the  $O_L$  and  $O_R$  regions appear just above histograms of the AFM-determined positions of CI particles bound to the indicated DNA fragments.

that for  $O_R1$  to  $O_R3$  is 47 bp (15.7 nm). Since the distance between pairs of adjacent particles found in the  $O_R$  or  $O_L$  region was 15.4 and 14.0 nm respectively, the experiment indicated non-cooperative binding of CI the  $O_L$  and  $O_3$  binding sites. These observations suggested that perhaps the presence of three operator sites in each region enhances the probability of capturing CI dimers such that sufficient numbers of proteins accumulate and stand ready to secure a loop when a random collision between  $O_R$  and  $O_L$  occurs. However, one cannot exclude that these species might have been looped molecules that did not survive deposition and washing during sample preparation.

#### Looping Equilibrium

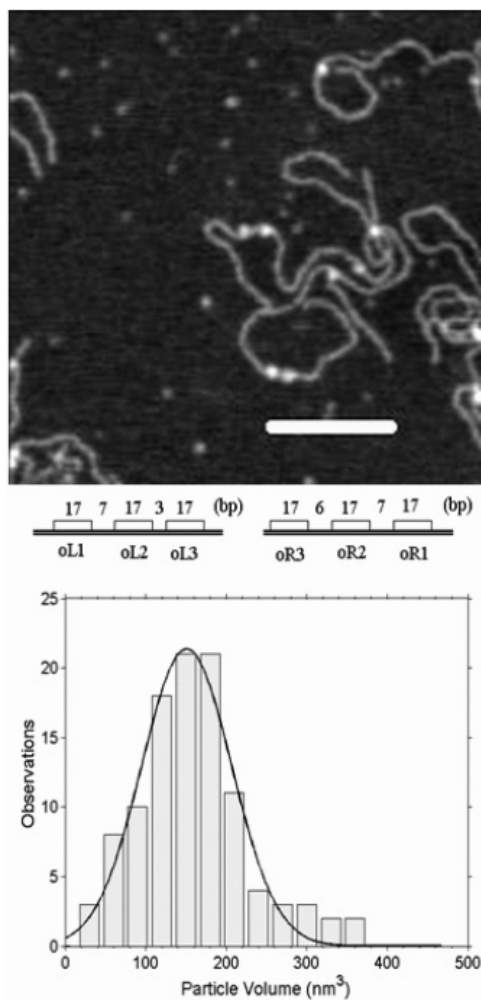
Indeed, the deposition process was reported to affect the measured equilibrium for protein-DNA complexes with 3D topology that distorts upon binding to the surface [23]. Although the operator sites to which CI binds to secure the

DNA loop lie closely spaced, there is a slight helical shift between the left and right  $O_2$  and  $O_3$  operator sites. This might add some three-dimensionality to a looped structure. However scoring 884 or 354 molecules with specifically bound CI particles as either “looped” or “unlooped” for wild-type or O3- DNA at a 20 nM concentration of CI led to 43.9 and 17.8% estimated looping probabilities respectively (Table 1). These are fairly close to the 40 and 10% probabilities measured using tethered particle motion for wild-type DNA segments in the presence of the same low concentration of CI [16]. Successful measurement of the looping equilibrium suggested that the molecular species in the AFM images were relevant to CI-mediated looping and should be characterized further.

#### Volume Calibration

Given the possibility for oligomerization of CI, the number of CI dimers securing a DNA loop may play an impor-

tant role in the dynamics of loop formation. However, there are few experimental methods apart from direct visualization with which to determine this oligomerization on looped molecules. AFM is well suited for this type of analysis, since the volume of the particle at the closure of a DNA loop can be measured directly in the topographs. Of course a calibration to relate the measured volume to the molecular weight, and hence the oligomerization of the protein, is essential.



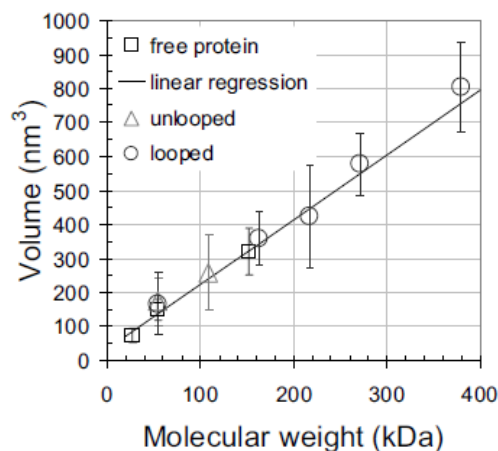
**Fig. (3).** Pairs of CI particles bound to adjacent  $O_{R1}$  and  $O_{R3}$  sites were observed in AFM images (*upper*). The scale bar represents 100 nm. (*lower*) The mean volume of these particle was  $174 \pm 70$  nm<sup>3</sup>.

Several calibration curves have been produced previously for tapping mode images of proteins with both silicon nitride [24] and etched silicon probes [25, 26]. Both the convolution of the probe shape and the compression that results from the tapping force affect the relationship, and linear fits to volume

**Table 1.** Percentages of CI-Mediated Loops in Wild-Type and O3- DNA Molecules Visualized Using AFM

	Wild-Type	O3
Number of molecules	884	354
% Looped	43.9%	17.6%

vs. molecular weight calibrations have slopes ranging from 1.2 to 1.75 for probes with spring constants near 40 N/m and area thresholds set low or at half-height. For the experiments reported here, *lac* repressor (*lacI*) was a convenient reference which maintains a tetrameric state both free and bound to the DNA [27] while free CI was expected to partition into a 7:1 ratio of monomeric and dimeric forms at a concentration of 20 nM. The distributions of protein particles measured for CI and *lacI* without DNA exhibited peaks at 75, 150 and 320 nm<sup>3</sup> (supplementary Figs. (S1 and S2)). For the etched silicon probes with a 3.5 N/m spring constant that were used in these experiments, a calibration considering monomeric and dimeric CI and tetrameric *lacI* proteins deposited on poly-L-ornithine-coated-mica gave a slope of 1.9 (Fig. (4)). This higher value most likely reflects both the softer cantilever which reduces compression and the low threshold used to delimit the area of individual proteins.



**Fig. (4).** AFM measurements of the volume of protein particles both free and bound to DNA. Standard deviations are indicated for all points. Linear regression of volume measurements of unbound  $\lambda$  and *lac* repressor proteins (dark squares) gave the calibration line (dark). The volumes of CI protein particles were measured on unlooped (grey triangles) and looped (grey circles) DNA and CI oligomerization values were assigned to the nearest dimer multiple using the calibration line. Numerical values for the plot are given in supplementary Table S3.

The volumes of *lacI* and CI oligomers bound to DNA were also measured. The *lacI* DNA contains two *lac* repressor binding sites,  $O_{Ld}$  and  $O_I$ . The specificity of particle binding was verified by tracing DNA segments as described for the CI data shown in Fig. (2). The average volume of



particles binding on linear DNA was  $355 \pm 73 \text{ nm}^3$ . Since *lac* repressor was expected to remain tetrameric in the conditions of the experiment (5 nM) [27], this volume was associated with an oligomer weighing 155 kDa. The difference between the measured volumes for protein free and bound to the DNA was about 30 nm which corresponds well to the volume of a segment of DNA as long as the *lacI* binding site, 21 bp.

The average volume of CI particles on unlooped DNA measured  $259 \text{ nm}^3$ . Employing the calibration curve and considering that the molecular weight of CI monomer is 26-28 kDa [28, 29] indicated that the average particles in the experiment could have corresponded to CI tetramers ( $240 \text{ nm}^3$  from the calibration curve). Of course the standard deviation of these measurements was larger than those of *lac* repressor, because the  $\lambda$  operator regions contain three adjacent binding sites, so that several stoichiometries of CI binding were possible. In fact some higher molecular weight particles were observed that are difficult to reconcile with the idea that a looped DNA scaffold is required to promote "head-to-head" binding between CI tetramers to give octamers [14, 30]. One interpretation is that specific binding nucleated adjacent non-specific binding.

**Loop Closures are Prevalently Oligomers 10-12**

Similarly large volume, high molecular weight CI particles were commonly found securing looped DNA molecules. In Fig. (5), the lower panel shows measurements of DNA segments corresponding to the length: from one end to the  $O_R$  site, of the loop (OL-OR), and from  $O_L$  to the other end of the DNA. The narrowly distributed measurements and the good correspondence with the expected values based on the DNA construct indicated loops secured by specifically bound CI. The volumes of these CI particles were distributed as shown in the upper panel of Fig. (5). The curve exhibits three central peaks in the distribution that roughly correspond to oligomers of (from right to left): 6-8, 10-12, and 14-16. This interpretation was developed using the calibration shown in Fig. (4) and assigning molecular weights to the nearest multiple of a dimer, since CI binds DNA as a dimer. The right-most and leftmost peaks were negligibly small and were not considered further.

Oligomers of 10-12 monomers were observed most frequently securing loop closures. Such oligomers would nearly or fully saturate the operator sites in the juxtaposed  $O_L$  and  $O_R$  regions and are consistent with the loop stabilization conferred by "octamer+tetramer" protein binding that was also found using modeling of tethered particle motion data [16]. A significant number of oligomers of 6-8 monomers were also observed at loop closures, but very little tetrameric CI, which corresponds well with the weaker loop stabilization afforded by oligomers lacking contacts between  $O_3$  regions [16]. Oligomers of more than 12 monomers constituted a minor fraction which suggested that CI specifically bound to operators in one region might nucleate adjacent binding of non-specifically bound CI. These additional CI dimers might further stabilize the closure through interaction with corresponding dimers from the opposite region.

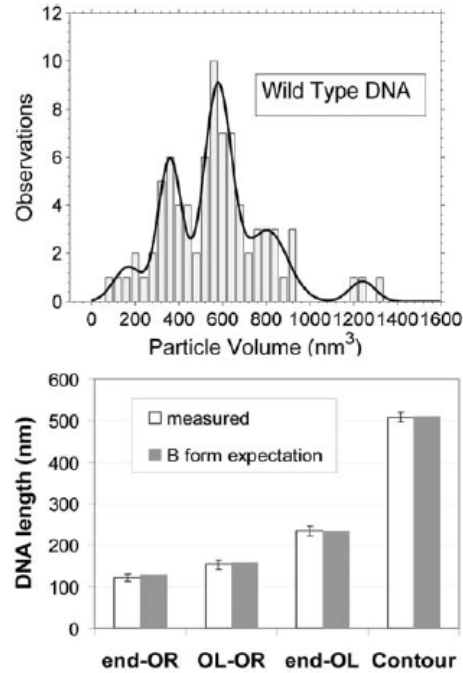


Fig. (5). Oligomerization of CI securing DNA loops. (upper) AFM measurements of the volumes of single CI particles securing DNA loops. (lower) The lengths of segments in the looped DNA correspond well with those expected from the design of the construct.

**Alternative Loop Closures**

A small number of DNA loops (3.2%) contained two adjacent CI particles (Fig. (6)). The average volume of these particles was  $425 \text{ nm}^3$  which identified them as CI octamers. By tracing the DNA in a subset of particularly distinct two-particle-loops, two types of conformer were established. One type was modeled with directly juxtaposed operators in which one octamer apparently included four specifically bound dimers at  $O_1$  and  $O_2$  (or  $O_2$  and  $O_3$ ), and another consisting of two specifically bound CI dimers at  $O_3$  (or  $O_1$ ) flanked by two non-specifically bound dimers to form a second octamer (Fig. (6c)). Whether non-specifically bound dimers preferentially flanked  $O_1$  or  $O_3$  could not be determined. The other type of conformer was modeled with staggered  $O_R$  and  $O_L$  regions leaving  $O_{R3}$  unoccupied (Fig. (6b)) and CI oligomers bridging non-specific sites adjacent to  $O_{R1}$ . Table 2 shows the results of measuring segments in these looped molecules as schematically shown in Fig. (6d). For such a small number of cases, statistically significant differences could not be established, but, as expected from the schematic diagrams, segments *a* and *e* were longer in the directly juxtaposed conformation while *c* was longer in the staggered conformation. These few conformers might represent early intermediates in the looping process that result from collisions between  $O_L$  and  $O_R$  regions that are nearly saturated with CI dimers. Such intermediates may include CI tetramers that bind "semi-specifically" between  $O_{L1}$  and a non-specific site adjacent to  $O_{L1}$ . Subsequent shifting to create complete juxtaposition of all of the specific operators

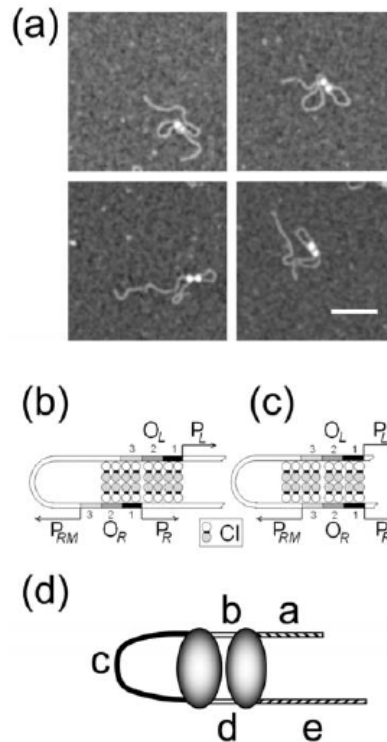
would be expected to increase the stability of the loop and sterically repress the CI promoter,  $P_{RM}$ , near  $O_{R3}$ .

### CONCLUSIONS

The data in this report strongly support the idea that CI binding to  $O3$  operators greatly stabilizes looping of  $\lambda$  DNA fragments. Overwhelmingly specific binding was exhibited by 20 nM CI protein to the  $\lambda$  operator sites. The intrinsic order of this binding,  $O_{L1} > O_{R1} > O_{L3} > O_{L2} > O_{R2} > O_{R3}$ , [6, 7] changes to  $O_{R1} > O_{L1} > O_{R2} > O_{L2} > O_{L3} > O_{R3}$  when cooperative interactions are considered, and this cooperative ranking was reflected in the slight shift of positions of CI particles on unlooped DNA upon mutation of the  $O_{L3}$  but not the  $O_{R3}$  operators. The strong affinity of the polyamine-coated mica for DNA preserved the looped-unlooped equilibrium of the DNA-protein complexes to permit relevant measurements of the protein oligomerization. The volumes of particles securing DNA loops corresponded most frequently to CI oligomers of 10-12, less often to oligomers of 6-8 and occasionally to oligomers of 14-16 that likely include non-specifically bound CI. This underscores the important role of the  $O3$  binding sites in loop stabilization. Finally, rare observations of dimers bound to adjacent operators, and adjacent CI octamers securing specific loops suggest that the tripartite binding sites in the operator regions enhance the targeting of CI to promote efficient looping and transcriptional repression at low protein concentrations.

### ACKNOWLEDGEMENTS

This work was supported by an Emory University graduate student scholarship (HW), Emory University (LF), and the laboratory of Sankar Adhya at the National Institutes of Health, National Cancer Institute and the Center for Cancer Research. We thank Chiara Zurla and Carlo Manzo for discussion and suggestions and William J. Dunn for assistance with image analysis.



**Fig. (6).** Specifically bound protein particles may nucleate adjacent semi-specific binding to secure DNA loops. (a) A small number of DNA loops were secured by two CI particles. Scale bar represents 100 nm. Possible models of CI binding to (b) staggered or (c) directly juxtaposed  $O_L$  and  $O_R$  regions. (d) Labeled segments of looped DNA molecules secured by two CI particles.

**Table 2.** Segment Lengths (nm) for DNA Loops Secured by Two Protein Particles (shown in Fig. 6)

Segment DNA molecule	a	b	c	d	e
<b>Directly juxtaposed operators</b>					
<i>expected</i>	129.0	14.0	125.7	13.0	237.5
1	126.8	20.2	113.7	19.6	233.4
2	127.1	14.1	123.2	16.5	238.4
3	125.2	11.7	113.0	13.6	221.8
4	125.3	20.0	117.9	17.7	222.6
5	124.0	19.5	104.3	20.4	231.2
<i>mean</i>	<b>125.7</b>	<b>17.1</b>	<b>114.4</b>	<b>17.6</b>	<b>229.5</b>
<b>Staggered operators</b>					
<i>expected</i>	116.8	14.7	142.2	14.7	230.0
6	116.1	18.5	123.7	15.9	223.1
7	119.3	16.5	130.9	18.0	231.1

## SUPPLEMENTARY MATERIAL

Supplementary material is available on the publishers Web site along with the published article.

## REFERENCES

- [1] Ptashne, M. (1986) *A Genetic Switch*. Cell Press: Cambridge, MA.
- [2] Ptashne, M. and Gann, A. (2004) *Genes & Signals*. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY.
- [3] Dodd, I.B.; Shearwin, K.E.; Perkins, A.J.; Burr, T.; Hochschild, A. and Egan, J.B. (2004) Cooperativity in long-range gene regulation by the lambda CI repressor. *Genes Dev.*, **18**(3), 344-354.
- [4] Maniatis, T. and Ptashne, M. (1973) Multiple repressor binding at operators in bacteriophage-lambda - (Nuclease protection polynucleotide sizing pyrimidine tracts supercoils *E. Coli*). *Proc. Natl. Acad. Sci. USA*, **70**(5), 1531-1535.
- [5] Oppenheim, A.B.; Kobiler, O.; Stavans, J.; Court, D.L. and Adhya, S. (2005) Switches in bacteriophage lambda development. *Ann. Rev. Gene.*, **39**, 409-429.
- [6] Koblan, K.S. and Ackers, G.K. (1992) Site-specific enthalpic regulation of DNA-transcription at bacteriophage-lambda Or. *Biochemistry*, **31**(1), 57-65.
- [7] Senear, D.F.; Brenowitz, M.; Shea, M.A. and Ackers, G.K. (1986) Energetics of cooperative protein DNA interactions - comparison between quantitative deoxyribonuclease footprint titration and filter binding. *Biochemistry*, **25**(23), 7344-7354.
- [8] Ptashne, M. (2004) *A genetic Switch: Phage Lambda Revisited*, 3rd ed: Cold Spring Harbor Laboratory Press, Cambridge, MA.
- [9] Bell, C.E.; Frescura, P.; Hochschild, A. and Lewis, M. (2000) Crystal structure of the lambda repressor C-terminal domain provides a model for cooperative operator binding. *Cell*, **101**(7), 801-811.
- [10] Jain, D.; Nickels, B.E.; Sun, L.; Hochschild, A. and Darst, S.A. (2004) Structure of a ternary transcription activation complex. *Mol. Cell*, **13**(1), 45-53.
- [11] Meyer, B.J.; Maurer, R. and Ptashne, M. (1980) Gene-regulation at the right operator (Or) of bacteriophage-lambda .2. Or1, Or2, and Or3 - their roles in mediating the effects of repressor and cro. *J. Mole. Biol.*, **139**(2), 163-194.
- [12] Nickels, B.E.; Dove, S.L.; Murakami, K.S.; Darst, S.A. and Hochschild, A. (2002) Protein-protein and protein-DNA interactions of sigma(70) region 4 involved in transcription activation by lambda cl. *J. Mol. Biol.*, **324**(1), 17-34.
- [13] Maurer, R.; Meyer, B.J. and Ptashne, M. (1980) Gene-regulation at the right operator (Or) of bacteriophage-lambda. 1. Or3 and auto-genous negative control by repressor. *J. Mol. Biol.*, **139**(2), 147-161.
- [14] Dodd, I.B.; Perkins, A.J.; Tsemitsidis, D. and Egan, J.B. (2001) Octamerization of lambda CI repressor is needed for effective repression of P-RM and efficient switching from lysogeny. *Genes Dev.*, **15**(22), 3013-3022.
- [15] Dodd, I.B.; Shearwin, K.E. and Egan, J.B. (2005) Revisited gene regulation in bacteriophage lambda. *Curr. Opin. Genet Dev.*, **15**(2), 145-152.
- [16] Zurla, C.; Manzo, C.; Dunlap, D.; Lewis, D.E.; Adhya, S.; Finzi, L. (2009) Direct demonstration and quantification of long-range DNA looping by the {lambda} bacteriophage repressor. *Nucleic Acids Res.*, **37**(9), 2789-2795.
- [17] Sadler, J.R.; Sasmor, H. and Betz, J.L. (1983) A perfectly symmetric lac operator binds the lac repressor very tightly. *Proc. Natl. Acad. Sci. USA*, **80**(22), 6785-6789.
- [18] Simons, A.; Tils, D.; von Wilcken-Bergmann, B. and Muller-Hill, B. (1984) Possible ideal lac operator: *Escherichia coli* lac operator-like sequences from eukaryotic genomes lack the central G X C pair. *Proc. Natl. Acad. Sci. USA*, **81**(6), 1624-1628.
- [19] Meijering, E.; Jacob, M.; Sarria, J.C.; Steiner, P.; Hirling, H. and Unser, M. (2004) Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry A*, **58**(2), 167-176.
- [20] Abramoff, M.D.; Magelhaes, P.J. and Ram, S.J. (2004) Image processing with image. *J. Biophoton. Inter.*, **11**(7), 36-42.
- [21] Claudio Rivetti, S.C. (2001) Accurate length determination of DNA molecules visualized by atomic force microscopy: evidence for a partial B- to A-form transition on mica. *Ultramicroscopy*, **87**, 12.
- [22] Koblan, K.S. and Ackers, G.K. (1992) Site-specific enthalpic regulation of DNA transcription at bacteriophage lambda OR. *Biochemistry*, **31**(1), 57-65.
- [23] Yang, Y.; Sass, L.E.; Du, C.; Hsieh, P. and Erie, D.A. (2005) Determination of protein-DNA binding constants and specificities from statistical analyses of single molecules: MutS-DNA interactions. *Nucleic Acids Res.*, **33**(13), 4322-4334.
- [24] Schneider, S.W.; Larmer, J.; Henderson, R.M. and Oberleithner, H. (1998) Molecular weights of individual proteins correlate with molecular volumes measured by atomic force microscopy. *Pflugers Arch.*, **435**(3), 362-367.
- [25] Neaves, K.J.; Cooper, L.P.; White, J.H.; Carnally, S.M.; Dryden, D.T.; Edwardson, J.M. and Henderson, R.M. (2009) Atomic force microscopy of the EcoKI Type I DNA restriction enzyme bound to DNA shows enzyme dimerization and DNA looping. *Nucleic Acids Res.*, **37**(6), 2053-2063.
- [26] Ratcliff, G.C. and Erie, D.A. (2001) A novel single-molecule study to determine protein-protein association constants. *J. Am. Chem. Soc.*, **123**(24), 5632-5635.
- [27] Hsieh, M. and Brenowitz, M. (1997) Comparison of the DNA association kinetics of the lac repressor tetramer, its dimeric mutant lac iadi and the native dimeric gal repressor. *J. Biol. Chem.*, **272**(35), 22092-22096.
- [28] Burz, D.S.; Beckett, D.; Benson, N. and Ackers, G.K. (1994) Self-assembly of bacteriophage lambda ci repressor: effects of single-site mutations on the monomer-dimer equilibrium. *Biochemistry*, **33**(28), 8399-8405.
- [29] Maniatis, T. and Ptashne, M. (1973) Multiple repressor binding at the operators in bacteriophage lambda. *Proc. Natl. Acad. Sci. USA*, **70**(5), 1531-1535.
- [30] Dodd, I.B.; Shearwin, K.E.; Perkins, A.J.; Burr, T.; Hochschild, A. and Egan, J.B. (2004) Cooperativity in long-range gene regulation by the lambda CI repressor. *Genes Dev.*, **18**(3), 344-354.

## Appendix B: DNA Looping in Prophage Lambda: New Insight from Single-Molecule Microscopy

### Chapter 9

# DNA Looping in Prophage Lambda: New Insight from Single-Molecule Microscopy

Laura Finzi, Carlo Manzo, Chiara Zurla, Haowei Wang,  
Dale Lewis, Sankar Adhya, and David Dunlap

## 9.1 Introduction

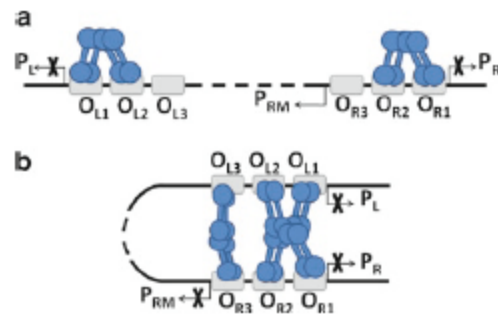
The lambda ( $\lambda$ ) bacteriophage epigenetic switch is a molecular mechanism that permits the quiescent (lysogenic) state of the bacteriophage to irreversibly switch to the virulent (lytic) state. After infection of its host, *E. coli*,  $\lambda$ , a temperate phage, most often grows lysogenically. The phage DNA integrates in the bacterial chromosome and is replicated along with it and transmitted to the bacterial progeny as a prophage. Lysogeny is very stable and yet, the switch to lysis is very efficient. Upon switching to lysis, the viral DNA is excised from the bacterial chromosome and the host machinery is used to produce viral progeny that is then released upon bursting of the host. The pathway to lysis is triggered in response to threats such as starvation, poisoning, or DNA damage.

The lysogenic state of  $\lambda$  prophages is maintained by the  $\lambda$  repressor, or CI protein [35]. During lysogeny, dimers of CI bind to the  $O_L$  and  $O_R$  control regions, located about 2.3 kbp apart on the phage genome (Fig. 9.1a) and repress  $P_L$  and  $P_R$  promoters for the lytic genes. Each control region contains three binding sites for CI,  $O_L1$ ,  $O_L2$ ,  $O_L3$  and  $O_R1$ ,  $O_R2$ ,  $O_R3$  [11, 26, 33]. CI binds to these operators with an intrinsic affinity  $O_L1 > O_R1 > O_L3 > O_L2 > O_R2 > O_R3$  [17, 38]. By studying the  $O_L$  and  $O_R$  regions separately and in isolation from the rest of the  $\lambda$  chromosome [34], it was found that pairs of dimers interact when bound to adjacent or nearby operators, forming tetramers (Fig. 9.1a). These cooperative interactions improve the specificity and strength of CI binding to  $O_R1$  and  $O_R2$ , and  $O_L1$  and  $O_L2$ , respectively, so that the order of binding affinity changes to  $O_R1 \sim O_L1 \sim O_R2 \sim O_L2 > O_L3 > O_R3$ . Biochemical and genetic studies have identified the contacts between amino acids in the C-terminal domain (CTD) that mediate these interactions. These contacts have been confirmed from the crystal structure of the isolated CTD tetramer [7] and are thought to

---

L. Finzi (✉) and D. Dunlap (✉)  
Cell Biology Department, Emory University, 615 Michael St.,  
Atlanta, GA 30322, USA  
e-mail: lfinzi@physics.emory.edu

M.C. Williams and L.J. Maher, III (eds.), *Biophysics of DNA-Protein Interactions*, 193  
Biological and Medical Physics, Biomedical Engineering,  
DOI 10.1007/978-0-387-92808-1\_9, © Springer Science+Business Media, LLC 2011



**Fig. 9.1** Model of CI regulation by long-range DNA looping proposed by Dodd et al. [11]. (a) CI dimers bound cooperatively at  $O_{R1}$  and  $O_{R2}$  repress transcription at  $P_R$  while the dimer at  $O_{R2}$  also activates transcription from  $P_{RM}$ . The dimers bound cooperatively at  $O_{L1}$  and  $O_{L2}$  repress transcription at  $pL$ . (b) Tetramers of CI bound at  $O_L$  and  $O_R$  interact forming an octameric complex and a 2.3 kbp DNA loop. This higher-order complex facilitates cooperative binding of another pair of CI dimers at  $O_{L3}$  and  $O_{R3}$ , resulting in the formation of another CI head-head tetramer and repression of transcription from  $P_{RM}$ .

contribute considerably to the stability of lysogeny. Occupancy of  $O_{R2}$  by CI also activates transcription of the CI gene from the  $P_{RM}$  promoter, giving CI a positive auto-regulatory mechanism [16, 30, 32] (Fig. 9.1a). This increases the level of CI to that required for repression of the lytic genes, as described above. At very high concentrations, CI was observed to also bind to  $O_{R3}$  and repress its own transcription from  $P_{RM}$  [29]. This negative auto-regulation had been suggested to be important to prevent an excessive accumulation of repressor and facilitate efficient switching to lysis when necessary. However, such a negative regulation did not seem possible at physiological concentrations and the role of both  $O_{L3}$  and  $O_{R3}$  remained controversial. In particular, the conventional wisdom was that  $O_{L3}$  was an evolutionary vestige.

A few years ago, it was suggested that, in the context of the intact  $\lambda$  chromosome, CI molecules bind cooperatively not just to adjacent sites, but also to sites separated by thousands of base pairs, inducing a regulatory loop in the viral DNA [11]. Looping is widely used as a gene regulation mechanism [28] but, in most of the prokaryotic cases, the loop length is limited to few hundred base pairs. Therefore, looping is generally viewed as a way to increase the local protein concentration, that is, as a mechanism enhancing the ability of a repressor to bind to a weak operator site through the interaction with a second repressor molecule bound to a stronger operator nearby. Clearly, this is not the case of the  $\lambda$  switch since the large loop size would not produce a significant increase in the local concentration (less than one order of magnitude).

This apparent quandary can be rationalized by the mechanistic hypothesis that the  $\lambda$  loop first forms by the interaction between two tetramers bound at  $O_{R1}-O_{R2}$  and  $O_{L1}-O_{L2}$ , respectively. This octamer-mediated loop brings  $O_{L3}$  and  $O_{R3}$  into juxtaposition and favors their occupancy by CI dimers that can interact “head-to-head” in a cooperative fashion, leading to a DNA loop mediated by a CI octamer plus tetramer. According to this hypothesis, the loop provides the correct scaffold for CI binding to weak  $O_{R3}$  at lysogenic concentrations [9–11] and effective

repression of  $P_{RM}$  (Fig. 9.1b). *In vivo* experiments also indicate that such a loop would increase repression from lytic  $P_R$  fourfold [11, 37], with DNA looping stabilizing the lysogenic state. Thus, stability of the lysogenic state, effective repression of  $P_{RM}$  and efficient switching to lysis in  $\lambda$  may all depend on DNA looping. These considerations piqued the interest of several groups [1–3, 33, 39, 48] led Ptashne to revise the third edition of his book on the  $\lambda$  bacteriophage genetic switch by adding a chapter on the newly proposed CI-mediated long-range interaction and its physiological implications [34].

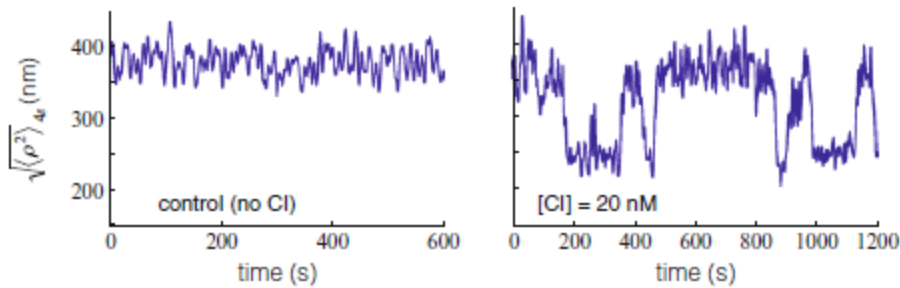
Here, we summarize new findings on  $\lambda$  repressor interaction with  $\lambda$  DNA. We describe how single-molecule experiments in our laboratories have produced direct evidence of CI-mediated  $\lambda$  DNA looping *in vitro* [47], the critical role of interactions between CI dimers bound at the  $O_3$  sites in loop stabilization, and the correlation between CI-mediated looping and activity of  $P_{RM}$  (Sect. 9.2). Furthermore, the complex kinetics of both loop formation and breakdown lead to a postulated kinetic scheme in which nonspecific binding by CI plays a significant role (Sect. 9.3). Strategies for testing this kinetic hypothesis and some preliminary evidence are described at the end of Sect. 9.3, while the conclusions are discussed in Sect. 9.4. The studies provide significant evidence that the multi-operator arrangement found in  $\lambda$  is the minimum necessary to guarantee robust lysogeny (strong repression of lytic genes) and efficient switching to lysis and mechanistic details of how looping confers epigenetic flexibility to cells.

## 9.2 Loop Stability and Correlation with In Vitro Transcription

### 9.2.1 CI Mediated Looping

Single molecule techniques have proven to be powerful tools for dissecting protein-induced conformational changes in DNA, such as looping. In particular, using the tethered particle motion (TPM) technique, direct evidence of loop formation and breakdown by CI was provided [47, 48]. In TPM, a submicron-sized bead is tethered by a single DNA molecule to the glass surface of a microscope flow chamber. The bead's lateral displacement with respect to the tether point  $\rho(t) = \sqrt{x^2(t) + y^2(t)}$  is recorded as function of time. The running average of the squared displacement over a suitable (4 s) time window  $\langle \rho^2 \rangle_{4s}^{1/2}$ , is a measurement of the amplitude of the Brownian motion of the bead. In the presence of looping events, this gives rise to a telegraph-like signal [5, 6, 12, 31] and allows loop detection and quantification (Fig. 9.2, right).

In order to reliably analyze the Brownian motion and interpret amplitude fluctuations in TPM measurements, experimental and theoretical methods have been developed and successfully applied in predicting the TPM signal for a broad range of DNA tether lengths [5, 27, 31].



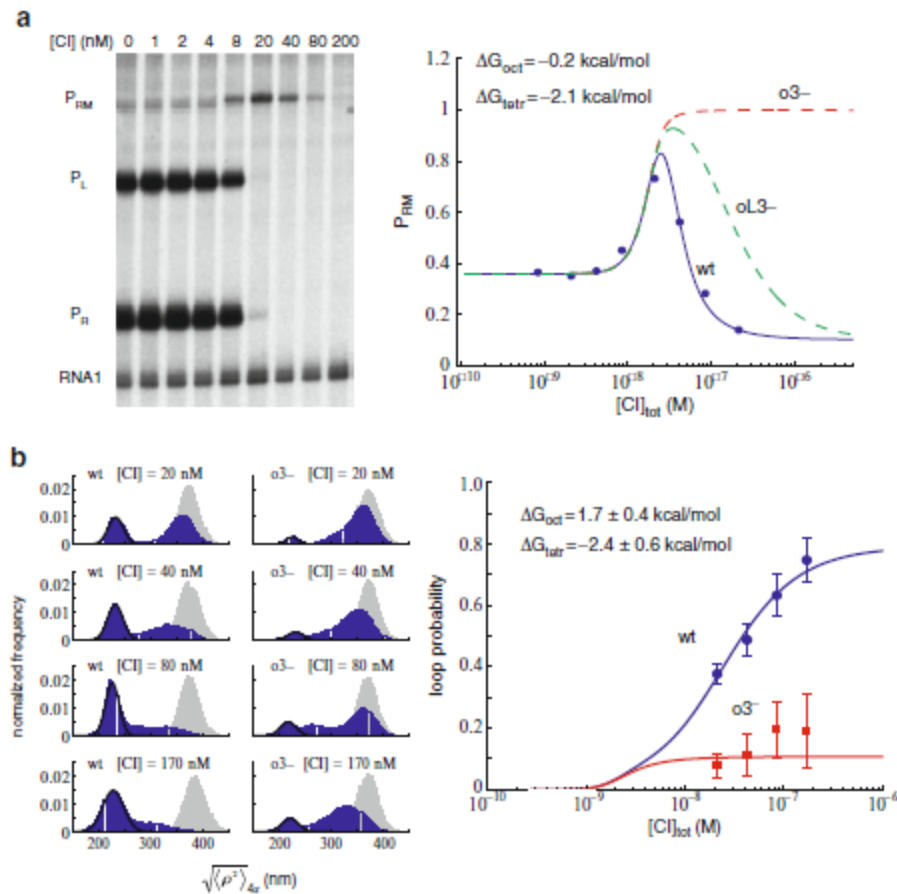
**Fig. 9.2**  $\sqrt{\langle \rho^2 \rangle_{4S}}^{1/2}$  as a function of time for beads tethered by a DNA fragment where the regulatory regions are separated by 2,317 bp. *Left*: In the absence of CI (control experiment); *Right*: In the presence of 20 nM CI (monomer). All measurements are performed in 1 buffer (10 mM Tris-HCl pH 7.4, 200 mM KCl, 5% DMSO, 0.1 mM EDTA, 0.2 mM DTT, and 0.1 mg/ml  $\alpha$ -casein). The length of the DNA tether is 3,477 bp

### 9.2.2 Dependence of Loop Probability/Stability on CI Concentration and Correlation to $P_{RM}$ Activity

It is now evident that strong stability of the lysogenic state, effective repression of  $P_{RM}$ , and efficient switching to lysis in  $\lambda$  all involve DNA looping, which regulates the amount of synthesized CI protein. Therefore, it is important to understand how looping depends on CI concentration, as this will yield insight into the robustness and sensitivity of the  $\lambda$  switch. With this goal, our laboratories have directly detected and thermodynamically characterized CI-mediated dynamic loop formation [48]. Loop formation was previously found to be correlated with transcriptional regulation of  $P_L$ ,  $P_R$ , and  $P_{RM}$  at various CI concentrations in the physiological range, based on the estimated number of CI monomers per lysogen [33]. *In vitro* transcription assays at different CI concentrations (Dale A.E. Lewis, unpublished results; Fig. 9.3a) can be compared with a series of TPM experiments, collectively constituting a “looping titration” (Fig. 9.3b).

For the purpose of this discussion, there are two important features of the transcription results. First, transcription from the lytic promoters  $P_L$  and  $P_R$ , which is high at low CI concentrations, is reduced at 8 nM CI and is completely repressed at 20 nM CI. Second, CI transcription from  $P_{RM}$  is a function of CI concentration, as expected from the model of positive/negative autoregulation described in Sect. 9.1. The assays indicate that transcription from  $P_{RM}$  is at a basal level for concentrations lower than 8 nM, increases up to maximal activation at 20 nM CI and is progressively repressed as the concentration of CI is increased further (Fig. 9.3a).

According to the model for  $P_{RM}$  autoregulation, repression should result from  $O_R3$  occupation, which in turn results from loop formation and  $O_L3$  occupation. Therefore, the probability of loop formation should be an increasing function of CI concentration (Fig. 9.3a, left panel). In order to test this idea, TPM measurements



**Fig. 9.3** (a)  $P_{RM}$  transcription concentration dependence. *Left*, *in vitro* transcription assays showing the amount of transcripts from  $P_{RM}$  and the lytic promoters  $P_L$  and  $P_R$ . *Right*, comparison of the trends of  $P_{RM}$  activity. Experimental data for wt  $\lambda$  DNA (dots). Solid line is best fit for wt  $\lambda$  and dashed lines are simulated repression curves for  $O3^-$  and  $O_L3^-$  DNA. All data are normalized to the maximum repression value obtained for  $O3^-$  DNA. (b) Dependence of DNA looping on CI concentration. *Left*, probability distribution of the TPM signal observed for tethered beads (about 40 for each condition) in the presence of different CI concentrations. The *left column* refers to wt DNA, while the *right column* refers to  $O3^-$  DNA. The *gray peaks* represent the control measurements without CI. *Right*, loop probability as a function of CI concentration for wt  $\lambda$  (dots) and  $O3^-$  DNA (squares). The *lines* are the result of the best fit performed as explained in [48]

on several hundred DNA-tethered beads were performed. Cumulative histograms representing the data from all these measurements at CI concentrations of 0, 20, 40, 80, and 170 nM are reported in the left panel of Fig. 9.3b. This figure shows that in the absence of CI, DNA is in the unlooped state only; just one peak appears in the frequency histogram (gray). These control measurements also show the high reproducibility of the TPM experiments. In the presence of CI, the histograms



show two peaks: one corresponding to the DNA unlooped state (higher amplitudes of Brownian motion) and the other to the looped conformation (lower amplitudes of Brownian motion). Note that the looped state, although not predominant, does occur at low CI concentrations. At 40 nM CI concentration, DNA molecules spend half of the time in the looped configuration and this correlates well with the partial repression of  $P_{RM}$  observed in Fig. 9.3a. In the presence of 80 nM CI, the DNA is prevalently looped, as expected from *in vitro* transcription assays. However, even at 170 nM CI, the DNA is not always looped. This indicates that the dynamic nature of the system is modulated by CI concentration, but never completely disappears at physiological concentrations. Interestingly, the probability of loop formation as derived from the histograms (Fig. 9.3b, right panel) was described very well by an extension of the statistical thermodynamic model developed by Ackers et al. [17, 38]. In order to relate the measured loop probabilities to the microscopic configurations of CI bound to the six operator sites, 81 unlooped and 32 looped configurations were considered according to the scheme proposed by Anderson and Yang [1, 2]. The probability for each DNA-protein configuration was expressed as,

$$f_i = \frac{[CI_2]^{s_i} \exp(-\Delta G_i / RT)}{\sum_i [CI_2]^{s_i} \exp(-\Delta G_i / RT)}, \quad (9.1)$$

where  $\Delta G_i$  is the sum of the free energies for binding, short range cooperativity, and looping of each configuration and  $s_i$  is the number of bound CI dimers ( $CI_2$ ; Fig. 9.4). CI dimer concentration was calculated from the expression for the total concentration of CI:

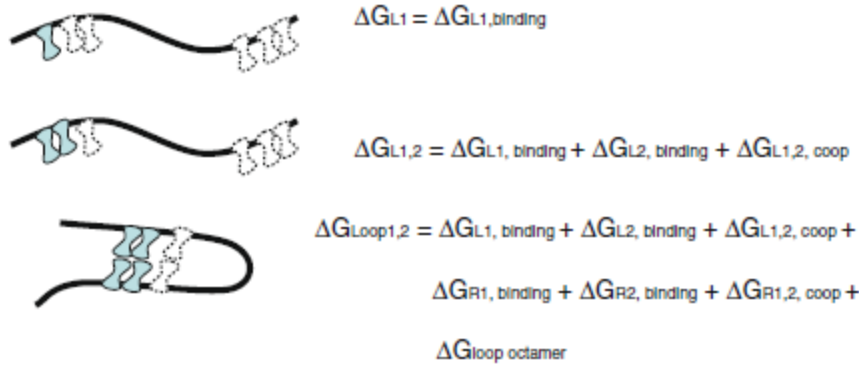
$$[CI]_{tot} = \sqrt{\frac{[CI_2]}{K_d}} + 2[CI_2] + 2K_{NS}l[DNA][CI_2] + [DNA] \sum_i s_i \cdot f_i, \quad (9.2)$$

where  $K_d$  is the dimerization constant for CI [4],  $K_{NS}$  is the nonspecific binding constant [4], and  $l$  is the DNA length in base pairs (see Table 1 in [48]). The terms of this equation from left to right represent: CI monomers, CI dimers, nonspecifically bound, and specifically bound CI. The concentration-dependent loop probability was then calculated as:

$$\text{Loop probability} = \frac{\sum_{i=82}^{113} f_i}{\sum_{i=1}^{113} f_i} \quad (9.3)$$

in which the sum in the numerator runs over all the 32 looped configurations as in [1, 2] and is normalized by the sum of all the possible (looped and unlooped) configurations.

This thermodynamic analysis assumes that the loop can be secured either by a CI octamer or by an octamer plus a tetramer. The difference in free energy between these two looped species represents the tetramerization free energy  $\Delta G_{tet}$ .



**Fig. 9.4** Illustration of the procedure used to calculate  $\Delta G_i$  in a few representative cases. The free energy for each unlooped species was expressed as the sum of all the free energies for binding and short-range cooperativity, available from previous work [17, 38]. The free energy expression for the looped species also included the term  $\Delta G_{\text{oct}}$  and the tetramerization term,  $\Delta G_{\text{tet}}$  was added only for configurations in which two CI dimers not involved in the octamer were juxtaposed [2]. Using this model, the probability of looping was expressed as a function of CI concentration. The experimental data obtained with both wt  $\lambda$  and  $\lambda O_3^-$  DNA were fitted simultaneously to estimate  $\Delta G_{\text{oct}}$  and  $\Delta G_{\text{tet}}$ .

To quantitatively correlate the single molecule experiment with the transcription assay, the same statistical mechanical model described above and used in [48] was employed to test whether it could also describe *in vitro* transcription from  $P_{RM}$ . Measurements of mRNA levels were made from the digitized image of the gel shown in the left panel of Fig. 9.3a and are shown in the right panel of Fig. 9.3a. Since the  $P_{RM}$  promoter is activated by occupancy of the  $O_R2$  operator and it is repressed by the binding of a CI dimer to  $O_R3$  [29], the expression for transcription from  $P_{RM}$  was as:

$$P_{RM} = P_{\text{repr}} \sum_{i \in O_R^3} f_i + P_{\text{act}} \sum_{\substack{i \in O_R^2 \\ i \text{ unlooped}}} f_i + \alpha P_{\text{act}} \sum_{\substack{i \in O_R^2 \\ i \text{ looped}}} f_i + P_{\text{bas}} \sum_{\text{else}} f_i, \quad (9.4)$$

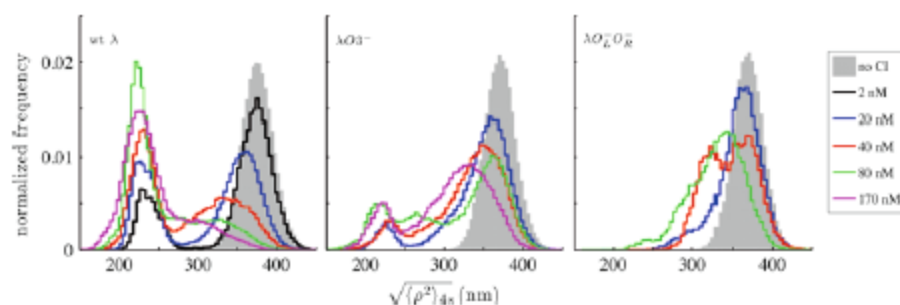
in which a basal transcription value was assigned to all the configurations except those having  $O_R2$  occupied (activated transcription) and those having  $O_R3$  occupied (repressed transcription). Although the effect of looping on the repression of transcription is still a matter of debate [1, 2, 11, 33], we assumed a two- to threefold ( $\alpha=2-3$ ) increase for the activated transcription in the looped state [1, 2]. In the fitting of this data, the activated transcription  $p_{\text{act}}$ ,  $\Delta G_{\text{oct}}$ , and  $\Delta G_{\text{tet}}$  values were left as free parameters. The remarkable fit demonstrates that the model also accurately describes how transcription depends on CI concentration.

It must be noted that the *in vitro* transcription experiments were carried out on plasmids having a loop length of only  $\sim 400$  bp, whereas in the TPM experiment the wt loop region ( $\sim 2,000$  bp) was used. Since the free energy of loop formation also depends on the physical properties of DNA (loop length, supercoiling state, and persistence length), a direct comparison of the looping free energy  $\Delta G_{\text{oct}}$  obtained in the

two cases is not possible. However, the lower value found for  $\Delta G_{\text{oct}}$  in the transcription assay presumably reflects the reduced cost of looping due to plasmid supercoiling and shorter loop size. On the other hand, since tetramer formation takes place after loop closure, the value of  $\Delta G_{\text{tet}}$  is not influenced by the loop length or the degree of supercoiling. Consequently, the values of  $\Delta G_{\text{tet}}$  determined in fitting the transcription assay ( $-2.1$  kcal/mol) and the TPM-derived  $\Delta G_{\text{tet}}$  ( $-2.4$  kcal/mol) [48] are very similar.

### 9.2.3 Role of the $O3$ Operators

If (1) the four sites  $O_L1$ ,  $O_L2$  and  $O_R1$ ,  $O_R2$  are occupied first and almost simultaneously because of strong and cooperative binding of CI, and (2) CI octamer-mediated loop formation is further secured by the “head-to-head” interaction of two dimers bound at the  $O_L3$  and  $O_R3$  operators, then mutation of the  $O3$  operators should interfere with stable CI-mediated looping. This idea was tested with TPM experiments using DNA tethers containing mutated  $O3$  sites ( $O3^-$  DNA) at four different CI concentrations (Fig. 9.3b). Details regarding these measurements are reported in [48]. First, it was verified that no loop formation was observed in  $O_L^-O_R^-$  DNA, where all six  $\lambda$  operators had been mutated (Fig. 9.5, third panel). This indicated that the selected point mutations effectively abrogate specific CI binding to operators. Then the probability of loop formation in  $O3^-$  DNA was found to be approximately 10% for all CI concentrations tested (Fig. 9.3b, middle panel). These results show that intact  $O3$  operators dramatically shift the equilibrium toward looping. According to the thermodynamic model, concentration-independent  $O3^-$  looping results from cooperative filling of all the remaining sites, even at low CI concentration [30]. Instead, in the wt DNA, concentration-driven occupancy of the  $O3$  sites drives the formation of octamer-plus-tetramer-mediated loops with lower free energy than CI octamer-mediated loops.

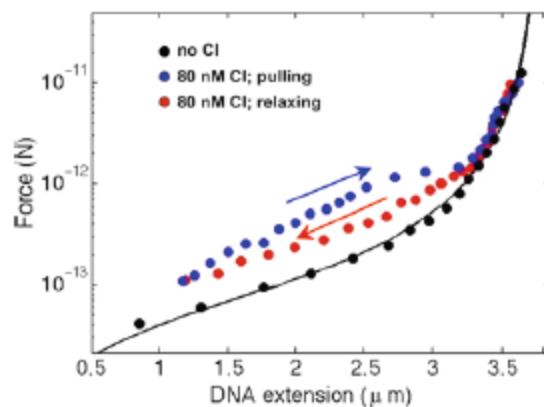


**Fig. 9.5** Histograms of the TPM signal measured for wt,  $O3^-$  and null DNA at various nanomolar concentrations of CI reveal that (left) looping increases with CI concentration for wild-type DNA, (middle) looping dramatically decreases when  $O3$  operators are mutated, and (right) simultaneous mutations to all operators abrogates looping

### 9.2.4 CI Nonspecific Binding

The peak corresponding to the extended, unlooped state in the frequency distribution histograms of the TPM signal shifts toward smaller values of  $\langle \rho^2 \rangle_{4S}^{1/2}$ , as CI concentration increases (Figs. 9.3b and 9.5). This could have two different causes: poor temporal resolution preventing resolution of fast loop breakdown events and/or progressive DNA compaction due to bending associated with nonspecific binding of CI to the double helix. Typically, the system is sampled at 50 Hz, but filtering reduces the resolution to 4 s. Even using statistical methods that do not employ filtering (see below), shortening of the unlooped DNA occurs. Thus, the effect of increasing CI concentrations was compared on null (*OL<sup>-</sup>OR<sup>-</sup>*), wt and *O3<sup>-</sup>* DNA (Fig. 9.5). The peak corresponding to unlooped DNA shifted leftward in all cases, suggesting that nonspecific CI binding is significant and considerably shortens the DNA tether. Work from the Cox lab lends strong support to the idea that nonspecific CI binding may be significant [44].

Further evidence of nonspecific, CI-induced DNA bending was obtained stretching and relaxing single DNA molecules by means of magnetic tweezers. The *force* versus *extension* curves that were obtained with or without CI were markedly different (Fig. 9.6). In order to produce equivalent DNA extension, more force was necessary in the presence of CI, indicating that CI may induce bends or kinks in the DNA through binding or transient interactions with nearby dimers that bend the intervening DNA. The noteworthy hysteresis between the relaxation and stretching cycles may reveal interesting mechanistic details and is under investigation. Furthermore, quantification of the number of nonspecifically bound proteins is possible using the recent model proposed by Zhang and Marko [46], Liebesny et al. [50].



**Fig. 9.6** DNA extension versus force for a wt, 11 kbp-long DNA molecule. *Dots* are averages of points from several pulling/relaxation cycles, in the absence of protein (*black*) and in the presence of 80 nM CI (*blue* and *red*). The *solid curve* is a worm-like-chain fit obtained assuming a persistence length for DNA of 52 nm. The *arrows* indicate that the experimental data were collected during stretching (*blue*) or relaxation of the DNA molecule (*red*)

Considering that CI mRNA is transcribed and translated close to  $P_{RM}$ , which likely produces a local concentration of CI dimers in excess with respect to the number of operators [33, 34], nonspecific binding may have physiological relevance, just as shown in the case of other proteins [14, 49].

### 9.3 Kinetics of CI-Mediated DNA Loop Formation and Breakdown

Kinetic analysis of formation and breakdown of the CI-mediated loop yields insight into the mechanism of looping and its regulatory effect on transcription. The kinetics of protein-induced loop formation and breakdown have been deduced from TPM data for a variety of simple systems [12, 40, 41]. In these studies, the TPM signal was analyzed using time filtering (averaging) of the raw data followed by thresholding to determine the state of the system. Then, the measured dwell times of each state were plotted as probability density functions and the average lifetime of the looped or unlooped state was determined. For such systems, characterized by a simple kinetics, a single (or at most a double) exponential function can fit the data satisfactorily. In this way, the rate constants for the looping and unlooping reactions have been obtained along with an estimate for the free energy of loop formation. Note, however, that in all these cases, time filtering of the raw data significantly impacts the time resolution of the measurements and the kinetic constants. Methods have been proposed to either correct for such a drawback [8, 40, 41] or determine the kinetic constants from the raw data [8, 36]. These approaches, however, require the knowledge of the kinetic mechanism of the reaction being considered, and their application is limited to cases in which: (a) there is only a small number of discrete states separated by fixed energy barriers and (b) the kinetic rate constants connecting these states are independent [21].

An ideal method of data analysis should have the highest possible time resolution and should be independent of physical models. In addition, it should avoid user-adjustable parameters, such as filtering, that skew the raw data.

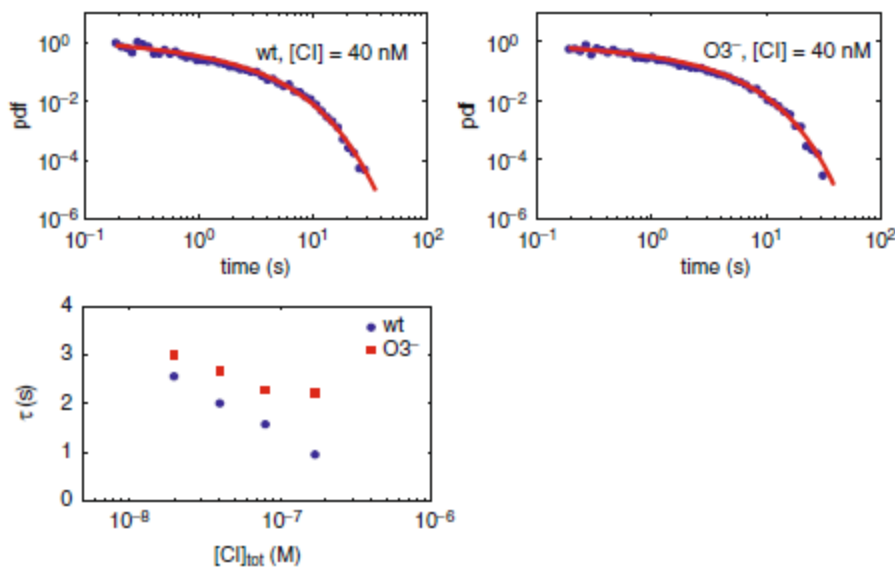
Therefore, an alternative approach to the analysis of TPM traces was devised by Manzo and Finzi [27] based on a method previously published for time traces exhibiting discrete jumps in intensity [45]. A generalized likelihood ratio test is first applied to determine the location of a TPM signal change point (cp). This test is applied recursively to an entire TPM trace, to identify all cps (transitions between different DNA configurations). Expectation–maximization (EM) clustering and the Bayesian information criterion are then used for accurate determination of the number of states accessible to the system. This procedure (cp-EM) allows objective and quantitative determination of TPM cps without the artificial time resolution limitations that arise from filtering and thresholding [45]. The applicability and performance of this analysis was tested on artificial TPM data assembled ad hoc from segments of TPM data acquired in the absence of CI (unlooped) interspersed between segments of TPM data from a DNA molecule of overall length comparable to that expected for the looped lambda DNA. Subsequently, the cp-EM algorithm

was used to analyze TPM data for a lambda DNA molecule in the presence of CI. This analysis confirmed that CI interaction with its operators produces most likely two states, which were commensurate with the looped and unlooped DNA states. The cp-EM approach allowed determination of the looped and unlooped dwell time distributions with a much increased time resolution [27].

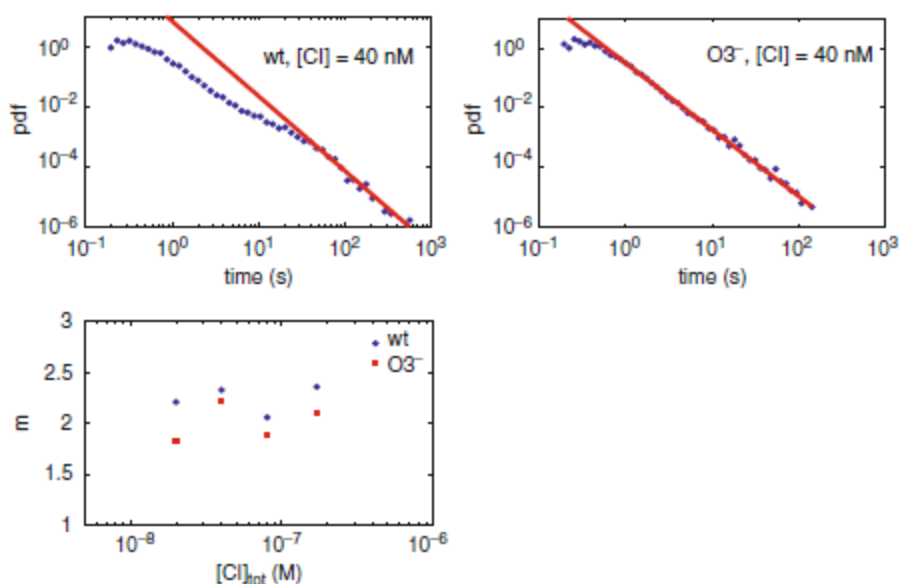
### 9.3.1 Analysis of the $\lambda$ Loop Kinetics

The cp-EM method described above and in [27] was applied to the analysis of the kinetics of CI-mediated loop formation and breakdown. Interestingly, the probability distribution function (pdf) for the dwell times of the looped and unlooped DNA conformations (states) span several orders of magnitude and show nonexponential tails at long times (Figs. 9.7 and 9.8). In the case of  $\lambda$  DNA, it is reasonable to assume that the tripartite operator organization might lead to complex dynamics for loop formation and breakdown. Nevertheless, the discrete and Markovian nature of the kinetic system still predicts an exponential behavior for the pdfs.

We recall here that nonexponential decays have been reported for other physical systems such as ion channel currents [20, 22–25], quantum dot blinking time [13] and for fluorescence emission by green fluorescent protein [18, 19]. Notably, the dwell time determination by means of the filtering-threshold approach produced qualitatively similar results, excluding the possibility that artifacts might have been



**Fig. 9.7** Kinetics of loop formation. *Top*: Probability distribution function obtained for the dwell times of the unlooped state of wt DNA (*left*) and O3<sup>-</sup> (*right*) in the presence of 40 nM CI. Lines are the result of fitting by a stretched exponential pdf. *Bottom*: Average time spent in the unlooped configuration. Wild type and O3<sup>-</sup> data points are blue and red, respectively



**Fig. 9.8** Kinetics of loop breakdown. *Top*: Probability distribution function obtained for the dwell times of the looped state of wt DNA (*left*) and  $O3^-$  (*right*) in the presence of 40 nM CI. *Lines* show the power law behavior at large times. *Bottom*: Power law exponent  $m$  determined from fitting the longer lifetimes. Wild type and  $O3^-$  data points are *blue* and *red*, respectively

introduced by the cp method. Also the fact that, even a single DNA molecule observed for a long time, exhibited looped and unlooped dwell times spanning several orders of magnitude (Fig. 9.9) ruled out the possibility that complex kinetics may arise from heterogeneous sample preparations.

To empirically describe the data, we tested several pdfs. We found that a stretched exponential form provided a satisfactory fitting for the probability distribution of the unlooped state dwell times at any tested CI concentration.

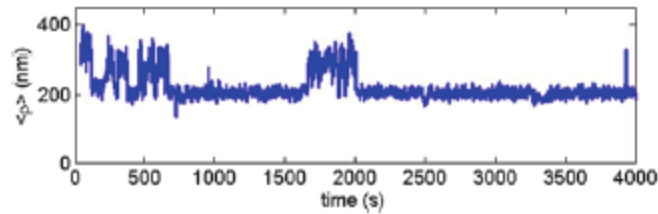
The stretched exponential distribution is expressed as:

$$\text{pdf}_{\text{st}} = c \frac{t^{c-1}}{t_0^c} \exp \left\{ - \left( \frac{t}{t_0} \right)^c \right\}, \quad (9.5)$$

where the two adjustable parameter are the scale parameter  $t_0$  and the exponent  $c$ , the latter being smaller than one and giving an exponential distribution in the limit  $c=1$ . From these parameters, the mean time is obtained as:

$$\tau = \frac{t_0}{c} \Gamma \left( \frac{1}{c} \right) \quad (9.6)$$

where  $\Gamma(x)$  represents the gamma function. The upper panels of Fig. 9.7 show the fit of the data relative to both wt and  $O3^-$  DNA in the presence of 40 nM CI using the



**Fig. 9.9** A TPM trace for a *wt* DNA molecule in the presence of 40 nM CI. The DNA is 3,477 bp long

stretched exponential distribution function. Both fitting parameters  $t_0$  and  $c$  show dependence on CI concentration and are significantly sensitive to the type of DNA (data not shown). As a consequence, their mean unlooped times,  $\tau$ , are similar, as shown in the lower panel of Fig. 9.7. For both *wt* $\lambda$  and *O3<sup>-</sup>* DNA,  $\tau$  decreases with CI concentration, although it does so more rapidly for the *wt* $\lambda$  than for the *O3<sup>-</sup>* DNA.

On the other hand, the dwell time distribution for the looped DNA state could not be fitted with a standard pdf; we observe, however, a power law decay at long times, resulting in a straight line on a log–log plot. The power law pdf:

$$\text{pdf}_{\text{pl}} \propto t^{-m} \quad (9.7)$$

is sometimes referred as a “fractal” distribution since it reveals no characteristic size of the system, and therefore makes it impossible to extract a mean lifetime from the data. The log–log plots of the looped time distribution for *wt* $\lambda$  and *O3<sup>-</sup>* DNA at 40 nM CI (Fig. 9.8, upper panels) show that, while the data from *O3<sup>-</sup>* DNA are well described by a power law function even at very short times, those from *wt* DNA are not. In both cases, however, the data for longer lifetimes are characterized by the exponent  $m \approx 2$  in all the experimental conditions tested (lower panel of Fig. 9.8).

### 9.3.2 Suggested Mechanism of Loop Formation and Breakdown and Role of Nonspecific Binding

The single-molecule experiments described in Sects. 9.2 and 9.3.1 highlight four important features of the  $\lambda$  regulatory loop: (1) the pivotal role of the *O3* sites in the thermodynamics of loop formation, (2) the concentration dependence of both loop formation and breakdown, (3) the presence of significant nonspecific CI binding even at low protein concentrations, which seems to bend or soften DNA and, finally, (4) the nonexponential kinetic behavior of both loop formation and breakdown. Taken together, these observations lead us to formulate the kinetic mechanism described below.

First, consider loop formation. The most direct information on this process is provided by the distribution of the lifetimes of the unlooped state, which indicates the

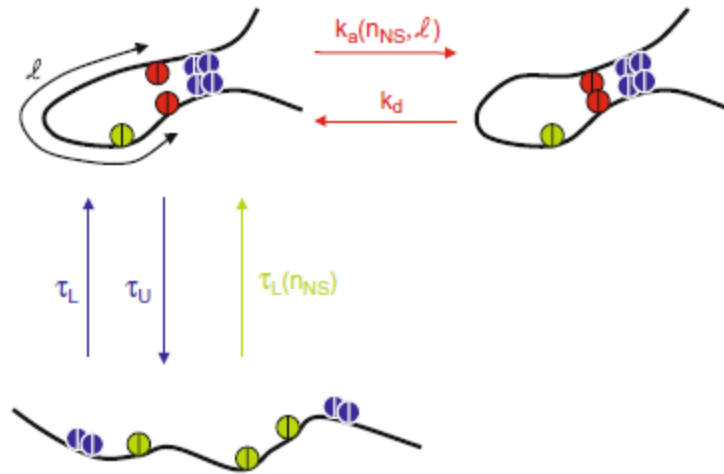


average time for loop formation. As mentioned previously, this distribution is described by a stretched exponential function and the average unlooped state lifetime varies with CI concentration and is affected by mutations of the *O3* sites (Fig. 9.7). According to the thermodynamic hypothesis that a CI octamer is the minimum requirement for loop formation [2, 11, 48], which is then stabilized by an additional CI tetramer, the concentration dependence can simply be explained by the increase in the population of the loop-forming substates (i.e., those having at least two pairs of adjacent, specifically bound CI dimers) as the amount of CI increases.

Kinetic complexity may arise from nonspecific binding of CI to DNA. Indeed, our TPM and force spectroscopy measurements show ample evidence of nonspecific CI binding (see Sect. 9.2.4), and there is other evidence from the Cox group [44]. Nonspecifically bound CI dimers may shorten the DNA by bending or softening the double helix upon binding or through interaction with nearby dimers. The ensuing DNA bending could facilitate loop formation by reducing the elastic energy necessary for loop closure which, in turn, increases the encounter probability among the proteins bound at these two regions. The relevance of these effects would depend on the number of nonspecifically bound proteins in the loop region,  $n_{NS}$ . The variation of the number of nonspecifically bound dimers would then generate a distribution of rate constants for loop formation (schematically represented by  $k_L(n_{NS})$  in Fig. 9.10) that could explain the observed stretched-exponential pdf. Note that nonspecific CI binding is likely to be relevant *in vivo* as well, since CI is produced in excess with respect to the number of operators [33].

In the case of loop breakdown kinetics, the distribution of the looped lifetimes also depends on CI concentration and on the presence of the *O3* sites, but cannot be described by a standard pdf. Moreover, the distribution of long dwell times shows a power law decay with an exponent which does not depend on the experimental conditions. According to the thermodynamic model described above [2, 48], the many possible looped substates can be grouped in two classes mediated by either four or six CI dimers, respectively. Each class will have a different breakdown rate. Consequently and in qualitative agreement with the experiments, the model predicts concentration-dependent loop breakdown kinetics and a simpler distribution in the case of *O3*- DNA in which octamer+tetramer loop cannot form. However, as for the distribution of unlooped lifetimes, the observed pdfs do not exhibit simple exponential behavior.

Nonspecific CI binding may contribute to the observed decay at long times. Indeed, nonspecifically bound CI dimers might interact through their CTD residues [7, 15] just as specifically bound dimers do. In particular, nonspecifically bound dimers within the loop may tetramerize [7, 15] to fortify the specific nucleoprotein complex that secures the loop. In Fig. 9.10,  $k_L(n_{NS})$  and  $k_U$  represent the rate constants of loop formation and breakdown. If no additional nonspecifically bound proteins are present,  $k_L(n_{NS} = 0)$  is single valued and the kinetics of the system are exponential. In the presence of nonspecifically bound CI, the variation of  $n_{NS}$  broadens the distribution of  $k_L(n_{NS})$ . Any further tetramerization between these nonspecifically bound dimers would be dependent on their number  $n_{NS}$  and on their relative separation,  $\ell$ , as schematically diagrammed in Fig. 9.10, where  $k_a(n_{NS}, \ell)$  represents a distribution of rate



**Fig. 9.10** A hypothetical kinetic scheme for loop formation and breakdown. In addition to the CI dimers at the operator sites (*blue*), nonspecifically bound dimers may affect the rate of loop formation (*green*) by DNA bending and may strengthen the loop providing additional loop closure elements (*red*). For simplicity only the case of an octameric loop is sketched

constants for the association between additional, loop-stabilizing CI tetramers (red), and  $k_d$  is the rate constant for their dissociation. These manifold looped states would give rise to a continuous distribution of waiting times for the breakdown of the  $\lambda$ -mediated loop, producing a power law-like decay.

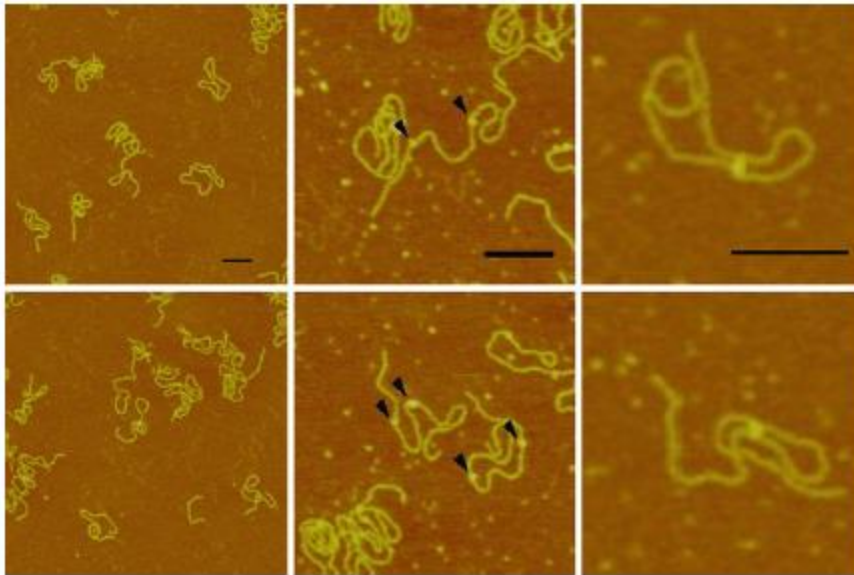
The model for loop formation and breakdown hypothesized here is based on two main ideas: the critical role of the  $O3$  sites and nonspecific CI binding. The latter might facilitate loop formation in this and other physiologically relevant protein-induced loops by decreasing the distance between the interacting sites, maintaining proximity of the specific regions, and by adding stabilizing protein–protein interactions. Indeed, CI has been estimated to be in excess with respect to the number of specific binding sites [33] and extra CI is thought to confer immunity against subsequent infection by additional phages [34].

This proposed mechanism of loop formation and breakdown by the CI repressor seems to be the simplest possible explanation that is consistent with and reconciles: (a) known biochemical data on CI/operator affinity and side-by-side cooperativity [17, 38], (b) *in vivo* transcription data [1], and (c) single-molecule data. This mechanism is also attractive because its key assumptions may easily be tested and the experiments should yield useful guidelines for similar loop-based, regulatory looping systems. Furthermore, we would like to draw the attention of the reader to the fact that, as pointed out by Vilar and Saiz [42], only a narrow range of looping free energies separates high-sensitivity activation and repression of the  $P_{RM}$  promoter. In this sense, nonspecific binding might function as a concentration-sensitive mechanism for the fine tuning of the looping free energy.

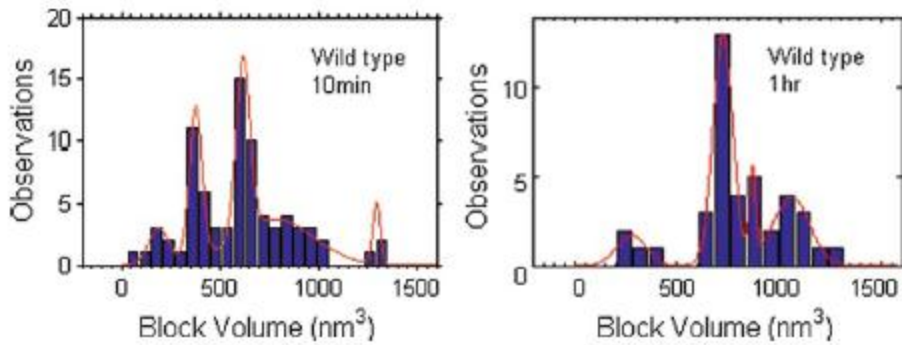
### 9.3.3 Evidence in Support of the Kinetic Scheme

The kinetic model outlined above may be tested in at least two ways: First, a theoretical formulation of the distribution of rate constants produced by nonspecific binding could be derived and compared to the experimental pdfs. Second, the various possible looped and unlooped species and their dependence on CI concentration could be characterized using atomic force microscopy. This single-molecule technique can directly reveal the number of nonspecifically bound CI dimers and their interactions, and some evidence has already been obtained. Figure 9.11 shows AFM images of  $\lambda$  DNA without CI,  $\lambda$  DNA bound specifically by CI, and CI-mediated loops in  $\lambda$  DNA. Volume analysis of the CI particle at the loop closure in several such AFM images revealed that the size of the protein particle increased over time (Fig. 9.12) [43]. Together, these data suggest that as time passes additional proteins bind to stabilize loops.

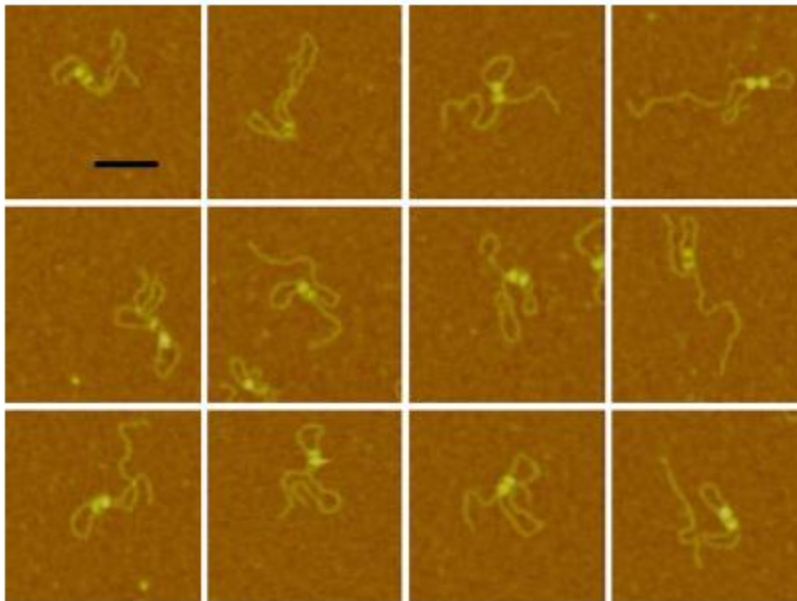
Furthermore, AFM images of DNA molecules in the presence of 40 nM CI included a few DNA loops secured by adjacent protein complexes (Fig. 9.13). These species, reported also in [43], might represent those with additional nonspecific loop closures hypothesized in the kinetic model proposed in Sect. 9.3.2. More experimentation is necessary to confirm these hypotheses. Luckily, experimental and analytical methods are now available to interrogate simple epigenetic switches, such as lambda, and characterize the molecular mechanisms involved.



**Fig. 9.11** *Left column: DNA; center column: CI bound at operators on unlooped DNA; right column: CI-mediated DNA looping. Scale bars represent 100 nm*



**Fig. 9.12** Histograms of particle volumes measured at the loop junction after incubating DNA with 20 nM CI for 10 min (*left*) or 1 h (*right*)



**Fig. 9.13** AFM images in which the CI-mediated DNA loop is secured by two distinct protein particles. Scale bar: 100 nm

## 9.4 Conclusions

From viruses to humans, transcription is regulated by proteins that topologically constrain DNA. In most cases genes are controlled by large, cooperative assemblies of proteins that wrap and loop the DNA. These protein-induced conformational changes in the DNA often constitute real “epigenetic switches” whereby shifting the equilibrium towards one configuration or the other transcriptionally commits the system to one developmental pathway or another. Such is the case of the  $\lambda$  bacteriophage and,

perhaps, of most temperate bacteriophages that may proliferate either quiescently (lysogenic mode) or virulently (lytic mode). Most often following infection, the repressor protein, CI, binds to multipartite operators and mediates cooperative, long-range interactions that repress the lytic genes and maintain a stable lysogenic state, until adverse conditions (DNA damage, toxicity, starvation, etc.) induce a cascade of events that leads to the dissociation of CI from the double helix and efficient switching to lysis. This switch that alternatively stabilizes lysogeny or commits the  $\lambda$  bacteriophage to lytic reproduction [16] is a convenient, experimentally tractable epigenetic paradigm of transcriptional regulation. Due to the importance of  $\lambda$  as a model system not only for transcriptional regulation, but also for genetic networks, the underpinnings of this regulatory loop are important for understanding the sensitivity, stability and operation of such networks in epigenetics.

In this chapter, we have discussed new approaches to reveal details of the molecular mechanism of  $\lambda$  repressor-mediated looping. Single molecule observations have, first of all, provided direct evidence of CI-induced looping between the wild-type lambda operators. In addition, a titration of loop closure probability versus CI concentration combined with a thermodynamic model has substantially confirmed the hypothetical octamer + tetramer mechanism of CI-mediated regulation of lysogeny. Finally, kinetic analyses of the looping dynamics suggest an unsuspected, but in retrospect, logical role that non-specific binding of the CI protein likely plays in lysogenic regulation. In a short time single molecule experimentation has provided detailed insight into how long-range interactions may govern epigenetic switching and these mechanisms will be pertinent to a variety of other systems with multipartite operators and multi- or heteromeric protein assemblies [30].

**Acknowledgments** We would like to thank previous and current members of our groups whose research has facilitated these studies. We are also grateful to Haw Yang, who has provided reagents, analytical tools, and advice. The work described in this chapter was supported by the Italian Funding of Basic Research to LF and DDD, by the HFSP(RGP0050/2002-C) to L.F. and S.A., by the Intramural Research Program of the National Institutes of Health, National Cancer Institute and the Center for Cancer Research to S.A., by the Emory University Research Council and the NIH (RGM084070A) to LF.

## References

1. Anderson L, Yang H (2008) A simplified model for lysogenic regulation through DNA looping. *Conf Proc IEEE Eng Med Biol Soc* 2008:607–610
2. Anderson LM, Yang H (2008) DNA looping can enhance lysogenic CI transcription in phage lambda. *Proc Natl Acad Sci U S A* 105(15):5827–5832
3. Atsumi S, Little JW (2006) Role of the lytic repressor in prophage induction of phage lambda as analyzed by a module-replacement approach. *Proc Natl Acad Sci USA* 103(12):4558–4563
4. Bakk A, Metzler R (2004) Nonspecific binding of the O-R repressors CI and Cro of bacteriophage lambda. *J Theor Biol* 231(4):525–533
5. Beausang JF, Zurla C, Finzi L, Sullivan L, Nelson PC (2007) Elementary simulation of tethered Brownian motion. *Am J Phys* 75:520–523

6. Beausang JF, Zurla C, Manzo C, Dunlap D, Finzi L, Nelson PC (2007) DNA looping kinetics analyzed using diffusive hidden Markov model. *Biophys J* 92(8):L64–L66
7. Bell CE, Frescura P, Hochschild A, Lewis M (2000) Crystal structure of the lambda repressor C-terminal domain provides a model for cooperative operator binding. *Cell* 101(7):801–811
8. Colquhoun D, Sigworth FJ (1983) Fitting and statistical analysis of single channel recording. Plenum, New York
9. Dodd IB, Perkins AJ, Tsemitsidis D, Egan JB (2001) Octamerization of lambda CI repressor is needed for effective repression of P-RM and efficient switching from lysogeny. *Genes Dev* 15(22):3013–3022
10. Dodd IB, Shearwin KE, Egan JB (2005) Revisited gene regulation in bacteriophage lambda. *Curr Opin Genet Dev* 15(2):145–152
11. Dodd IB, Shearwin KE, Perkins AJ, Burr T, Hochschild A, Egan JB (2004) Cooperativity in long-range gene regulation by the lambda CI repressor. *Genes Dev* 18(3):344–354
12. Finzi L, Gelles J (1995) Measurement of lactose repressor-mediated loop formation and breakdown in single DNA-molecules. *Science* 267(5196):378–380
13. Frantsuzov P, Kuno M, Janko B, Marcus RA (2008) Universal emission intermittency in quantum dots, nanorods and nanowires. *Nat Phys* 4(5):519–522 [10.1038/nphys1001]
14. Guerra RF, Imperadori L, Mantovani R, Dunlap DD, Finzi L (2007) DNA compaction by the nuclear factor-Y. *Biophys J* 93(1):176–182
15. Hochschild A, Ptashne M (1988) Interaction at a distance between lambda-repressors disrupts gene activation. *Nature* 336(6197):353–357
16. Jain D, Nickels BE, Sun L, Hochschild A, Darst SA (2004) Structure of a ternary transcription activation complex. *Mol Cell* 13(1):45–53
17. Koblan KS, Ackers GK (1992) Site-specific enthalpic regulation of DNA-transcription at bacteriophage-lambda OR. *Biochemistry* 31(1):57–65
18. Leiderman P, Huppert D, Agmon N (2006) Transition in the temperature-dependence of gfp fluorescence: from proton wires to proton exit. *Biophys J* 90(3):1009–1018. doi:10.1529/biophysj.105.069393
19. Leiderman P, Huppert D, Remington SJ, Tolbert LM, Solntsev KM (2008) The effect of pressure on the excited-state proton transfer in the wild-type green fluorescent protein. *Chem Phys Lett* 455(4–6):303–306. doi:10.1016/j.cplett.2008.02.079
20. Liebovitch LS (1989) Analysis of fractal ion channel gating kinetics – kinetic rates, energy-levels, and activation-energies. *Math Biosci* 93(1):97–115
21. Liebovitch LS (1989) Testing fractal and Markov-models of ion channel kinetics. *Biophys J* 55(2):373–377
22. Liebovitch LS, Fischbarg J, Koniarek JP (1987) Ion channel kinetics – a model based on fractal scaling rather than multistate Markov-processes. *Math Biosci* 84(1):37–68
23. Liebovitch LS, Fischbarg J, Koniarek JP, Todorova I, Wang M (1987) Fractal model of ion-channel kinetics. *Biochim Biophys Acta* 896(2):173–180
24. Liebovitch LS, Sullivan JM (1987) Fractal analysis of a voltage-dependent potassium channel from cultured mouse hippocampal-neurons. *Biophys J* 52(6):979–988
25. Liebovitch LS, Toth TI (1991) Distributions of activation-energy barriers that produce stretched exponential probability-distributions for the time spent in each state of the 2 state reaction a-reversible-B. *Bull Math Biol* 53(3):443–455
26. Maniatis T, Ptashne M (1973) Multiple repressor binding at operators in bacteriophage-lambda – (nuclease protection polynucleotide sizing pyrimidine tracts supercoils E. coli). *Proc Natl Acad Sci U S A* 70(5):1531–1535
27. Manzo C, Finzi L (2010) Quantitative analysis of DNA looping kinetics from tethered particle motion experiments in *Methods in Enzymology*, volume 475 “Molecule Tools, Part B: Super-Resolution, Particle Tracking, Multiparameter, and Force Based Methods”, Ed. Nils G. Walter, pp 199–220.
28. Matthews KS (1992) DNA looping. *Microbiol Rev* 56(1):123–136
29. Maurer R, Meyer BJ, Ptashne M (1980) Gene-regulation at the right operator (Or) of bacteriophage-lambda. I. Or3 and autogenous negative control by repressor. *J Mol Biol* 139(2):147–161

30. Meyer BJ, Maurer R, Ptashne M (1980) Gene-regulation at the right operator (Or) of bacteriophage-lambda.2. Or1, Or2, and Or3 – their roles in mediating the effects of repressor and Cro. *J Mol Biol* 139(2):163–194
31. Nelson PC, Zurla C, Brogioli D, Beausang JF, Finzi L, Dunlap D (2006) Tethered particle motion as a diagnostic of DNA tether length. *J Phys Chem B* 110(34):17260–17267
32. Nickels BE, Dove SL, Murakami KS, Darst SA, Hochschild A (2002) Protein–protein and protein–DNA interactions of sigma(70) region 4 involved in transcription activation by lambda cl. *J Mol Biol* 324(1):17–34
33. Oppenheim AB, Kobiler O, Stavans J, Court DL, Adhya S (2005) Switches in bacteriophage lambda development. *Annu Rev Genet* 39:409–429
34. Ptashne M (2004) A genetic switch: phage lambda revisited, vol 3, 3rd edn. Cold Spring Harbor Laboratory, New York
35. Ptashne M, Gann A (2002) Genes and signals. Cold Spring Harbor Laboratory, New York
36. Qian H (2000) A mathematical analysis for the Brownian dynamics of a DNA tether. *J Math Biol* 41(4):331–340
37. Revet B, von Wilcken-Bergmann B, Bessert H, Barker A, Muller-Hill B (1999) Four dimers of lambda repressor bound to two suitably spaced pairs of lambda operators form octamers and DNA loops over large distances. *Curr Biol* 9(3):151–154
38. Senear DF, Brenowitz M, Shea MA, Ackers GK (1986) Energetics of cooperative protein–DNA interactions – comparison between quantitative deoxyribonuclease footprint titration and filter binding. *Biochemistry* 25(23):7344–7354
39. Stayrook S, Jaru-Ampornpan P, Ni J, Hochschild A, Lewis M (2008) Crystal structure of the lambda repressor and a model for pairwise cooperative operator binding. *Nature* 452(7190):1022–1025
40. van den Broek B, Vanzi F, Normanno D, Pavone FS, Wuite GJL (2006) Real-time observation of DNA looping dynamics of type III restriction enzymes NaeI and NarI. *Nucleic Acids Res* 34(1):167–174
41. Vanzi F, Broggio C, Sacconi L, Pavone FS (2006) Lac repressor hinge flexibility and DNA looping: single molecule kinetics by tethered particle motion. *Nucleic Acids Res* 34(12):3409–3420
42. Vilar JMG, Saiz L (2005) DNA looping in gene regulation: from the assembly of macromolecular complexes to the control of transcriptional noise. *Curr Opin Genet Dev* 15(2):136–144
43. Wang H, Finzi L, Lewis D, Dunlap D (2009) AFM studies of the CI oligomers that secure DNA loops. *J Pharm Biotechnol* 10:494–501
44. Wang Y, Guo L, Golding I, Cox EC, Ong NP (2009) Quantitative transcription factor binding kinetics at the single-molecule level. *Biophys J* 96:609–620
45. Watkins LP, Yang H (2005) Detection of intensity change points in time-resolved single-molecule measurements. *J Phys Chem B* 109(1):617–628
46. Zhang HY, Marko JF (2008) Maxwell relations for single-DNA experiments: monitoring protein binding and double-helix torque with force-extension measurements. *Phys Rev E* 77(3):031916.1–031916.9
47. Zurla C, Franzini A, Galli G, Dunlap DD, Lewis DEA, Adhya S et al (2006) Novel tethered particle motion analysis of CI protein-mediated DNA looping in the regulation of bacteriophage lambda. *J Phys Condens Matter* 18(14):S225–S234
48. Zurla C, Manzo C, Dunlap DD, Lewis DEA, Adhya S, Finzi L (2009) Direct demonstration and quantification of long-range DNA looping by the lambda bacteriophage repressor. *Nucleic Acids Res* 37:2789–2795
49. Zurla C, Samuely T, Bertoni G, Valle F, Dietler G, Finzi L et al (2007) Integration host factor alters LacI-induced DNA looping. *Biophys Chem* 128(2–3):245–252
50. Liebesny P, Goyal S, Dunlap D, Fereydoon family, Finz L, Fereydoon Family, Determination of the Number of Proteins Bound Non-Specifically to DNA. *JPCM* (in press)

## Appendix C: 186 CI paper draft

# A missing link between transcription factors and nucleosomes: the bacteriophage 186 CI repressor wraps and loops DNA

### Introduction

- Idea of a binding specificity continuum.

TFs: Small protein-DNA contact region, high specificity

Nucleosomes. Large protein-DNA contact region, low specificity

186 CI: Large protein-DNA contact region, high specificity

- Nucleosomes:

Structure

Wrapping/unwrapping of DNA

Looping – when relocated

Low sequence specificity

'Sub-nucleosome' binding: the H3-H4 tetramer

Higher order structures: nucleosome-nucleosome interactions (e.g. 30 nm fibre)

- 186 CI:

Structural model

Biochemical info

Regulatory model



- Approach/results

## Materials and Methods

### AFM sample preparation

1584 bp-long DNA fragments were produced by cutting plasmids derived from pBluescript containing wild type 186 operators (FL, FR, pR, pL) with two restriction enzymes: NgoMIV and XmaI (New England BioLabs). The digestion product was isolated and purified (QIAGEN gel purification kit). The position of the midpoint of each operator from one end is: 178bp/56.7nm (FL), 484bp/154.9nm (baricenter of pR. In particular, 463bp/148.2nm (pR1), 484bp/154.9nm (pR2), 505bp/161.6nm (pR 3)), 567bp/181.4nm (pL) and 857bp/274.2nm (FR).

The following forward and reverse primers were used to amplify various DNA fragments as follows: 5'-TTACCGGAGAAGGAGAAGCA-3' and 5'-ATTAATGCAGCTGGCAGCAC-3' (524 bp-long DNA containing only FL), and Biotin5'-CTTTCTTGACGCTTTACGG-3' and 5'-TTTACAAATGCTTCTCCTTCTCC-3' (528 bp-long DNA containing just pR and pL).

Wild-type 186 CI repressor was prepared and purified as described previously [1]. The protein was diluted to the desired final concentration (5nM, 50 and 100 nM) in the presence of 1 nM DNA in a buffer containing 50 mM HEPES, 150 mM NaCl and 0.1 mM EDTA (pH 7.0). All steps were conducted at  $T_{\text{room}}$ . The mixture was incubated for 20 min. The biotin-labelled DNA fragment was incubated in a mixture containing also 1 µg/ml streptavidin. Shortly before deposition, a 10 µl drop of 0.01 µg/ml poly-L-ornithine (1 kDa MW, Sigma-Aldrich, St. Louis, MO) was incubated on freshly cleaved mica for one minute. The poly-L-ornithine-coated mica was then washed with 0.4 ml HPLC water and dried with compressed air. Then, 10 µl of the solution containing DNA and protein were deposited on the poly-L-ornithine-coated mica and incubated for one minute. The droplet was rinsed with 0.4 ml HPLC water and dried gently with compressed air. The sample was left overnight in a desiccator before imaging. Images were acquired with a NanoScope MultiMode AFM microscope (Digital Instrument, Santa Barbara, CA) operated in tapping mode using uncoated, etched silicon tips (MirkoMasch, San Jose, CA). The oscillation amplitude was 50-60 mV with a resonance frequency of 75 kHz (NSC18, MirkoMasch, San Jose, CA). Areas of  $1 \times 1 \mu\text{m}^2$  were scanned at a rate of 1.2 Hz and with a resolution of  $512 \times 512$  pixels. After filtering images to remove scan line offsets and bowing, DNA molecules were interactively traced with NeuronJ [2], a plug-in function for ImageJ [3].

### TPM sample preparation:

1898 bp-long wt or mutated DNA segments were produced by PCR after inserting a relevant fragment from pBluescript into pDL611 (ref). The following primers: 5' TCC AGA GGC GCC GGG GGG TTC GTG CAC ACA G and 5'TGGTAACCTAGGTAAACAAATAGGGGTTCCGCGCAC were used to amplify by pcr the 186 region contained in pBluescript. pDL611 and the pcr product were then digested with EcoR1 and Pst1 in order to insert by ligation the 186 region

from pBluescript into pDL611. The final TPM tether was obtained by pcr using this modified plasmid and the following 5' end biotin and digoxigenin-labeled oligos:

5'-**bio**-CGCAATTAATGTGAGTTAGCTCACTCATTAGGCACCCCAGGC-3' and 5'-**dig**-GCATTGCTTATCAATTTGTTGCAACGAACAGGTCACTATCAGTC-3'

The FL- or FR- DNA fragments contained mutated FL or FR operators to prevent CI binding. In  $\Delta$ pR DNA the region containing the pR binding sites was replaced with an equally long, but unrelated DNA.

The TPM microchamber and experiment were prepared and run as previously described [4-6]. In brief, the glass surface of a microscope flowchamber was coated with biotin-BSA and incubated with streptavidin. DNA tethers were labeled with anti-digoxigenin-coated beads with a diameter of 0.48  $\mu$ m (Indicia Diagnostics, Oullins, France). Interaction of the 186 CI protein with DNA was monitored as a reduction in the amplitude of the Brownian motion of the bead as previously described [4, 7-9].

## Results

### Confirmation of basic model

#### *The repressor wheel.*

The 186 bacteriophage repressor, 186 CI, binds to DNA as a dimer, and it was suggested to assemble into a oligomer of 14 monomers in solution [10]. In support of this suggestion, a crystallographic study showed that the CTD of 186 CI assembles into a wheel of seven dimers (hereafter referred to as the 186 heptamer) [11]. This led to the hypothesis that the whole protein, including the NTD DNA-binding domain, may too form wheel-shaped heptamers. Therefore, AFM was used to image 186 CI free, as well as bound to 1584 bp-long DNA fragments to characterize its shape and dimension. The results, summarized in [Figure 1](#) strongly support the idea that the protein oligomerizes to form wheel-shaped heptamers. Visual inspection of the CI particles in the AFM images shows that not only they have a round shape, consistent with that of a wheel ([Figure 1, left](#)), but also that the average particle diameter is close to the diameter of a wheel composed of seven dimers, as estimated from the X-ray crystal structure of the 186 CI CTD [11] ([Figure 1, center](#)). Furthermore, the volume of the wheels imaged by AFM was measured and compared to a calibration curve previously obtained [12] ([Figure 1, right](#)). Also this volume analysis is consistent with the idea that the wheels may be composed of seven dimers. Finally, since such wheels are very abundant in the images obtained using only 50 nM CI, which is a much lower than the 1100 nM estimated for the lysogen, it is likely that 186 CI associates into a heptamer at an early stage after infection and that this state of assembly is robust through the host cell division.

A 186 heptamer may bind cooperatively to multiple operators [11, 13, 14], giving rise to physiologically relevant nucleoprotein complexes with different structure and conformation, and with different impact on the 186 transcriptional regulatory network. Indeed, the fact that lysogeny maintenance requires repression of pR and tight control of transcription from pL, and that pR and pL face one another, suggests that different nucleoprotein species may be in equilibrium in different repressor concentration regimes, so that the probability of pL being unoccupied decreases with increasing CI concentration [13]. [Figure 2](#) shows the possible species and equilibria that have been suggested, together with AFM images confirming the existence of these complexes.

Understanding the 186 regulatory mechanism requires characterization of the specific interaction of the 186 wheel with the operators FL, pR, and FR and quantification of the probability of occurrence of each species. Thus a statistical analysis of the AFM images acquired was performed. [Figure 3](#) shows that the occupancy of the operators ranks as follows: pR > FL > FR, independently of the DNA conformation that the protein mediated. [Table 1](#) reports the distribution of the nucleoprotein complexes found. The images reveal that the 186 wheel may interact with DNA either by wrapping or by looping it.

#### *Pseudo sites*

The wheel higher affinity for FL than for FR revealed in [figure 3](#) may be explained by cooperativity between FL and an adjacent pseudo site. In agreement with previous DNase digestions [15], closer analysis of the complexes at FL, performed on 524 bp/167.7 nm-long DNA fragments containing only this operator, revealed the presence of a pseudo site on the side away from pR ([Figure 4](#)). The distance from each end of the DNA to the point of contact with the wheel was measured. The distribution of the length of free DNA measured on each end of the bound wheel is shown in [figure 4](#). FL is not centered in these DNA fragments and is closer to the end that points in the direction of pR and FR ([Figure 4, top](#)). Thus, these histograms show that FL and an adjacent pseudo site in the direction away from pR were always occupied. Each distribution shows two peaks separated by about 10 nm. This corresponds to the footprint of one dimer in the wheel since it is one seventh of the perimeter of the 186 heptamer. The left histogram shows that the free DNA on the left of the bound wheel was, in average, either 9.4 or 20.0 nm long. Since FL was centered in this DNA fragment 25 nm from the end in the direction of pR (short end in the diagram in [figure 4](#)), the peak values indicate that one dimer of the wheel binds at FL, leaving approximately 20 nm of free DNA to the left. However, the next 10 nm of this free DNA may bind dynamically to the next dimer in the wheel. On the other hand, the right histogram in [figure 4](#) shows that the free DNA on the right of the bound wheel was, in average, either 122 or 132 nm long. FL was centered 150.6 nm from the end of the DNA fragment away from pR (long end in the diagram in [figure 4](#)). Thus, the peak values indicate that two dimers of the wheel bind both FL and an adjacent pseudo site, leaving approximately 132 nm of free DNA to the right. Ten more nm of this free DNA may bind dynamically to yet the next dimer in the

wheel leaving 122 nm free. On the basis of these observations it is suggested that a pseudo site for binding of the 186 repressor exists next to FL on the side away from pR. Note also that DNA binding to successive dimers around the wheel leads to its wrapping by DNA.

Asymmetric DNA wrapping on the 186 wheel was also observed in 528 bp/179 nm-long DNA fragments that contained only pR (Figure 5). Here, the wheel is not centered on pR because it most often occupies a pseudo site, containing pL, as well. This is consistent with the idea that the protein bound at pR will repress pL leading to 186 CI negative autoregulation, unless competition from distal sites frees the repressor promoter [16].

### *DNA wrapping/unwrapping*

AFM imaging of 1584 bp-long fragments of wt 186 DNA containing all binding sites showed that the degree of wrapping of DNA around the wheel depends on the operator. The 186 wheel bound at pR is most often found to be fully wrapped by DNA (Figure 2, species 2, 3, 6 and 7), while at FL and FR may be more often only partially wrapped such that the DNA going in does not cross over the DNA coming out of the wheel (Figure 2, species 7 and Table 1). However, the wheel may also mediate a loop between either FL or FR and pR (Figure 2, species 4 and Table 1). Furthermore, in the presence of a wheel already wrapped at pR, a second wheel may bridge FL and FR (Figure 2, species 6 and Table 1).

The fully wrapped conformation at pR was observed also by TPM using 1898 bp-long FL<sup>-</sup>.pRpL.FR<sup>-</sup> DNA tethers. Addition of repressor in the microchamber caused an immediate and stable decrease of the TPM signal,  $\rho_{\perp}$ , by 12.2 nm (Figure 6A) which corresponds, according to a calibration curve obtained in identical buffer conditions (Figure S1), to a shortening of the DNA tether of 210 bp. This is the decrease expected for a full wrapping event assuming that each 186 dimer binds 10 nm of DNA and that a heptamer will therefore wrap approximately 70 nm or 210 bp of DNA. This assumption is justified by the structural information available (see above) and by the AFM study on the DNA fragment containing only FL described above. Interestingly, TPM assays performed on 1898 bp-long DNA tethers containing only the FL site (FL.ΔpRpL.FR- DNA) showed a similar stable shortening of about 11.3 nm (Figure 6B). In this case too, the TPM traces recorded did not show transitions between the wrapped and unwrapped conformations as shown by the representative traces (Figure S2), their associated frequency distribution histograms, and by the frequency distribution of the average TPM signal for each of the beads analyzed for the FL.ΔpRpL.FR- DNA tethers in the absence and in the presence of 50 nM 186 CI (Figure 6B).

### *DNA looping*

Although wrapping seems to be preferred (Table 1), AFM images revealed the presence of nucleoprotein complexes including wheel-mediated DNA looping (Figure 2, species 4 and 6). These complexes were classified and their relative weight was measured for wt DNA (FL<sup>+</sup> pR<sup>+</sup> FR<sup>+</sup>), as well as for FL<sup>+</sup> pR<sup>+</sup> FR<sup>-</sup>, where the FR site was mutated, and for FL<sup>+</sup> ΔpR FR<sup>+</sup>, where the pR sequence was replaced with a sequence of equal length that did not bind 186 Cl. The results of this statistical analysis are reported in Tables 1-4. In all cases, DNA wrapping around the repressor is more common than repressor mediated looping. However less probable, the looped species are likely to be physiologically relevant since the pR-FL (or -FR) loop may free pL for transcription, while the FL-FR loop may free pR.

Tables 3&4 show a statistical analysis of AFM images of the DNA fragment carrying only the FL and pR sites. According to the Boltzmann distribution, the ratio between different states,  $S_i$ , in equilibrium depends only on the free energy of each state. If the Cl wheel binds to pR and FL independently, the free energy of the state where both sites are occupied ( $\Delta G_{pR,FL}$ ) should be the sum of free energy changes associated with the formation of each of the other two states: the state with only one wheel bound at pR ( $\Delta G_{pR}$ ) and the state with only one wheel at FL ( $\Delta G_{FL}$ ). Therefore, the population of four states ( $S_1$ : no protein;  $S_2$ : only pR occupied;  $S_3$ : only FL occupied;  $S_4$ : pR and FL both occupied) will be related as follows:

$$S_1/ S_2 = S_3/ S_4$$

Since  $S_4$  is much higher than expected, cooperativity may exist between FL and pR which reduces the free energy of  $S_4$ . The same thing can be (cannot be) argued for FR and pR. Therefore,....

In solution, 186 repressor-mediated looping versus wrapping was investigated by TPM. After addition of repressor to wt 186 DNA, most of the tethers adopted either one of two conformations, characterized by an average decrease in  $\langle \rho_{\perp} \rangle$  of 14.5 nm (most probable) and 37.0 nm, each, which correspond to a shortening of the DNA tether of approx 250 bp and 580 bp, respectively (Figure 6C). The 250 bp shortening is greater than the one associated with a full wrapping event. Thus, it could result from a wrapping event at the strong pR sites and a partial wrapping at one of the flanking sites as well as from a looping event between pR and either FL or FR. In this respect, notice that the histogram is quite broad. The 580 bp shortening may be interpreted as due to the wrapping of the DNA around three wheels bound one to each operator (FL, pR and FR) or to the formation of a loop between FL and FR, since the distance between the centers of these two operators is 678 bp. Notice that in this looped state, a second wheel may be bound at pR, but would not cause a detectable TPM signal. Out of 31 molecules that were analyzed, only 5 displayed just one or two transitions between the two states in 20 min of observation, but never back to the free DNA state. Their frequency histogram was, therefore, bimodal. Although TPM measurements did not show all the nucleoprotein complexes revealed by AFM, one should notice that the TPM histograms are quite broad, and it is possible that several nucleoprotein complexes, including the loop between pR and one of the flanking sites, coexist in equilibrium, without being clearly resolved by TPM.

TPM measurements performed on DNA tethers containing only FL and pR (Figure 6D), showed a 14.5 nm decrease in  $\langle \rho_{\perp} \rangle$ , corresponding to 245 bp shortening of the DNA tether. This shortening, as already discussed for the wt case, may be interpreted as due to a full wrapping event, probably at pR, which is stronger than FL accompanied by a partial wrapping at FL. However, the broad TPM frequency distribution histogram may also be consistent with a loop which was dynamically forming and breaking between FL and pR. This loop would consume some 300 bp of DNA if the two binding sites came in direct contact, but the wheel would reduce the observed shortening. Indeed, three of 44 FL<sup>+</sup>.pRpL.FR<sup>-</sup> DNA tethers display two peaks, one at 18.9 nm and the other at 0, respectively, and can be explained by the transition between the looped and the unlooped DNA at FL and pR.

TPM of FL+ ΔpR FR+ DNA was also performed (Figure 6E). These molecules are not expected to bind the 186 wheel at pR. DNA tethers which displayed just one peak after addition of repressor could be separated into two groups. One group showed an average decrease in  $\langle \rho_{\perp} \rangle$  of 24.9 nm, corresponding to 410 bp shortening of DNA tether. This may be consistent with two fully wrapped wheels at FL and FR. This could happen since the ratio between 186 monomer to DNA is 50:1, one wheel needs 14 monomers to form, and there is a complex equilibrium between several protein oligomerization states which lowers the number of wheels in solution. Therefore, in these conditions of CI concentration, one DNA may in average have 2-3 wheels. If there is no pR, FL and FR may always be occupied. This would prevent loop formation by just one wheel bound simultaneously at FL and at FR. However, this latter, looped conformation, may be induced in some of the tethers and explain the broadness of the histogram.

Another group of DNA tethers showed an average 7.0 nm decrease of  $\langle \rho_{\perp} \rangle$ , which, is shorter than that expected for a full wrapping event, but, considering the standard deviation of the data, could be due to a single wheel partially wrapped at FR or FL. Once again, TPM seems to reveal fewer nucleoprotein complexes than AFM. In particular, the loop between the two flanking sites was not distinctly detected in the TPM measurements performed on this mutated 186 DNA fragment, and the proportion between one wrapped and two wrapped wheels is not the same as in the AFM images despite the similar DNA/repressor concentration ratio in the two types of measurements.

The overall interpretation of all these observations should not neglect to consider the possible role of nonspecific binding. An occupancy analysis, performed on the AFM images of the FL+ ΔpR FR+ DNA (Figure 7), revealed several weaker binding sites, which may play a role in shaping the equilibria between the nucleoprotein complexes involving FL, pR and FR. Indeed, DNA loops between a specific and a nonspecific site were observed by AFM in the absence of pR (Table 2). Therefore, the histograms of TPM signals may be broadened also by transient interactions with nonspecific sites which may have the physiological role of facilitating and/or stabilizing specific interactions that regulate the 186 bacteriophage genetic switch.

## Other CI binding forms and non-specific binding

The 186 repressor can bind non-specifically, just as many prokaryotic repressors and probably most transcriptional factors. This ability is clear from the analysis of AFM images of FL<sup>-</sup>.ΔpRpL.FR<sup>-</sup> and 186 CI nucleoprotein complexes (Figure 7) at 50 nM and from the beads-on-a-string fiber that 186 DNA forms in the presence of 300 nM repressor (Figure 8). Non specific binding is eliminated when using 186 CI mutant (Table 5). AFM imaging also showed there is some kind of non specific interaction between wild type protein and non-related DNA (lambda) or the FL- delta pR-pL FR- DNA.

## Discussion (outline)

Several 186 DNA-repressor nucleoprotein complexes were revealed by AFM and TPM. TPM measurements on several tethers showed that protein-induced DNA remodeling is stable. Indeed transitions back to the DNA unbound were extremely rare. Also transitions between wrapping and looping were rare. However, the collective histograms of several DNA tethers in the presence of protein are broad and may represent several degrees of wrapping, where there is only one binding site, and did not allow distinction of different species when multiple types of multiprotein complexes were consistent with the observed shortening.

Pseudo sites play a role in stabilizing some of these complexes and, thus, in shaping the relative equilibria.

Discussion of physiological relevance of these complexes.

.....

## Bibliography

1. Shearwin, K.E. and J.B. Egan, *Purification and self-association equilibria of the lysis-lysogeny switch proteins of coliphage 186*. J Biol Chem, 1996. **271**(19): p. 11525-31.
2. E. Meijering, M.J., J. C. Sarria, P. Steiner, H. Hirling, and M. Unser, *Design and Validation of a Tool for Neurite Tracing and Analysis in Fluorescence Microscopy Images*. Cytometry A, 2004. **58**: p. 9.
3. Rasband, W.S., *Image J*. 2008.

4. Dunlap, D., et al., *Probing DNA topology with Tethered particle Motion*, in *Methods in Molecular biology: Single-Molecule Analysis: Methods and Protocols*, E.J.G. Peterman and G. Wuite, Editors, Humana Press.
5. Finzi, L. and D. Dunlap, *Single-molecule studies of DNA architectural changes induced by regulatory proteins*. *Methods Enzymol*, 2003. **370**: p. 369-78.
6. Zurla, C., Manzo, C, Dunlap, DD, Lewis, DEA, Adhya, S, Finzi, L, *Direct Demonstration and Quantification of Long-Range DNA looping by the Lambda Bacteriophage Repressor*. *Nucleic Acids Res*, 2009. **37**: p. 2789-2795.
7. Finzi, L. and D.D. Dunlap, *Single-molecule approaches to structure, kinetics and thermodynamics of transcriptional regulatory nucleoprotein complexes*". *J. Biol. Chem.*, 2010. **285**: p. 18973-18978.
8. Manzo, C. and L. Finzi, *Quantitative analysis of DNA looping kinetics from tethered particle motion experiments*, in *Methods In Enzymology: Molecule Tools, Part B: Super-Resolution, Particle Tracking, Multiparameter, and Force Based Methods*, N.G. Walter, Editor. 2010, Academic Press, Elsevier. p. 199-220.
9. Nelson, P.C., et al., *Tethered particle motion as a diagnostic of DNA tether length*. *Journal Of Physical Chemistry B*, 2006. **110**(34): p. 17260-17267.
10. Egan, K.E.S.a.J.B., *Purification and self-association equilibria of the lysis-lysogeny switch proteins of coliphage 186*. *The Journal of Biological Chemistry*, 1996. **271**(May 10): p. 7.
11. Pinkett, H.W., et al., *The structural basis of cooperative regulation at an alternate genetic switch*. *Mol Cell*, 2006. **21**(5): p. 605-15.
12. Wang, H., Finzi, L., Lewis, D. and Dunlap, D., *AFM studies of the CI oligomers that secure DNA loops*. *J. Pharmaceutical Biotechnology*, 2009. **10**: p. 494-501.
13. Dodd, I.B., K.B. Shearwin, and K. Sneppen, *Modelling transcriptional interference and DNA looping in gene regulation*. *Journal Of Molecular Biology*, 2007. **369**(5): p. 1200-1213.
14. Egan, I.B.D.a.J.B., *Action at a distance in CI repressor regulation of the bacteriophage 186 genetic switch*. *Molecular Microbiology*, 2002. **45**(3): p. 14.
15. Egan, I.B.D.a.J.B., *DNA binding by the coliphage 186 repressor protein CI*. *The Journal of Biological Chemistry*, 1996. **271**(May 10): p. 9.
16. Ian B. Dodd, K.E.S.a.K.S., *Modelling transcriptional interference and DNA looping in gene regulation*. *J. Mol. Biol.*, 2007. **369**: p. 14.



## Appendix D: Source code for measurement of blob volume from AFM images and polymer chain and particle simulation

### volum\_C.m

```
001    % Manually select particle region and calculate particle volume.
002
003    % Authur: Haowei Wang (hwang23@emory.edu)
004    % Last updated Sep. 10th, 2011
005
006    clear all
007    close all
008
009    % Set parameters.
010
011    z_scale=8.0;
012    pixel_area=(1000/512)^2;
013    v_result=rand(10,5); %#ok<NASGU>
014    v_result=0.0;
015
016    % Begin to proceed
017    for n=1:10
018        % Display file number
019        n
020        expand=input('Input filename: ', 's');
021        if expand=='100'
022            break;
023        end
024        % Read image data
025        image = readimage(strcat('44citks8.',expand));
026        pcolor(image);
027        shading flat
028
029        % Select the particle by mouse clicking
030        range_raw=ginput;
031        range = range_raw(end-1:end,:);
032        sub_image = image( range(1,2):range(2,2), range(1,1):range(2,1));
033        pcolor(sub_image);
034        shading flat
035
036        % Input the threshold for background, input '0' to end the changing
037        % (last selcted value will be perserved).
038        bearing = -5000;
039        while bearing~=0
040            sub_i=sub_image;
041            bearing = input('Input Bearing: ');
042            if bearing~=0
043                bearing_f=bearing;
044                sub_i(sub_i<bearing)=bearing;
045                pcolor(sub_i);
046                shading flat
047            end
048        end
049        baseline=sub_image(:);
050        base=mean(baseline(baseline<bearing_f));
```

```

051
052 threshold = -5000;
053
054 % Input the threshold for DNA height, input '0' to end the changing
055 % (last selected value will be preserved).
056
057 while threshold~=0
058     sub_i=sub_image;
059     threshold = input('Input Threshold: ');
060     if threshold~=0
061         threshold_f=threshold;
062         sub_i(sub_i<threshold) = base;
063         pcolor(sub_i);
064         shading flat
065     end
066 end
067
068 % Eliminate pixels not belong to the particle.
069 sub_i=sub_image;
070 sub_i(sub_i<threshold_f)=base;
071 pcolor(sub_i);
072 shading flat
073 elimin=input('Eliminate? y=1,n=0: ');
074 while elimin==1
075     range_raw=ginput;
076     range = range_raw(end-1:end,:);
077     sub_i( range(1,2):range(2,2), range(1,1):range(2,1))=base;
078     pcolor(sub_i);
079     shading flat
080     elimin=input('Eliminate? y=1,n=0: ');
081 end
082
083 % Calculate volume, area and height
084 sub_i=sub_i-base;
085 volume=sum(sum(sub_i))*pixel_area*z_scale/65536;
086 v_result(n,4)=max(max(sub_i))*z_scale/65536;
087 v_result(n,5)=(threshold_f-base)*z_scale/65536;
088 sub_i(sub_i>0)=1;
089 area=sum(sum(sub_i))*pixel_area;
090 v_result(n,1)=str2double(expand);
091 v_result(n,2)=volume;
092 v_result(n,3)=area;
093 expand %#ok<NOPTS>
094 % result2: block volume;
095 % result3: block area;
096 % result4: the highest peak of the block;
097 % result5: the threshold, ie. the height of the DNA.
098 end

```

### **hundredsM.M**

```

01 % This program simulates polymer chain with worm-like-chain model.
02 mNum=300;
03 length=1500;
04 data=zeros(length,2,mNum);
05 Dstd=sqrt(0.34/25);

```

```

06
07 for i=1:mNum
08
09 % Start point is set to (0,0)
10 data(1,1,i)=0;
11 data(1,2,i)=0;
12
13 % Initial the start direction
14 direction=rand*2.0*pi;
15 for j=2:length
16
17 % The polymer walk one step in each round following the direction
18 % provided by Worm-Like-Chain
19 data(j,1,i)=0.34*cos(direction)+data(j-1,1,i);
20 data(j,2,i)=0.34*sin(direction)+data(j-1,2,i);
21 direction=direction+normrnd(0,Dstd);
22 end
23
24 % Plot the simulated polymer.
25 plot(data(:,1,i),data(:,2,i));
26 hold on
27 end

```

### imageG.m

```

01 % This program scans the simulated polymer chain with a virtual tip
02 % and convert it into 512*512 images.
03
04 close all
05 clear all
06
07 load data
08
09 range=65536;
10 zScale=8;
11 xyScale=512/1000;
12
13 rDNA=1.0;
14 rProbe=2.7;
15
16 % End of parameter initiation.
17
18 data=data*xyScale+256;
19 lnd=size(data);
20
21 AffectRange=8;
22
23 % lnd(3) is the total number of simulated polymers
24
25 for i=1:lnd(3)
26 image=zeros(512,512);
27
28 % The program looks at polymers one by one. Every polymers will be
29 % saved into a TIF file at the end.
30 % lnd(1) is the number of points inside each polymer chain.
31 % The program looks polymer chains as a group of points. Every

```

```

32 % point has a radius equal to rDNA. Detected height of each pixels
33 % will be recorded and the program picks the highest value to
34 % establish the topology of polymer.
35
36     for j=1:Ind(1)
37         center=round(data(j,:,i));
38         res=data(j,:,i)-center;
39         for k=-5:5
40             for l=-5:5
41                 dis2=((k-res(1))^2+(l-res(2))^2)/xyScale^2;
42                 height2=(rDNA+rProbe)^2-dis2;
43                 if height2<0
44                     height2=0;
45                 end
46                 height=sqrt(height2)-rProbe;
47
48                 if image(center(1)+k,center(2)+l)<height
49                     image(center(1)+k,center(2)+l)=height;
50                 end
51             end
52         end
53     end
54
55 % Adding random noise.
56
57     image=image+rand(512)*0.4;
58     image=image*range/zScale;
59
60 %   pcolor(image);
61 %   shading flat
62 %   hold on
63 %   plot(data(:,2,i),data(:,1,i));
64
65 % Scaling to 256 degree of brightness.
66     MaxImage=max(max(image));
67     scaleImage=image*254/MaxImage;
68     Nimage=uint8(scaleImage);
69
70 % Saving TIF image.
71
72     imwrite(Nimage, strcat('Mole', int2str(i), '.tif'), 'ColorSpace', 'cielab', 'Compression',
'none');
73     end

```

### ellipsoid.m

```

01 % This program simulates half-ellipsoid particles and save it into
02 % TIF images.
03
04 % Authur: Haowei Wang (hwang23@emory.edu)
05 % Last updated Sep. 10th, 2011
06
07 close all
08 clear all

```

```

09
10     area=30;
11
12     range=65536;
13     zScale=8;
14     xyScale=512/1000;
15     rProbe=8.0;
16
17     % End of preparing parameters.
18
19     image=zeros(512,512);
20
21     for i=1:area
22         for j=1:area
23
24             % begin to calculate the height of the point
25             rParticle=9;
26             MaxHeight=4.5;
27             distance2=(15-i)^2+(15-j)^2;
28             if distance2<rParticle^2
29                 heightP=MaxHeight/2+MaxHeight/2*sqrt(1-distance2/rParticle^2);
30             else heightP=0;
31             end
32             % calculate the height of the point
33
34             % begin to scan the point
35             PositionI=[i*512/1000, j*512/1000];
36             PositionP=round(PositionI);
37             res=PositionI-PositionP;
38             PositionP=PositionP+area;
39             for k=-5:5
40                 for l=-5:5
41                     dis2=((k-res(1))^2+(l-res(2))^2)/xyScale^2;
42                     height2=(heightP+rProbe)^2-dis2;
43                     if height2<0
44                         height2=0;
45                     end
46                     height=sqrt(height2)-rProbe;
47
48             % The program only record the highest effect generated by pixels
49             % belong to a particle.
50                 if image(PositionP(1)+k,PositionP(2)+l)<height
51                     image(PositionP(1)+k,PositionP(2)+l)=height;
52                 end
53             end
54         end

```

```

55         % end of scan
56     end
57 end
58
59 image=image+rand(512)*0.4;
60 image=image*range/zScale;
61
62 % Scaling to 256 degree of color
63
64 MaxImage=max(max(image));
65 scaleImage=image*254/MaxImage;
66 Nimage=uint8(scaleImage);
67
68 imwrite(Nimage, 'Mole.tif', 'ColorSpace', 'cielab', 'Compression', 'none');

```

### **slope.m**

```

01 % This program simulate cornic particle and save it in TIF image
02 % for the study of particle diameter under AFM.
03
04 % Authur: Haowei Wang (hwang23@emory.edu)
05 % Last updated Sep. 10th, 2011
06 close all
07 clear all
08
09 area=30;
10
11 range=65536;
12 zScale=8;
13 xyScale=512/1000;
14
15
16 rProbe=2.5;
17
18 image=zeros(512,512);
19
20 for i=1:area
21     for j=1:area
22
23         % begin to calculate the height of the point
24         rParticle=7.1;
25         MaxHeight=3.5;
26         distance2=(15-i)^2+(15-j)^2;
27         if distance2<rParticle^2
28             heightP=MaxHeight/rParticle*(rParticle-sqrt(distance2));
29         else heightP=0;

```

```

30     end
31     % calculate the height of the point
32
33     % begin to scan the point
34     PositionI=[i*512/1000, j*512/1000];
35     PositionP=round(PositionI);
36     res=PositionI-PositionP;
37     PositionP=PositionP+area;
38     for k=-5:5
39         for l=-5:5
40             dis2=((k-res(1))^2+(l-res(2))^2)/xyScale^2;
41             height2=(heightP+rProbe)^2-dis2;
42             if height2<0
43                 height2=0;
44             end
45             height=sqrt(height2)-rProbe;
46
47             % Keep the higher value for each pixels.
48             if image(PositionP(1)+k,PositionP(2)+l)<height
49                 image(PositionP(1)+k,PositionP(2)+l)=height;
50             end
51         end
52     end
53     % end of scan
54 end
55 end
56
57 % Scaling to 256 degree of color;
58 image=image+rand(512)*0.1;
59 image=image*range/zScale;
60 MaxImage=max(max(image));
61 scaleImage=image*254/MaxImage;
62 Nimage=uint8(scaleImage);
63
64 imwrite(Nimage, 'Mole.tif', 'ColorSpace', 'cielab', 'Compression', 'none');
65 pcolor(image);
66 shading flat
67
68 figure
69 bar=1:61;
70 plot(bar,image(8:68,39));
71 subimage=image(20:80,20:80);
72 figure
73 surf(subimage);

```

## Appendix E: Source code DNA protein analysis toolbox

### tracing.m

```
001
002 % This program calculate the basal level(background) and overwhelm value
003 % (most of the case, DNA height).
004
005 % The file structure of traces contains all traces of each image. The first
006 % raw is [BASE OVERWHELM]. The tracing data are put as follow: the first
007 % row of every segment is [-1 traceID]; the last row is [-1 0]. Trace ID is
008 % a postive number generated by tracing program to identify each segments
009 % in one image.
010 % Tracing files are saved as *tr.txt, where * represents the original image
011 % filename.
012
013 % This program need subfunction "thresCal.m", please keep it in the same
014 % folder of the program.
015
016 % Authur: Haowei Wang (hwang23@emory.edu)
017 % Last updated Sep. 10th, 2011
018
019 clear all
020 close all
021
022 Button='Yes';
023
024 Auto='Yes';
025 % If Auto is set to 'Yes', then the program will not ask for parameters.
026
027 [filename, pathname, filterindex]=uigetfile('*. *', 'pick a file', 'Multiselect', 'on');
028
029 currentP=pwd;
030 path(path,currentP);
031
032 cd(pathname);
033
034 if iscell(filename)
035     fileNum=size(filename, 2);
036 else
037     fileNum=1;
038 end
039
040 minLength=10;
041
042 for traceN=1:fileNum
043     if fileNum==1
044         file=filename;
045     else
046         file=char(filename(traceN));
047     end
048     if ~isequal(file, 0)
049         IM = readimage(file);
```



```

050     trF=figure;
051     pcolor(IM);
052     shading flat
053     thresC=thresCal(IM);
054
055     if thresC.error==1
056         BASE=round(mean(mean(IM)));
057         OVERWHELM=round(mean(max(IM)));
058     else
059         BASE=round(thresC.base);
060         OVERWHELM=round(thresC.overwhelm);
061     end
062
063     % Set threshold for the background. All pixels below BASE will be
064     % considered as background
065
066     BASE=BASE+round((OVERWHELM-BASE)*0.08);
067
068     % Set up original value for threshold modification.
069     if Auto(1)=='Y'
070         BASE1=BASE;
071         OVERWHELM1=OVERWHELM;
072     else
073         BASE1=-1;
074         OVERWHELM1=-1;
075     end
076
077     % If 'Auto' is set to 'Yes' at the beginning of the code, the program will
078     % skip this part of code and automatically trace all selected images with
079     % calculated threshods without asking user. Otherwise, the while loop below
080     % will be excuted and a question dialog will present so that the user can
081     % change and compare different sets of thresholds.
082
083     while true
084
085         if BASE==BASE1 && OVERWHELM==OVERWHELM1
086             break;
087         else
088             image=IM;
089             image(image<BASE)=BASE;
090             baseF=figure('position',[10 150 560 420]);
091             newp=pcolor(image);
092             title('BASE Cutting');
093             shading flat
094
095             overF=figure('position',[590 150 560 420]);
096             image=IM;
097             image(image<OVERWHELM)=BASE;
098             newp2=pcolor(image);
099             title('Overwhelming');
100             shading flat
101
102             def={num2str(BASE),num2str(OVERWHELM)};
103             prompt={'Enter new BASE:', 'Enter new OVERWHELM:'};
104             answer=inputdlg(prompt, 'Change parameter', 1, def);
105             BASE1=BASE;

```

```

106         OVERWHELM1=OVERWHELM;
107         BASE=str2double(char(answer(1)));
108         OVERWHELM=str2double(char(answer(2)));
109         close(baseF);
110         close(overF);
111     end
112 end
113 end
114
115     THRESHOLD=0.4; % set the default threshold value in percentage.
116     ReMax=1500; % the maximum number of points on one tracing
117     Record=zeros(ReMax,2);
118
119 % If the threshold is set to a very low number by mistake, the program will
120 % correct it with 30%
121
122     if (THRESHOLD<=0.05)||((THRESHOLD>=1)
123         THRESHOLD=0.3;
124     end
125     traceNum=0;
126     TracingWindow=zeros(1,2);
127
128     test{1}=[-1 -1 0; 0 -1 0; 0 0 0];
129     test{2}=[0 -1 -1; 0 -1 0; 0 0 0];
130     test{3}=[0 0 -1; 0 -1 -1; 0 0 0];
131     test{4}=[0 0 0; 0 -1 -1; 0 0 -1];
132     test{5}=[0 0 0; 0 -1 0; 0 -1 -1];
133     test{6}=[0 0 0; 0 -1 0; -1 -1 0];
134     test{7}=[0 -1 0; 0 -1 -1; 0 0 0];
135     test{8}=[0 0 0; 0 -1 -1; 0 -1 0];
136     %left test arrays
137
138     image=IM;
139     pcolor(image);
140     shading flat
141
142     Msize=size(image);
143     mask=zeros(Msize);
144
145     biColor=image;
146     biColor(biColor<(OVERWHELM-BASE)*THRESHOLD+BASE)=0;
147 %   biColor(biColor>OVERWHELM)=0;
148     biColor(biColor>0)=1;
149
150     biColor(:,1)=0;
151     biColor(:,Msize(2))=0;
152     biColor(:,Msize(2)-1)=0;
153     biColor(1,:)=0;
154     biColor(Msize(1),:)=0;
155     biColor(Msize(1)-1,:)=0;
156     %clear the edge
157
158     modify=-1;
159
160 % Begin to thinning. 'modify' is set to zero at the beginning of each
161 % round. After each round modify is set to -n where n is the number of

```

```

162 % pixels removed in the round. The while-loop will excute untill there is
163 % no pixel removed in one round.
164
165     while (modify<0)
166         modify=0;
167
168 % The image matrix will rotate 90 degree in each round of this for-loop. By
169 % this mean, pixels will be removed ordially from each side of the image.
170 % After four round, the image will come back to origin direction.
171     for i=1:4
172
173 % Pick up pixels insde the image one by one and begin to test.
174         for j=2:Msize(2)-1
175             for k=2:Msize(1)-1
176
177 % The pixel has value and in the left edge of a blob will be tested for
178 % removal.
179                 if (biColor(k,j)>0)
180                     if(biColor(k,j-1)==0)
181
182 % Begin to test if the pixel can be removed. If the answer is yes, the
183 % relat position in mask will be set to '-1'.
184
185                         around=biColor(k-1:k+1,j-1:j+1);
186                         sumR=sum(sum(around));
187                         if (sumR>3)
188                             mask(k,j)=-1;
189                             if (around(1,1)==1)
190                                 if (around(1,2)==0)
191                                     mask(k,j)=0;
192                                 end
193                             end
194                             if (around(1,2)==1)
195                                 if (around(1,1)+around(1,3)+around(2,3)==0)
196                                     mask(k,j)=0;
197                                 end
198                             end
199                             if (around(1,3)==1)
200                                 if(around(1,2)+around(2,3)==0)
201                                     mask(k,j)=0;
202                                 end
203                             end
204                             if (around(2,3)==1)
205                                 if(around(1,2)+around(1,3)+around(3,2)+around(3,3)==0)
206                                     mask(k,j)=0;
207                                 end
208                             end
209                             if (around(3,3)==1)
210                                 if (around(2,3)+around(3,2)==0)
211                                     mask(k,j)=0;
212                                 end
213                             end
214                             if (around(3,2)==1)
215                                 if (around(3,1)+around(2,3)+around(3,3)==0)
216                                     mask(k,j)=0;
217                                 end

```

```

218         end
219         if (around(3,1)==1)
220             if (around(3,2)==0)
221                 mask(k,j)=0;
222             end
223         end
224
225         if(sumR>4)
226             if (around(2,3)==0)
227                 mask(k,j)=0;
228             end
229
230         end
231 % This part preserve the pixels that can break the skeleton if be removed
232         if sumR==4
233             AdjTest=around(1,1)*around(1,2)+around(1,2)*around(1,3);
234             AdjTest=AdjTest+around(1,3)*around(2,3)+around(2,3)*around(3,3);
235             AdjTest=AdjTest+around(3,3)*around(3,2)+around(3,2)*around(3,1);
236             if AdjTest==0
237                 mask(k,j)=-1;
238             end
239         end
240
241 % This part removes the pixels split the skeleton
242
243         elseif (sumR==3)
244             for l=1:8
245                 testAround=abs(around+test{l});
246                 if (sum(sum(testAround))==0)
247                     mask(k,j)=-1;
248                     break;
249                 end
250             end
251         end
252 %         if (mask(k,j)==-1)
253 %             around
254 %         end
255
256     end
257
258     end
259 end
260 end
261
262 pcolor(biColor);
263 shading flat
264
265 % Add mask to biColor matrix to set the selected pixels to zeros (remove
266 % selected pixels).
267     biColor=biColor+mask;
268     modify=modify+sum(sum(mask));
269     Msize=Msize*[0 1; 1 0];
270     biColor=rot90(biColor);
271     mask=zeros(Msize);
272 % Rotate 90 degree to repeat
273     end

```

```

274     end
275
276 % Display tracing result.
277     image=image.*(1-biColor);
278
279     pcolor(image);
280     shading flat
281
282 % User can choose to save the traces or not here. If 'Auto' was set to
283 % 'Yes' at the beginning of the code, the program will skip the question
284 % dialog and save all traces automatically.
285     if Auto(1)=='N'
286         Button=questdlg('Save tracing?');
287     end
288     if Button(1)=='Y'
289         fd=fopen(strcat(file, 'tr.txt'), 'w');
290         fprintf(fd, '%d %d \n\r', round([BASE OVERWHELM]));
291
292 % The program will go through the whole image to look for a start end for
293 % traces.
294     for i=2:Msize(1)-1
295         for j=2:Msize(2)-1
296             if biColor(i,j)==1
297                 around=biColor(i-1:i+1,j-1:j+1);
298                 if sum(sum(around))==2
299                     Record(1,1)=i;
300                     Record(1,2)=j;
301                     Record(1,:)=Record(1,:);
302                     traceNum=traceNum+1;
303                     m=2;
304                     k=i;
305                     l=j;
306                     testNext=[1 2 3; 4 5 6; 7 8 9];
307                     while (sum(sum(around))>1)
308                         biColor(k,l)=0;
309                         around(2,2)=0;
310                         tNext=sum(sum(testNext.*around));
311
312                     switch tNext
313                         case 1
314                             k=k-1; l=l-1;
315                         case 2
316                             k=k-1;
317                         case 3
318                             k=k-1; l=l+1;
319                         case 4
320                             l=l-1;
321                         case 6
322                             l=l+1;
323                         case 7
324                             k=k+1; l=l-1;
325                         case 8
326                             k=k+1;
327                         case 9
328                             k=k+1; l=l+1;
329                     otherwise

```

```

330     %             traceNum=traceNum-1;
331                 Record=zeros(ReMax,2);
332                 break;
333             end
334             Record(m,1)=k;
335             Record(m,2)=l;
336             Record(m,:)=Record(m,:);
337             m=m+1;
338             if m==ReMax
339                 break;
340             end
341             around=biColor(k-1:k+1,l-1:l+1);
342         end
343         biColor(k,l)=0;
344         if m<minLength
345             traceNum=traceNum-1;
346             Record=zeros(ReMax,2);
347         elseif Record(1,1)+Record(1,2)==0
348             m=1;
349         else
350             fprintf(fd, '%d %d \n\r', -1, traceNum);
351             for n=1:ReMax
352                 if Record(n,1)+Record(n,2)==0
353                     fprintf(fd, '%d %d \n\r', -1, 0);
354                     break;
355                 end
356                 fprintf(fd, '%d %d \n\r', Record(n,:));
357             end
358             Record=zeros(ReMax,2);
359         end
360     end
361 end
362 end
363 end
364 fclose(fd);
365 end
366 close(trF);
367 end
368 cd(currentP);

```

## Readimage.m

```

01     % This function read images from AFM data and export it in a double-precision matrix.
02
03     % Authur: Haowei Wang (hwang23@emory.edu)
04     % Last updated Sep. 10th, 2011
05     function data = readimage(filename)
06
07     f = fopen(filename);
08     magnify=1;
09
10     % Read file head and find out scaling factor ('magnify').
11     while true
12         line=fgets(f);
13         if size(line,2)<13

```

```

14     continue;
15     end
16     if strcmp('\@Z magnify:', line(1:12))
17         n=size(line,2);
18         magnify=str2double(line(28:n));
19         break;
20     end
21     if strcmp(line, '\*File list end')
22         break;
23     end
24 end
25 fseek(f, 0, 'bof');
26
27 fread(f, 40960, 'int8');
28
29 % Read data
30 data_unshaped = fread(f, 512*512, 'int16');
31
32 data=reshape(data_unshaped, 512, 512);
33
34 % Rotating and scaling
35
36 data=rot90(data)/magnify;
37 fclose(f);
38
39 % Codes above for data obtained by AFM. If the user wants to use other
40 % image file formate, simply replace the code above with users' code.
41
42 % NOTICE: the output data should be a square array of double. Otherwise,
43 % the program may not working.
44
45 % data=readtif(filename);

```

### **thresCal.m**

```

01 % This function calculate thresholds for DNA tracing (BASE and OVERWHELM
02 % value).
03
04 % Authur: Haowei Wang (hwang23@emory.edu)
05 % Last updated Sep. 10th, 2011
06
07 function out=thresCal(dataM)
08
09 data=dataM(:);
10 out.base=mean(data);
11 out.error=0;
12
13 pr_V=data;
14 lower=min(pr_V);
15 upper=max(pr_V);
16 binS=round(sqrt(upper-lower)/1.2);
17 ave=mean(pr_V);
18 aLim=size(pr_V,1)/10;
19

```

```

20 % Put all pixels into a histogram by the height.
21
22 xout=[lower:binS:upper*1.2];
23 n=histc(pr_V,xout);
24 % bar(xout,n)
25 % hold on
26
27 % Fit the histogram with Gaussian curve. The center of the Gaussian will be
28 % the level of background.
29
30 try
31 cfun=fit(xout',n,'gauss1','Lower',[0 lower 0],'Upper',[aLim 2*ave-lower upper-ave]);
32 catch
33     out.error=1;
34     out.overwhelm=0;
35     return
36 end
37 % xout2=lower:1:upper*1.2;
38 % f2=feval(cfun,xout2');
39 % plot(xout2,(f2),'r-','Linewidth',2);
40
41 % hold off
42 % pause
43
44 if abs(cfun.b1-ave)>cfun.c1/1.5
45     out.error=1;
46 else
47
48 % Look for the range of main peak and cut it away.
49
50     lowBound=cfun.b1+cfun.c1*4;
51     if lowBound>=max(data)
52         out.error=1;
53         out.overwhelm=0;
54         return
55     end
56
57     pr_V=data(data>lowBound);
58     lower=min(pr_V);
59     upper=max(pr_V);
60
61 % Make another histogram of residue tail.
62     binS=sqrt(upper-lower)/1.2;
63     xout=lower:binS:upper*1.2;
64     n=histc(pr_V,xout);
65 %     bar(xout,n)
66
67 % Look for the sudden drop point on the residue tail.
68     uppB=max(n);
69     n(n>uppB*0.5)=0;
70     [max1, i1]=max(n);
71     n(n>uppB*0.3)=0;
72     [max2,i2]=max(n);
73
74 % Fit the drop with linear regression and find out the interction on X
75 % axis. This is the value of OVERWHELM.

```



```

76     if i2<i1
77         out.error=1;
78     elseif max2>=max1
79         out.error=1;
80     else
81         i=(max1*i2-max2*i1)/(max1-max2);
82         out.overwhelm=binS*i+lowBound;
83     end
84 end

```

## Mask.m

```

0001 % This program is written for modificating DNA traces generated by tracing
0002 % program. Users can use it to delete slected traces, connect traces
0003 % together and measure the segment length of a trace or a part of it.
0004
0005 % Authur: Haowei Wang (hwang23@emory.edu)
0006 % Last updated Sep. 10th, 2011
0007
0008 function varargout = maskM(varargin)
0009 % MASKM M-file for maskM.fig
0010 %   MASKM, by itself, creates a new MASKM or raises the existing
0011 %   singleton*.
0012 %
0013 %   H = MASKM returns the handle to a new MASKM or the handle to
0014 %   the existing singleton*.
0015 %
0016 %   MASKM('CALLBACK',hObject,eventData,handles,...) calls the local
0017 %   function named CALLBACK in MASKM.M with the given input arguments.
0018 %
0019 %   MASKM('Property','Value',...) creates a new MASKM or raises the
0020 %   existing singleton*. Starting from the left, property value pairs are
0021 %   applied to the GUI before maskM_OpeningFcn gets called. An
0022 %   unrecognized property name or invalid value makes property application
0023 %   stop. All inputs are passed to maskM_OpeningFcn via varargin.
0024 %
0025 %   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
0026 %   instance to run (singleton)".
0027 %
0028 % See also: GUIDE, GUIDATA, GUIHANDLES
0029
0030 % Edit the above text to modify the response to help maskM
0031
0032 % Last Modified by GUIDE v2.5 09-Aug-2011 11:44:28
0033
0034 % Begin initialization code - DO NOT EDIT
0035 gui_Singleton = 1;
0036 gui_State = struct('gui_Name',    mfilename, ...
0037                   'gui_Singleton', gui_Singleton, ...
0038                   'gui_OpeningFcn', @maskM_OpeningFcn, ...
0039                   'gui_OutputFcn', @maskM_OutputFcn, ...
0040                   'gui_LayoutFcn', [] , ...
0041                   'gui_Callback', []);
0042 if nargin && ischar(varargin{1})
0043     gui_State.gui_Callback = str2func(varargin{1});

```

```

0044     end
0045
0046     if nargin
0047         [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
0048     else
0049         gui_mainfcn(gui_State, varargin{:});
0050     end
0051     % End initialization code - DO NOT EDIT
0052
0053
0054
0055     global filename pathname fileNum ImageID image traces lineW;
0056
0057
0058
0059     % --- Executes just before maskM is made visible.
0060     function maskM_OpeningFcn(hObject, eventdata, handles, varargin)
0061     % This function has no output args, see OutputFcn.
0062     % hObject    handle to figure
0063     % eventdata  reserved - to be defined in a future version of MATLAB
0064     % handles    structure with handles and user data (see GUIDATA)
0065     % varargin   command line arguments to maskM (see VARARGIN)
0066
0067     % Choose default command line output for maskM
0068     handles.output = hObject;
0069
0070     % Update handles structure
0071     guidata(hObject, handles);
0072
0073
0074
0075     % UIWAIT makes maskM wait for user response (see UIRESUME)
0076     % uiwait(handles.figure1);
0077
0078
0079     % --- Outputs from this function are returned to the command line.
0080     function varargout = maskM_OutputFcn(hObject, eventdata, handles)
0081     % varargout  cell array for returning output args (see VARARGOUT);
0082     % hObject    handle to figure
0083     % eventdata  reserved - to be defined in a future version of MATLAB
0084     % handles    structure with handles and user data (see GUIDATA)
0085
0086     % Get default command line output from handles structure
0087     varargout{1} = handles.output;
0088
0089
0090
0091     % -----
0092     function FileMenu_Callback(hObject, eventdata, handles)
0093     % hObject    handle to FileMenu (see GCBO)
0094     % eventdata  reserved - to be defined in a future version of MATLAB
0095     % handles    structure with handles and user data (see GUIDATA)
0096
0097
0098     % -----
0099     function OpenMenuItem_Callback(hObject, eventdata, handles)

```

```

0100 % hObject handle to OpenMenuItem (see GCBO)
0101 % eventdata reserved - to be defined in a future version of MATLAB
0102 % handles structure with handles and user data (see GUIDATA)
0103
0104 % This function collects filenames that need to be looked at.
0105 % The function will read and display data of the first image also.
0106
0107 global filename CurrentP pathname fileNum ImageID image traces lineW traceFile
0108
0109 % lineW is the variable of line width that will be used in figures.
0110 lineW=2;
0111
0112 [filename, pathname, filterindex]=uigetfile('*.txt', 'pick a file', 'Multiselect', 'on');
0113
0114 CurrentP=pwd;
0115 path(path,CurrentP);
0116 cd(pathname);
0117
0118 if iscell(filename)
0119     fileNum=size(filename, 2);
0120 else
0121     fileNum=1;
0122 end
0123 if fileNum==1
0124     traceFile=filename;
0125 else
0126     traceFile=char(filename(1));
0127 end
0128
0129 ImageID=1;
0130
0131 % Read data of the first image.
0132
0133 traces=readtr(traceFile);
0134 NameLength=size(traceFile, 2)-6;
0135 imageFile=traceFile(1:NameLength);
0136 image=readimage(imageFile);
0137 cla
0138 pcolor(image);
0139 shading flat
0140 title(gca, imageFile);
0141 N=size(traces.tr, 1);
0142
0143 traceP=zeros(2,2);
0144 k=1;
0145
0146 % Display traces of the first image.
0147
0148 for j=2:N
0149     if traces.tr(j,1)==-1
0150         if traces.tr(j,2)==0
0151             hold on
0152             plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', lineW);
0153             traceP=zeros(2,2);
0154             k=1;
0155         end

```

```

0156     else
0157         traceP(k,:)=traces.tr(j,:);
0158         k=k+1;
0159     end
0160 end
0161
0162
0163
0164 % -----
0165 function PrintMenuItem_Callback(hObject, eventdata, handles)
0166 % hObject    handle to PrintMenuItem (see GCBO)
0167 % eventdata  reserved - to be defined in a future version of MATLAB
0168 % handles    structure with handles and user data (see GUIDATA)
0169 printdlg(handles.figure1)
0170
0171 % -----
0172 function CloseMenuItem_Callback(hObject, eventdata, handles)
0173 % hObject    handle to CloseMenuItem (see GCBO)
0174 % eventdata  reserved - to be defined in a future version of MATLAB
0175 % handles    structure with handles and user data (see GUIDATA)
0176
0177 % This function closes the program and all images.
0178
0179 global CurrentP;
0180 selection = questdlg(['Close ' get(handles.figure1,'Name') '?'],...
0181                    ['Close ' get(handles.figure1,'Name') '...'],...
0182                    'Yes','No','Yes');
0183 if strcmp(selection,'No')
0184     return;
0185 end
0186 cd(CurrentP);
0187 clear all;
0188 close all;
0189
0190
0191
0192 % --- Executes on selection change in popupmenu1.
0193 function popupmenu1_Callback(hObject, eventdata, handles)
0194 % hObject    handle to popupmenu1 (see GCBO)
0195 % eventdata  reserved - to be defined in a future version of MATLAB
0196 % handles    structure with handles and user data (see GUIDATA)
0197
0198 % Hints: contents = get(hObject,'String') returns popupmenu1 contents as cell array
0199 %       contents{get(hObject,'Value')} returns selected item from popupmenu1
0200
0201
0202 % --- Executes during object creation, after setting all properties.
0203 function popupmenu1_CreateFcn(hObject, eventdata, handles)
0204 % hObject    handle to popupmenu1 (see GCBO)
0205 % eventdata  reserved - to be defined in a future version of MATLAB
0206 % handles    empty - handles not created until after all CreateFcns called
0207
0208 % Hint: popupmenu controls usually have a white background on Windows.
0209 %       See ISPC and COMPUTER.
0210 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

```

```

0211     set(hObject,'BackgroundColor','white');
0212 end
0213
0214     set(hObject, 'String', {'plot(rand(5))', 'plot(sin(1:0.01:25))', 'bar(1:.5:10)',
'plot(membrane)', 'surf(peaks)'});
0215
0216
0217 % --- Executes on button press in pushbutton4.
0218 function pushbutton4_Callback(hObject, eventdata, handles)
0219 % hObject    handle to pushbutton4 (see GCBO)
0220 % eventdata  reserved - to be defined in a future version of MATLAB
0221 % handles    structure with handles and user data (see GUIDATA)
0222
0223
0224 % --- Executes on button press in pushbutton5.
0225 function pushbutton5_Callback(hObject, eventdata, handles)
0226 % hObject    handle to pushbutton5 (see GCBO)
0227 % eventdata  reserved - to be defined in a future version of MATLAB
0228 % handles    structure with handles and user data (see GUIDATA)
0229
0230
0231 % -----
0232 function Operation_Callback(hObject, eventdata, handles)
0233 % hObject    handle to Operation (see GCBO)
0234 % eventdata  reserved - to be defined in a future version of MATLAB
0235 % handles    structure with handles and user data (see GUIDATA)
0236
0237
0238 % -----
0239 function Delete_Callback(hObject, eventdata, handles)
0240 % hObject    handle to Delete (see GCBO)
0241 % eventdata  reserved - to be defined in a future version of MATLAB
0242 % handles    structure with handles and user data (see GUIDATA)
0243
0244 % This function allows user to select and delete traces from current images
0245 % displayed on screen.
0246
0247 global traces image lineW;
0248 input=round(ginput(1));
0249 inp(1)=input(2);
0250 inp(2)=input(1);
0251
0252 % FindTr will look at the traces and find out the trace been clicked by the
0253 % user.
0254
0255 tr=FindTr(inp);
0256
0257 % trace.N=0;
0258 % trace.N2=0;
0259 % trace.ID=0;
0260 % trace.start=zeros(1,2);
0261 % trace.end=zeros(1,2);
0262 if tr.N==0
0263     msgbox('Cannot find the trace.', 'Error', 'warn');
0264 else
0265

```

```

0266 % Display selected trace and ask user if the traces should be deleted or
0267 % not.
0268
0269     traceP=traces.tr(tr.N+1:tr.N2-1,:);
0270     hold on
0271     plot(traceP(:,2),traceP(:,1),'Color', 'yellow', 'LineWidth', 2);
0272
0273     button=questdlg('Delete this tracing?', 'Delete');
0274
0275 % Delete the trace.
0276
0277     if button(1)=='Y'
0278         tracesT=traces.tr;
0279         traces.tr=zeros(2,2);
0280         N=size(tracesT,1);
0281         add=1;
0282         j=1;
0283         for i=1:N
0284             if add==1 && tracesT(i,2)==tr.ID && tracesT(i,1)<0
0285                 add=0;
0286             end
0287             if add==0 && tracesT(i,2)==0 && tracesT(i,1)<0
0288                 add=1;
0289                 continue;
0290             end
0291
0292             if add==1
0293                 traces.tr(j,:)=tracesT(i,:);
0294                 j=j+1;
0295             end
0296         end
0297
0298 %     plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', 2);
0299
0300     cla
0301     pcolor(image);
0302     shading flat
0303
0304     traceP=zeros(2,2);
0305     k=1;
0306     N=size(traces.tr,1);
0307
0308     for j=2:N
0309         if traces.tr(j,1)==-1 || j==N
0310             hold on
0311             plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', lineW);
0312             traceP=zeros(2,2);
0313             k=1;
0314             continue;
0315         else
0316             traceP(k,:)=traces.tr(j,:);
0317             k=k+1;
0318         end
0319     end
0320 end
0321 end

```

```

0322
0323
0324
0325
0326
0327 % -----
0328 function Connect_Callback(hObject, eventdata, handles)
0329 % hObject handle to Connect (see GCBO)
0330 % eventdata reserved - to be defined in a future version of MATLAB
0331 % handles structure with handles and user data (see GUIDATA)
0332
0333 % This function allows user to select two traces and connect them together.
0334
0335 global traces image lineW
0336
0337 % trace.N=0;
0338 % trace.N2=0;
0339 % trace.ID=0;
0340 % trace.start=zeros(1,2);
0341 % trace.end=zeros(1,2);
0342
0343 select=zeros(1,2);
0344 tr1.N=0;
0345 tr1.N2=0;
0346 tr1.ID=0;
0347 tr1.start=zeros(1,2);
0348 tr1.end=zeros(1,2);
0349 tr2=tr1;
0350
0351 % User selects two traces that needs to be connected.
0352
0353 while sum(select)<2
0354     if select(1)==0
0355         msgbox('Select the first trace. ');
0356         pause
0357         input=ginput(1);
0358         inp(1)=input(2);
0359         inp(2)=input(1);
0360
0361 % FindTr is a function to look for the trace been clicked by user.
0362         tr1=FindTr(inp);
0363         if tr1.N==0
0364             msgbox('Cannot find the trace.', 'Error', 'warn');
0365             pause
0366         elseif tr2.ID==tr1.ID
0367             msgbox('Two traces cannot be the same.', 'Error', 'warn');
0368             pause
0369         else
0370             traceP=traces.tr(tr1.N+1:tr1.N2-1,:);
0371             hold on
0372             plot(traceP(:,2),traceP(:,1),'Color', 'yellow', 'LineWidth', 2);
0373             select(1)=1;
0374         end
0375     end
0376     if select(2)==0
0377         msgbox('Select the second trace. ');

```

```

0378     pause
0379     input=ginput(1);
0380     inp(1)=input(2);
0381     inp(2)=input(1);
0382     tr2=FindTr(inp);
0383     if tr2.N==0
0384         msgbox('Cannot find the trace.', 'Error', 'warn');
0385         pause
0386     elseif tr2.ID==tr1.ID
0387         msgbox('Two traces cannot be the same.', 'Error', 'warn');
0388         pause
0389     else
0390         traceP2=traces.tr(tr2.N+1:tr2.N2-1,:);
0391         hold on
0392         plot(traceP2(:,2),traceP2(:,1),'Color', 'yellow', 'LineWidth', 2);
0393         select(2)=1;
0394     end
0395 end
0396 end
0397
0398 % Look for the minimal range contains the two selected traces.
0399
0400 lower(1)=min(min(traceP(:,1)), min(traceP2(:,1)))-2;
0401 if lower(1)<0
0402     lower(1)=0;
0403 end
0404 lower(2)=min(min(traceP(:,2)), min(traceP2(:,2)))-2;
0405 if lower(2)<0
0406     lower(2)=0;
0407 end
0408 higher(1)=max(max(traceP(:,1)), max(traceP2(:,1)))+2;
0409 if higher(1)>size(image,1)
0410     higher(1)=size(image,1);
0411 end
0412 higher(2)=max(max(traceP(:,2)), max(traceP2(:,2)))+2;
0413 if higher(2)>size(image,2)
0414     higher(2)=size(image,2);
0415 end
0416
0417 % Zoom in to the minimal range contains two selected traces and label four
0418 % ends of two traces with different marker and color.
0419
0420 subimage=image(lower(1):higher(1), lower(2):higher(2));
0421
0422 fig1=figure;
0423 pcolor(subimage);
0424 shading flat
0425 tsStart1=tr1.start-lower+1;
0426 tsStart2=tr2.start-lower+1;
0427 tsEnd1=tr1.end-lower+1;
0428 tsEnd2=tr2.end-lower+1;
0429 hold on
0430 scatter(tsStart1(2),tsStart1(1), 'd','filled', 'MarkerEdgeColor','green', 'LineWidth', lineW);
0431 hold on
0432 scatter(tsEnd1(2),tsEnd1(1), 'o','filled', 'MarkerEdgeColor', 'cyan', 'LineWidth', lineW);
0433 hold on

```



```

0434 scatter(tsStart2(2), tsStart2(1), 's','filled', 'MarkerEdgeColor', 'magenta', 'LineWidth',
lineW);
0435 hold on
0436 scatter(tsEnd2(2), tsEnd2(1), '>', 'filled', 'MarkerEdgeColor', 'red', 'LineWidth', lineW);
0437
0438 op={'Diamond to square', 'Diamond to triangle', 'Circle to square', 'Circle to triangle'};
0439
0440 cnx=[0 0];
0441 cny=[0 0];
0442
0443 % Display a multiple choice dialog so that the user can choose the way of
0444 % connection.
0445 % The loop will be excuted until the user is satisfied with the connection.
0446
0447 while true
0448     [Selection, ok]=listdlg('PromptString', 'Select a connection', 'SelectionMode',
'single', 'ListString', op);
0449     if ok==0
0450         break;
0451     end
0452
0453     insert=zeros(2);
0454     insertN=1;
0455
0456 % Connection.
0457
0458     switch Selection
0459     case 1
0460         cnx(1)=tsStart1(2);
0461         cnx(2)=tsStart2(2);
0462         cny(1)=tsStart1(1);
0463         cny(2)=tsStart2(1);
0464         hold on
0465         plot(cnx, cny, 'Color', 'yellow', 'LineWidth', 2);
0466
0467         while true
0468             if cnx(1)==cnx(2) && cny(1)==cny(2)
0469                 break;
0470             end
0471
0472             disx=cnx(2)-cnx(1);
0473             disy=cny(2)-cny(1);
0474             if disx==0
0475                 if disy>0
0476                     cny(1)=cny(1)+1;
0477                 else
0478                     cny(1)=cny(1)-1;
0479                 end
0480             else
0481                 tanD=disy/disx;
0482                 if disy>0
0483                     if disx>0
0484                         if tanD<0.414
0485                             cnx(1)=cnx(1)+1;
0486                         elseif tanD>2.414
0487                             cny(1)=cny(1)+1;

```

```

0488         else
0489             cny(1)=cny(1)+1;
0490             cnx(1)=cnx(1)+1;
0491         end
0492     else
0493         if tanD>-0.414
0494             cnx(1)=cnx(1)-1;
0495         elseif tanD<-2.414
0496             cny(1)=cny(1)+1;
0497         else
0498             cnx(1)=cnx(1)-1;
0499             cny(1)=cny(1)+1;
0500         end
0501     end
0502 else %disy<0
0503     if disx>0
0504         if tanD>-0.414
0505             cnx(1)=cnx(1)+1;
0506         elseif tanD<-2.414
0507             cny(1)=cny(1)-1;
0508         else
0509             cny(1)=cny(1)-1;
0510             cnx(1)=cnx(1)+1;
0511         end
0512     else
0513         if tanD<0.414
0514             cnx(1)=cnx(1)-1;
0515         elseif tanD>2.414
0516             cny(1)=cny(1)-1;
0517         else
0518             cnx(1)=cnx(1)-1;
0519             cny(1)=cny(1)-1;
0520         end
0521     end
0522 end
0523 end
0524     insert(insertN,:)=cny(1)+lower(1)-1 cnx(1)+lower(2)-1];
0525     insertN=insertN+1;
0526 end
0527 traceAll=[flipud(traceP2); flipud(insert); traceP];
0528
0529 case 2
0530     cnx(1)=tsStart1(2);
0531     cnx(2)=tsEnd2(2);
0532     cny(1)=tsStart1(1);
0533     cny(2)=tsEnd2(1);
0534     hold on
0535     plot(cnx, cny, 'Color', 'yellow', 'LineWidth', 2);
0536     while true
0537         if cnx(1)==cnx(2) && cny(1)==cny(2)
0538             break;
0539         end
0540
0541     disx=cnx(2)-cnx(1);
0542     disy=cny(2)-cny(1);
0543     if disx==0

```

```

0544         if disy>0
0545             cny(1)=cny(1)+1;
0546         else
0547             cny(1)=cny(1)-1;
0548         end
0549     else
0550         tanD=disy/disx;
0551         if disy>0
0552             if disx>0
0553                 if tanD<0.414
0554                     cnx(1)=cnx(1)+1;
0555                 elseif tanD>2.414
0556                     cny(1)=cny(1)+1;
0557                 else
0558                     cny(1)=cny(1)+1;
0559                     cnx(1)=cnx(1)+1;
0560                 end
0561             else
0562                 if tanD>-0.414
0563                     cnx(1)=cnx(1)-1;
0564                 elseif tanD<-2.414
0565                     cny(1)=cny(1)+1;
0566                 else
0567                     cnx(1)=cnx(1)-1;
0568                     cny(1)=cny(1)+1;
0569                 end
0570             end
0571         else %disy<0
0572             if disx>0
0573                 if tanD>-0.414
0574                     cnx(1)=cnx(1)+1;
0575                 elseif tanD<-2.414
0576                     cny(1)=cny(1)-1;
0577                 else
0578                     cny(1)=cny(1)-1;
0579                     cnx(1)=cnx(1)+1;
0580                 end
0581             else
0582                 if tanD<0.414
0583                     cnx(1)=cnx(1)-1;
0584                 elseif tanD>2.414
0585                     cny(1)=cny(1)-1;
0586                 else
0587                     cnx(1)=cnx(1)-1;
0588                     cny(1)=cny(1)-1;
0589                 end
0590             end
0591         end
0592     end
0593     insert(insertN,:)=cny(1)+lower(1)-1 cnx(1)+lower(2)-1];
0594     insertN=insertN+1;
0595 end
0596 traceAll=[traceP2; flipud(insert); traceP];
0597 case 3
0598     cnx(1)=tsEnd1(2);
0599     cnx(2)=tsStart2(2);

```

```

0600     cny(1)=tsEnd1(1);
0601     cny(2)=tsStart2(1);
0602     hold on
0603     plot(cnx, cny, 'Color', 'yellow', 'LineWidth', 2);
0604     while true
0605         if cnx(1)==cnx(2) && cny(1)==cny(2)
0606             break;
0607         end
0608
0609         disx=cnx(2)-cnx(1);
0610         disy=cny(2)-cny(1);
0611         if disx==0
0612             if disy>0
0613                 cny(1)=cny(1)+1;
0614             else
0615                 cny(1)=cny(1)-1;
0616             end
0617         else
0618             tanD=disy/disx;
0619             if disy>0
0620                 if disx>0
0621                     if tanD<0.414
0622                         cnx(1)=cnx(1)+1;
0623                     elseif tanD>2.414
0624                         cny(1)=cny(1)+1;
0625                     else
0626                         cny(1)=cny(1)+1;
0627                         cnx(1)=cnx(1)+1;
0628                     end
0629                 else
0630                     if tanD>-0.414
0631                         cnx(1)=cnx(1)-1;
0632                     elseif tanD<-2.414
0633                         cny(1)=cny(1)+1;
0634                     else
0635                         cnx(1)=cnx(1)-1;
0636                         cny(1)=cny(1)+1;
0637                     end
0638                 end
0639             else %disy<0
0640                 if disx>0
0641                     if tanD>-0.414
0642                         cnx(1)=cnx(1)+1;
0643                     elseif tanD<-2.414
0644                         cny(1)=cny(1)-1;
0645                     else
0646                         cny(1)=cny(1)-1;
0647                         cnx(1)=cnx(1)+1;
0648                     end
0649                 else
0650                     if tanD<0.414
0651                         cnx(1)=cnx(1)-1;
0652                     elseif tanD>2.414
0653                         cny(1)=cny(1)-1;
0654                     else
0655                         cnx(1)=cnx(1)-1;

```

```

0656             cny(1)=cny(1)-1;
0657         end
0658     end
0659 end
0660 end
0661     insert(insertN,:)=cny(1)+lower(1)-1 cnx(1)+lower(2)-1];
0662     insertN=insertN+1;
0663 end
0664 traceAll=[traceP; insert; traceP2];
0665 otherwise
0666     cnx(1)=tsEnd1(2);
0667     cnx(2)=tsEnd2(2);
0668     cny(1)=tsEnd1(1);
0669     cny(2)=tsEnd2(1);
0670     hold on
0671     plot(cnx, cny, 'Color', 'yellow', 'LineWidth', 2);
0672     while true
0673         if cnx(1)==cnx(2) && cny(1)==cny(2)
0674             break;
0675         end
0676
0677         disx=cnx(2)-cnx(1);
0678         disy=cny(2)-cny(1);
0679         if disx==0
0680             if disy>0
0681                 cny(1)=cny(1)+1;
0682             else
0683                 cny(1)=cny(1)-1;
0684             end
0685         else
0686             tanD=disy/disx;
0687             if disy>0
0688                 if disx>0
0689                     if tanD<0.414
0690                         cnx(1)=cnx(1)+1;
0691                     elseif tanD>2.414
0692                         cny(1)=cny(1)+1;
0693                     else
0694                         cny(1)=cny(1)+1;
0695                         cnx(1)=cnx(1)+1;
0696                     end
0697                 else
0698                     if tanD>-0.414
0699                         cnx(1)=cnx(1)-1;
0700                     elseif tanD<-2.414
0701                         cny(1)=cny(1)+1;
0702                     else
0703                         cnx(1)=cnx(1)-1;
0704                         cny(1)=cny(1)+1;
0705                     end
0706                 end
0707             else %disy<0
0708                 if disx>0
0709                     if tanD>-0.414
0710                         cnx(1)=cnx(1)+1;
0711                     elseif tanD<-2.414

```

```

0712             cny(1)=cny(1)-1;
0713             else
0714                 cny(1)=cny(1)-1;
0715                 cnx(1)=cnx(1)+1;
0716             end
0717         else
0718             if tanD<0.414
0719                 cnx(1)=cnx(1)-1;
0720             elseif tanD>2.414
0721                 cny(1)=cny(1)-1;
0722             else
0723                 cnx(1)=cnx(1)-1;
0724                 cny(1)=cny(1)-1;
0725             end
0726         end
0727     end
0728     end
0729     insert(insertN,:)=cny(1)+lower(1)-1 cnx(1)+lower(2)-1];
0730     insertN=insertN+1;
0731 end
0732 traceAll=[traceP; insert; flipud(traceP2)];
0733 end
0734
0735 % Ask user if the connction is correct or not.
0736
0737     button= questdlg('Do you want to keep the connection?');
0738
0739     if button(1)=='Y'
0740         close(fig1)
0741         N=size(traces.tr,1);
0742         traceLabel=[-1 tr1.ID; traceAll; -1 0];
0743         minN=min(tr1.N, tr2.N);
0744         minN2=min(tr1.N2, tr2.N2);
0745         maxN=max(tr1.N, tr2.N);
0746         maxN2=max(tr1.N2, tr2.N2);
0747         traceT=[traces.tr(1:minN,:); traceLabel; traces.tr(minN2:maxN,:);
traces.tr(maxN2:N,:)];
0748         traces.tr=traceT;
0749
0750         cla
0751         pcolor(image);
0752         shading flat
0753         N=size(traces.tr, 1);
0754         traceP=zeros(2,2);
0755         k=1;
0756         for j=2:N
0757             if traces.tr(j,1)==-1 || j==N
0758                 hold on
0759                 plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', 1);
0760                 traceP=zeros(2,2);
0761                 k=1;
0762                 continue;
0763             else
0764                 traceP(k,:)=traces.tr(j,:);
0765                 k=k+1;
0766             end

```

```

0767         end
0768
0769         break;
0770     end
0771
0772     button= questdlg('Do you want to redo connection on same molecules?');
0773
0774     if button(1)=='N'
0775         close(fig1);
0776         break;
0777     end
0778     cla
0779     pcolor(subimage);
0780     shading flat
0781     hold on
0782     scatter(tsStart1(2),tsStart1(1), 'd','filled', 'MarkerEdgeColor','green', 'LineWidth', 1);
0783     hold on
0784     scatter(tsEnd1(2),tsEnd1(1), 'o','filled', 'MarkerEdgeColor', 'cyan', 'LineWidth', 1);
0785     hold on
0786     scatter(tsStart2(2), tsStart2(1), 's','filled', 'MarkerEdgeColor', 'magenta', 'LineWidth',
0787     1);
0787     hold on
0788     scatter(tsEnd2(2), tsEnd2(1), '>','filled', 'MarkerEdgeColor', 'red', 'LineWidth', 1);
0789
0790 end
0791
0792 %
0793 % %     plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', 2);
0794 %
0795 %     cla
0796 %     pcolor(image);
0797 %     shading flat
0798 %
0799
0800
0801
0802
0803 % -----
0804 function NextImage_Callback(hObject, eventdata, handles)
0805 % hObject    handle to NextImage (see GCBO)
0806 % eventdata reserved - to be defined in a future version of MATLAB
0807 % handles    structure with handles and user data (see GUIDATA)
0808
0809 % This function will ask the user to save the modified traces or not and
0810 % move on to the nexe image.
0811
0812 global filename fileNum ImageID image traces lineW traceFile
0813
0814 if iscell(filename)
0815     fileNum=size(filename, 2);
0816 else
0817     fileNum=1;
0818 end
0819
0820 if fileNum==1
0821     traceFile=filename;

```

```

0822     else
0823         traceFile=char(filename(ImageID));
0824     end
0825
0826     %save new tracing
0827     Button=questdlg('Save new tracing?');
0828
0829     if Button(1)=='Y'
0830         fd=fopen(traceFile, 'w');
0831         fprintf(fd, '%d %d \n\r', traces.BASE, traces.OVERWHELM);
0832         N=size(traces.tr,1);
0833         for i=1:N
0834             fprintf(fd, '%d %d \n\r', traces.tr(i,:));
0835         end
0836         fclose(fd);
0837     end
0838
0839     % Loading the next image. If current image is the last one, the program
0840     % will display a warning.
0841
0842     if ImageID >= fileNum
0843         msgbox(strcat(traceFile, ' is the last image. '), 'Last image', 'warn');
0844     else
0845         ImageID=ImageID+1;
0846         traceFile=char(filename(ImageID));
0847         traces=readtr(traceFile);
0848         NameLength=size(traceFile, 2)-6;
0849         imageFile=traceFile(1:NameLength);
0850         image=readimage(imageFile);
0851         cla
0852         pcolor(image);
0853         shading flat
0854         title(gca, imageFile);
0855
0856         N=size(traces.tr, 1);
0857
0858         traceP=zeros(2,2);
0859         k=1;
0860
0861         for j=2:N
0862             if traces.tr(j,1)==-1 || j==N
0863                 hold on
0864                 plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', lineW);
0865                 traceP=zeros(2,2);
0866                 k=1;
0867                 continue;
0868             else
0869                 traceP(k,:)=traces.tr(j,:);
0870                 k=k+1;
0871             end
0872         end
0873     end
0874
0875
0876     % Function of looking for traces been clicked by the user.
0877

```



```

0878 function trace = FindTr(Input)
0879 global traces;
0880 trace.N=0;
0881 trace.N2=0;
0882 trace.ID=0;
0883 trace.start=zeros(1,2);
0884 trace.end=zeros(1,2);
0885
0886 N=size(traces.tr,1);
0887 for i=1:N
0888     test=abs(traces.tr(i,:)-Input);
0889     if sum(test)<=5
0890         for j=i-1:1
0891             if traces.tr(j,1)== -1
0892                 trace.N=j;
0893                 trace.ID=traces.tr(j,2);
0894                 trace.start=traces.tr(j+1,:);
0895                 break;
0896             end
0897         end
0898         for j=i:N
0899             if traces.tr(j,1)== -1
0900                 trace.N2=j;
0901                 trace.end=traces.tr(j-1,:);
0902                 break;
0903             end
0904         end
0905     end
0906 end
0907
0908
0909 % -----
0910 function SegmentLength_Callback(hObject, eventdata, handles)
0911 % hObject   handle to SegmentLength (see GCBO)
0912 % eventdata reserved - to be defined in a future version of MATLAB
0913 % handles   structure with handles and user data (see GUIDATA)
0914
0915 % This function allows user to select a trace or a part of it and measure
0916 % the DNA length of selected segment.
0917
0918 global traces image traceFile lineW
0919
0920 % Display a zoom in figure to improve the convinence of trace selection.
0921
0922 hf=figure('Position',[100 200 900 700]);
0923
0924 lengthP=0;
0925 while true
0926     pcolor(image);
0927     shading flat
0928     title(gca, traceFile);
0929     hold on
0930     Closel='n';
0931     k=1;
0932     traceP=zeros(2);
0933     N=size(traces.tr,1);

```

```

0934     for j=2:N
0935         if traces.tr(j,1)==-1 || j==N
0936             hold on
0937             plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', lineW);
0938             traceP=zeros(2,2);
0939             k=1;
0940             continue;
0941         else
0942             traceP(k,:)=traces.tr(j,:);
0943             k=k+1;
0944         end
0945     end
0946
0947     input=round(ginput(1));
0948     inp(1)=input(2);
0949     inp(2)=input(1);
0950
0951     % Look for the selected trace.
0952
0953     tr=FindTr(inp);
0954
0955     % trace.N=0;
0956     % trace.N2=0;
0957     % trace.ID=0;
0958     % trace.start=zeros(1,2);
0959     % trace.end=zeros(1,2);
0960     if tr.N==0
0961         msgbox('Cannot find the trace.', 'Error', 'warn');
0962     else
0963         traceP=traces.tr(tr.N+1:tr.N2-1,:);
0964         hold on
0965         plot(traceP(:,2),traceP(:,1),'Color', 'yellow', 'LineWidth', 2);
0966
0967     % Calculate the segment length of selected trace. SegLength is a subfunction
0968     % calculate the length of a set of coordinates.
0969
0970         lengthC=SegLength(traceP);
0971         length=lengthC*1000/512;
0972         basepair=length/0.32;
0973
0974     % User can choose to select next trace or break current traces and continue
0975     % with a part of the segment.
0976
0977     while true
0978         qust=strcat('Segment length is:', num2str(length),'nm;_',
num2str(basepair),'basepairs. Measure next segment?');
0979         button=questdlg(qust, 'Measure', 'Break/Add it up', 'Next segment', 'Close
image', 'Close image');
0980         if button(1)=='C'
0981             Close1='y';
0982             break;
0983
0984         elseif button(1)=='B'
0985             button=questdlg('Break or Add it up?', 'Sub function', 'Break', 'Add it up',
'Break');
0986

```

```

0987 % Add in selection will keep current measurement into a buffer and display
0988 % it. The user can then select another segment and add it with the buffered
0989 % value.
0990
0991         if button(1)=='A'
0992             lengthP=lengthP+lengthC;
0993             lengthPn=lengthP*1000/512;
0994             qust=strcat('Current sum is:', num2str(lengthPn),'nm;_',
num2str(lengthPn/0.32),'basepairs. Clear it?');
0995             button=questdlg(qust, 'Sum', 'Yes', 'No', 'No');
0996             if button(1)=='Y'
0997                 lengthP=0;
0998                 button=questdlg('Go to next image?', 'End of sum', 'Yes', 'No', 'No');
0999                 if button(1)=='Y'
1000                     Close1='y';
1001                 end
1002             end
1003             break;
1004
1005 % Start to break current trace. The user will select the start and end
1006 % point of the segment which is interested.
1007
1008         else
1009             hold off
1010             pcolor(image);
1011             shading flat
1012 %             title(gca, traceFile);
1013             hold on
1014             plot(traceP(:,2),traceP(:,1),'Color', 'red', 'LineWidth', lineW);
1015             maxy=size(image,1);
1016             maxx=size(image,2);
1017             lowy=min(traceP(:,1))-10;
1018             lowx=min(traceP(:,2))-10;
1019             upy=max(traceP(:,1))+10;
1020             upx=max(traceP(:,2))+10;
1021             lowy=max([1 lowy]);
1022             lowx=max([1 lowx]);
1023             upy=min([maxy upy]);
1024             upx=min([maxx upx]);
1025
1026 % Zoom in to the segment needs to be break down.
1027
1028             set(gca, 'XLim', [lowx upx], 'YLim', [lowy upy]);
1029
1030             while true
1031
1032 % Look for the break point.
1033
1034                 breakp=round(ginput(2));
1035                 inb(:,1)=breakp(:,2);
1036                 inb(:,2)=breakp(:,1);
1037                 bro=1;
1038                 brop=[0 0];
1039                 test1=zeros(2,1);
1040                 test2=zeros(2,1);
1041                 for j=1:size(traceP,1)

```

```

1042         test1(j)=sum(abs(traceP(j,:)-inb(1,:)));
1043         test2(j)=sum(abs(traceP(j,:)-inb(2,:)));
1044     end
1045     [a brop(1)]=min(test1);
1046     [a brop(2)]=min(test2);
1047
1048     if brop(1)==brop(2)
1049         brop=[0 0];
1050         bro=0;
1051     end
1052     if bro==1
1053         break;
1054     else
1055         msgbox('Cannot find the break points.', 'Error', 'warn');
1056         pause
1057     end
1058 end
1059
1060 % Display the breaking result.
1061
1062         traceP2=traceP(min(brop):max(brop),:);
1063         lengthC=SegLength(traceP2);
1064         length=lengthC*1000/512;
1065         basepair=length/0.32;
1066         hold off
1067         pcolor(image);
1068         shading flat
1069         title(gca, traceFile);
1070         hold on
1071         plot(traceP2(:,2),traceP2(:,1),'Color', 'yellow', 'LineWidth', 2);
1072         set(gca, 'XLim', [lowx upx], 'YLim', [lowy upy]);
1073     end
1074     else
1075         break;
1076     end
1077 end
1078 hold off
1079 if Closel=='y'
1080     break;
1081 end
1082 end
1083 end
1084 close(hf);

```

### SegLength.m

```

001 % This function calculate the segment length of a set of coordinates.
002 % User can choose different method of calculating by active different part
003 % of codes.
004
005 % Authur: Haowei Wang (hwang23@emory.edu)
006 % Last updated Sep. 10th, 2011
007
008 function length = SegLength(traceP)
009
010 % Original algorithm

```

```

011
012     step=2;
013     Cstep=0;
014     prevP=traceP(1,:);
015     n=size(traceP,1);
016     length=0;
017     for j=2:n
018         if Cstep==step || j==n
019             length=length+sqrt((traceP(j,1)-prevP(1))^2+(traceP(j,2)-prevP(2))^2);
020             prevP=traceP(j,:);
021             Cstep=0;
022         else
023             Cstep=Cstep+1;
024         end
025     end
026
027     % End of original algorithm
028
029     % Freeman estimator
030
031     % prevP=traceP(1,:);
032     % n=size(traceP,1);
033     % length=0;
034     % for j=1:1:n
035     %     if sum(abs(traceP(j,:)-prevP))>1
036     %         length=length+1.414;
037     %     else
038     %         length=length+1;
039     %     end
040     %     prevP=traceP(j,:);
041     % end
042
043     % End of Freeman estimator
044
045     % MPO estimator
046
047     % prevP=traceP(1,:);
048     % n=size(traceP,1);
049     % ne=0;
050     % no=0;
051     % for j=1:1:n
052     %     if sum(abs(traceP(j,:)-prevP))>1
053     %         no=no+1;
054     %     else
055     %         ne=ne+1;
056     %     end
057     %     prevP=traceP(j,:);
058     % end
059     %
060     % length=sqrt((ne+no)^2+ne^2);
061
062     % End of MPO estimator
063
064     % Kulpa estimator
065
066     % prevP=traceP(1,:);

```

```

067 % n=size(traceP,1);
068 % ne=0;
069 % no=0;
070 % for j=1:1:n
071 %   if sum(abs(traceP(j,:)-prevP))>1
072 %     no=no+1;
073 %   else
074 %     ne=ne+1;
075 %   end
076 %
077 %   prevP=traceP(j,:);
078 % end
079 %
080 % length=0.948*ne+1.343*no;
081
082 % End of Kulpa estimator
083
084 % Corner chain estimator
085
086 % prevP=traceP(1,:);
087 % n=size(traceP,1);
088 % ne=0;
089 % no=0;
090 % nc=0;
091 % prev=0;
092 % for j=1:1:n
093 %   if sum(abs(traceP(j,:)-prevP))>1
094 %     % id is 1
095 %     no=no+1;
096 %     if prev==2
097 %       nc=nc+1;
098 %     end
099 %     prev=1;
100 %   else
101 %     % id is 2
102 %     ne=ne+1;
103 %     if prev==1
104 %       nc=nc+1;
105 %     end
106 %     prev=2;
107 %   end
108 %   prevP=traceP(j,:);
109 % end
110 %
111 % length=0.98*ne+1.406*no-0.091*nc;
112
113 % End of Corner chain estimator

```

### **lengthC.m**

```

001
002 % This program calculate the total contour length of each tracing.
003 % The final result will be saved into an array named final.
004 % The algorithm is defined in subfunction "SegLength.m". Please keep it in
005 % the same folder of the program.
006 % The subfunction of "readtr.m" is also needed by this program.

```

```

007 % Users can test different algorithm by change "SegLength.m"
008 % The upper and lower threshold are used to eliminate broken DNA and bad
009 % tracing.
010 % An overview of all tracing and images obtained by tracing program are
011 % suggested. It will be great helpful to delete bad images and tracing file
012 % before using this program.
013 % User can make change on image parameters below in the code.
014
015 % Authur: Haowei Wang (hwang23@emory.edu)
016 % Last updated Sep. 10th, 2011
017
018 clear all
019 close all
020
021 % Change parameters here: xyScale is the real length of one side of a
022 % square image in nanometer. imagesize is number of pixels of one line/row
023 % of the images.
024
025 xyScale=1000;
026 imagesize=512;
027
028 % End of changing parameters.
029
030 % Asking user to decide the threshold of segment selection. DNA traces
031 % longer than maximum (decided by upper bound) or shorter than minimum
032 % (decided by lower bound) will not be considered.
033
034 def={num2str(160),num2str(0.8), num2str(2.0)};
035 prompt={'Expected DNA length (nm):', 'Lower bound:', 'Upper bound'};
036 answer=inputdlg(prompt, 'Change parameter', 1, def);
037 eLength=str2double(char(answer(1)));
038 cutoff=str2double(char(answer(2)));
039 upperB=str2double(char(answer(3)));
040
041 [filename, pathname, filterindex]=uigetfile('*.txt', 'pick a file', 'Multiselect', 'on');
042
043 CurrentP=pwd;
044 path(path,CurrentP);
045 cd(pathname);
046
047 if iscell(filename)
048     fileNum=size(filename, 2);
049 else
050     fileNum=1;
051 end
052
053 fn=1;
054 length=zeros(2,1);
055
056 % This loop looks at selected images one after another.
057
058 for i=1:fileNum
059     if fileNum==1
060         traceFile=filename;
061     else
062         traceFile=char(filename(i));

```

```

063     end
064     traces=readtr(traceFile);
065     traceFile
066
067     prevP=[0 0];
068     k=1;
069     n=size(traces.tr,1);
070     traceP=zeros(2);
071
072     % This loop goes through all traces inside current image and calculate the
073     % length.
074     % Calculated length will be put into an array named final1.
075
076     for j=2:n
077         if traces.tr(j,1)==-1
078             if traces.tr(j,2)>0
079                 traceP=zeros(2);
080                 k=1;
081             else
082                 length(fn)=SegLength(traceP);
083                 fn=fn+1;
084             end
085         else
086             traceP(k,:)=traces.tr(j,:);
087             k=k+1;
088         end
089     end
090
091     % Traces longer or shorter than thresholds will be disgarded.
092
093     final1=length(length>eLength*imagesize/xyScale*cutoff);
094     final1=final1(final1<eLength*imagesize/xyScale*upperB);
095
096     end
097
098     % Convert length into nanometers.
099
100     pr_V=final*xyScale/imagesize;
101
102     % Prepare histograms.
103
104     lower=min(pr_V);
105     upper=max(pr_V);
106     % binS=2;
107     eStd=std(pr_V);
108
109     binS=eStd/15;
110     aLim=size(pr_V,1)/20;
111
112     xout=lower:binS:upper*1.2;
113     n=histc(pr_V,xout);
114     [nmax ni]=max(n);
115     bar(xout,n)
116     hold on
117
118     % Fitting with Gaussian curve and display.

```



```

119
120 cfun=fit(xout',n,'gauss1','Lower',[0 0 0],'Upper',[nmax 2*ni*binS+lower upper-lower])
121
122 xout2=lower:1:upper*1.2;
123 f2=feval(cfun,xout2');
124 plot(xout2,(f2),'r-','Linewidth',2);
125
126
127 cd(pwd);

```

### ParticleAnalysis.m

```

001 % This program look for particles bound or unbound by DNA molecules and
002 % calculates volume, crosssection and height of every molecules.
003
004 % This program need these subfunctions: addit.m, readtr.m, readimage.m,
005 % findtr.m.
006 % Please keep them in the same folder of the program.
007
008 % Authur: Haowei Wang (hwang23@emory.edu)
009 % Last updated Sep. 10th, 2011
010
011 function varargout = ParticleAnalysis(varargin)
012 %PARTICLEANALYSIS M-file for ParticleAnalysis.fig
013 %   PARTICLEANALYSIS, by itself, creates a new PARTICLEANALYSIS or raises the
existing
014 %   singleton*.
015 %
016 %   H = PARTICLEANALYSIS returns the handle to a new PARTICLEANALYSIS or
the handle to
017 %   the existing singleton*.
018 %
019 %   PARTICLEANALYSIS('Property','Value',...) creates a new PARTICLEANALYSIS
using the
020 %   given property value pairs. Unrecognized properties are passed via
021 %   varargin to ParticleAnalysis_OpeningFcn. This calling syntax produces a
022 %   warning when there is an existing singleton*.
023 %
024 %   PARTICLEANALYSIS('CALLBACK') and
PARTICLEANALYSIS('CALLBACK',hObject,...) call the
025 %   local function named CALLBACK in PARTICLEANALYSIS.M with the given input
026 %   arguments.
027 %
028 %   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
029 %   instance to run (singleton)".
030 %
031 % See also: GUIDE, GUIDATA, GUIHANDLES
032
033 % Edit the above text to modify the response to help ParticleAnalysis
034
035 % Last Modified by GUIDE v2.5 02-Sep-2011 09:48:10
036
037 % Begin initialization code - DO NOT EDIT
038 gui_Singleton = 1;
039 gui_State = struct('gui_Name',    mfilename, ...
040                  'gui_Singleton', gui_Singleton, ...

```

```

041         'gui_OpeningFcn', @ParticleAnalysis_OpeningFcn, ...
042         'gui_OutputFcn', @ParticleAnalysis_OutputFcn, ...
043         'gui_LayoutFcn', [], ...
044         'gui_Callback', []);
045     if nargin && ischar(varargin{1})
046         gui_State.gui_Callback = str2func(varargin{1});
047     end
048
049     if nargout
050         [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
051     else
052         gui_mainfcn(gui_State, varargin{:});
053     end
054     % End initialization code - DO NOT EDIT
055
056
057     % --- Executes just before ParticleAnalysis is made visible.
058     function ParticleAnalysis_OpeningFcn(hObject, eventdata, handles, varargin)
059     % This function has no output args, see OutputFcn.
060     % hObject    handle to figure
061     % eventdata  reserved - to be defined in a future version of MATLAB
062     % handles    structure with handles and user data (see GUIDATA)
063     % varargin   unrecognized PropertyName/PropertyValue pairs from the
064     %            command line (see VARARGIN)
065
066     % Choose default command line output for ParticleAnalysis
067     handles.output = hObject;
068
069     % Update handles structure
070     guidata(hObject, handles);
071
072     % UIWAIT makes ParticleAnalysis wait for user response (see UIRESUME)
073     % uiwait(handles.figure1);
074
075
076     % --- Outputs from this function are returned to the command line.
077     function varargout = ParticleAnalysis_OutputFcn(hObject, eventdata, handles)
078     % varargout  cell array for returning output args (see VARARGOUT);
079     % hObject    handle to figure
080     % eventdata  reserved - to be defined in a future version of MATLAB
081     % handles    structure with handles and user data (see GUIDATA)
082
083     % Get default command line output from handles structure
084     varargout{1} = handles.output;
085
086
087     % --- Executes on button press in calculate.
088     function calculate_Callback(hObject, eventdata, handles)
089     % hObject    handle to calculate (see GCBO)
090     % eventdata  reserved - to be defined in a future version of MATLAB
091     % handles    structure with handles and user data (see GUIDATA)
092
093     % This function select files that need to be analyzed.
094     % If the file type is set to 'Free particles', the program will open AFM
095     % data directly; otherwise, it will open tracing files instead.
096

```

```

097 global analyT FileType filename pathname
098
099 FileType=analyT;
100
101 if FileType(1)=='F'
102     str='*. *';
103 else
104     str='*.txt';
105 end
106
107 [filename, pathname, filterindex]=uigetfile(str, 'pick a file', 'Multiselect', 'on');
108
109 CurrentP=pwd;
110 path(path,CurrentP);
111 cd(pathname);
112
113
114 % --- Executes on button press in reset.
115 function reset_Callback(hObject, eventdata, handles)
116 % hObject    handle to reset (see GCBO)
117 % eventdata  reserved - to be defined in a future version of MATLAB
118 % handles    structure with handles and user data (see GUIDATA)
119
120 % This function will look at particles through images and do the
121 % calculation.
122
123 global imageP analyP FileType filename data biColor
124
125 if iscell(filename)
126     fileNum=size(filename, 2);
127 else
128     fileNum=1;
129 end
130
131 final=zeros(3);
132 fn=1;
133
134 % THRESHOLD=1.0; % Set the threshold for elimination of DNA.
135 % AreaSize=10; % Blobs contain less than this number of pixels will not be counted.
136 % imageSize=512;
137
138 zScale=imageP.zScale/imageP.zRange;
139 xyScale=imageP.xyScale/imageP.ImageSize;
140
141 for i=1:fileNum
142     if fileNum==1
143         traceFile=filename;
144     else
145         traceFile=char(filename(i));
146     end
147
148 % Display the filename of current image.
149
150     traceFile
151
152 % For bound particle analysis, the program will read both traces and image

```

```

153 % data; otherwise, only images data are read.
154
155     if FileType(1)=='B'
156
157 % Read traces.
158
159         traces=readtr(traceFile);
160         NameLength=size(traceFile, 2)-6;
161
162 % Reconstruct image filename
163
164         imageFile=traceFile(1:NameLength);
165     else
166         NameLength=size(traceFile, 2);
167         if traceFile(NameLength-2)~='0'
168             continue;
169         end
170         imageFile=traceFile;
171     end
172
173 % Read image data
174
175     image=readimage(imageFile);
176     ImageSize=size(image,1);
177
178 % Set background level for analysis. For bound particle, the background
179 % level is read out from tracing file. For unbound particle analysis, the
180 % background level is calculated by averaging.
181
182     if FileType(1)=='B'
183         basel=traces.BASE;
184     else
185         basel=mean(mean(image));
186     end
187
188     image=image-basel;
189
190 % Use threshold to cut particle pixels from the image.
191
192     biColor=image;
193     biColor(biColor<analyP.threshold/zScale)=0;
194     biColor(biColor>0)=1;
195     biColor(1,:)=0;
196     biColor(ImageSize,:)=0;
197     biColor(:,1)=0;
198     biColor(:,ImageSize)=0;
199
200     blobs=zeros(2);
201     blobsI=1;
202     bID=1;
203     blobsT=zeros(2);
204
205 % This part looks for the area of each blobs
206
207     for m=1:ImageSize
208         for n=1:ImageSize

```

```

209         if biColor(m,n)==1
210
211         % The addit function will start from one point and look for all points
212         % belong to the same particle and put coordinates into one array.
213
214         blobsT=addit(m,n);
215
216         % Particles smaller than the minimum value will be disregarded.
217
218         if size(blobsT,1)>analyP.MinSize*xyScale^2
219             blobs(blobsI,:)=[-1 bID];
220             lowerx=min(blobsT(:,1));
221             lowery=min(blobsT(:,2));
222             upperx=max(blobsT(:,1));
223             uppery=max(blobsT(:,2));
224             blobs(blobsI+1,:)=[lowerx lowery];
225             blobs(blobsI+2,:)=[upperx uppery];
226             blobs(blobsI+3,:)=[-1 0];
227             blobsI=blobsI+3;
228             nextI=blobsI+size(blobsT,1);
229             blobs(blobsI+1:nextI,:)=blobsT;
230             blobsI=nextI+1;
231             bID=bID+1;
232             blobsT=zeros(2);
233         end
234     end
235 end
236 end
237
238 % Display particles found by program.
239
240 if i==1
241     figure('Position',[10 500 400 300]);
242     pcolor(image)
243     shading flat
244     hold on
245     plot(blobs(:,2), blobs(:,1), 'LineStyle', 'none', 'Color', 'red', 'Marker', '.',
'MarkerSize',2);
246     figure('Position',[10 50 400 300]);
247     pcolor(image)
248     shading flat
249 end
250
251
252 % The maskB matrix saved all found particles. The value of every pixels is
253 % set to the particle ID instead of real heigth.
254
255 maskB=zeros(ImageSize);
256 for j=1:size(blobs,1)
257     if blobs(j,1)==-1
258         if blobs(j,2)>0
259             bID=blobs(j,2);
260             blobs(j+1,1)=0;
261             blobs(j+2,1)=0;
262         end
263     else

```

```

264         if blobs(j,1)>0
265             maskB(blobs(j,1), blobs(j,2))=bID;
266         end
267     end
268 end
269
270 trID=0;
271 trN=1;
272
273 % For analyzing bound particles, only particles overlap with one or more
274 % DNA traces will be find out and considered.
275
276 if FileType(1)=='B'
277     maskT=maskB;
278     for j=1:size(traces.tr,1)
279         if traces.tr(j,1)>0
280             if maskT(traces.tr(j,1), traces.tr(j,2))>0
281                 trID(trN)=maskT(traces.tr(j,1), traces.tr(j,2));
282                 trN=trN+1;
283                 maskT(maskT==trID(trN-1))=0;
284             end
285         end
286     end
287 else
288     maxID=max(max(maskB));
289     for j=1:maxID
290         maskT=maskB;
291         maskT(maskT~=j)=0;
292         if sum(sum(maskT))==0
293             continue;
294         end
295         trID(trN)=j;
296         trN=trN+1;
297     end
298 end
299
300 maskT=maskB;
301
302 for j=1:size(trID,2)
303     maskT=maskB;
304     IM=image;
305     tID=trID(j);
306     maskT(maskT~=tID)=0;
307     maskT=maskT/tID;
308     IM=IM.*maskT;
309     Bheight=max(max(IM));
310
311 % Eliminate blobs too high or too low
312 if Bheight<analyP.MinHeight/zScale
313     continue;
314 end
315
316 if Bheight>analyP.MaxHeight/zScale
317     continue;
318 end
319

```

```

320 % Eliminate blobs too big or too small
321
322     if sum(sum(maskT))>analyP.MaxSize*xyScale^2
323         continue;
324     end
325
326     if sum(sum(maskT))<analyP.MinSize*xyScale^2
327         continue;
328     end
329
330
331 % Blobs too close to the edge will not be considered in.
332
333     si=size(IM,1);
334     xEdge1=IM(1:analyP.margin,:);
335     xEdge2=IM(si-analyP.margin+1:si,:);
336     yEdge1=IM(:,1:analyP.margin);
337     yEdge2=IM(:, si-analyP.margin+1:si);
338     sumE=sum(sum(xEdge1+xEdge2))+sum(sum(yEdge1+yEdge2));
339     if sumE>0
340         continue;
341     end
342
343 % Calculate volume diameter;
344 % IM is the deduced image contains only one particle.
345 % Bheight is the maximum height of the particle.
346
347     volume=sum(sum(IM));
348
349     if analyP.CrossSectionP>0.1 && analyP.CrossSectionP<100
350         CrossSection=Bheight*analyP.CrossSectionP/100;
351     else
352         CrossSection=analyP.CrossSection/zScale;
353     end
354
355     IM(IM<CrossSection)=0;
356     IM(IM>0)=1;
357     CroSec=sum(sum(IM));
358     dia=2*sqrt(CroSec/3.14);
359     dia=dia/xyScale;
360     final(fn,1)=dia;
361     final(fn,2)=Bheight*zScale;
362     final(fn,3)=volume*zScale/xyScale/xyScale;
363     fn=fn+1;
364 end
365 end
366
367 data=final;
368
369 % Display figure of diameter distribution.
370
371 figure('Position',[400 500 400 300]);
372
373 pr=final(:,1);
374 pr_V=pr(pr>0);
375 xout=0:0.5:max(pr_V)*1.2;

```

```

376     n=histc(pr_V,xout)/size(pr_V,1);
377     bar(xout,n)
378     title('Blobs Diameter');
379     xlabel('Blobs Diameter (nm)');
380     ylabel('Percentage');
381
382     % Display figure of height distribution.
383
384     figure('Position',[400 50 400 300]);
385     pr=final(:,2);
386     pr_V=pr(pr>0);
387     xout=0:0.5:max(pr_V)*1.2;
388     n=histc(pr_V,xout)/size(pr_V,1);
389     bar(xout,n)
390     title('Blobs Height');
391     xlabel('Blobs Height (nm)');
392     ylabel('Percentage');
393
394     % Display figure of volume distribution.
395
396     figure('Position',[800 50 400 300]);
397
398     pr=final(:,3);
399     pr_V=pr(pr>0);
400     xout=0:10:max(pr_V)*1.2;
401     n=histc(pr_V,xout)/size(pr_V,1);
402     bar(xout,n)
403     title('Blobs Volume');
404     xlabel('Blobs Volume (nm^3)');
405     ylabel('Percentage');
406
407
408     % imageP.ImageSize
409     % imageP.xyScale
410     % imageP.zRange
411     % imageP.zScale
412     %
413     % analyP.threshold
414     % analyP.CrossSection
415     % analyP.MaxHeight
416     % analyP.MinHeight
417     % analyP.MaxSize
418     % analyP.MinSize
419     % analyP.margin
420
421     imageP
422     analyP
423
424
425
426     % --- Executes when selected object is changed in unitgroup.
427     function unitgroup_SelectionChangeFcn(hObject, eventdata, handles)
428     % hObject    handle to the selected object in unitgroup
429     % eventdata  structure with the following fields (see UIBUTTONGROUP)
430     %     EventName: string 'SelectionChanged' (read only)
431     %     OldValue: handle of the previously selected object or empty if none was selected

```



```

432 %     NewValue: handle of the currently selected object
433 % handles  structure with handles and user data (see GUIDATA)
434
435 % This function allows user to choose analyze free particles or particles
436 % bound by DNA traces.
437 % AnalyT is 'Free' or 'Bound'.
438
439 global analyT
440 analyT=get(hObject, 'string');
441
442
443 % --- Executes on button press in pushbutton9.
444 function pushbutton9_Callback(hObject, eventdata, handles)
445 % hObject    handle to pushbutton9 (see GCBO)
446 % eventdata  reserved - to be defined in a future version of MATLAB
447 % handles    structure with handles and user data (see GUIDATA)
448
449 % This function will save the analyzing data into excel files.
450
451 global filename data pathname imageP analyP FileType
452
453 % Generating filename for excel.
454
455 if iscell(filename)
456     fileNum=size(filename, 2);
457 else
458     fileNum=1;
459 end
460
461 if fileNum==1
462     File=filename;
463 else
464     File=char(filename(1));
465 end
466
467 for i=1:size(File,2)
468     if File(i)=='.'
469         break;
470     end
471 end
472
473 putfile=[File(1:i-1), '.xls'];
474 cd(pathname);
475
476 [filep, pathp, filterindex]=uiputfile(putfile, 'Save data');
477
478 % Saving file head.
479
480 wdata={'Diameter (nm)', 'Height (nm)', 'Volume (nm^3)', FileType};
481 xlswrite(filep, wdata, 1);
482
483 % Saving data
484
485 xlswrite(filep, data, 1, 'A2');
486
487 % Saving parameters.

```

```

488
489     d={'Image Size', imageP.ImageSize; 'X-Y Scale', imageP.xyScale; 'Z Range',
imageP.zRange; 'Z Scale', imageP.zScale; 'Threshold', analyP.threshold; 'Cross Section',
analyP.CrossSection; 'Cross Section %', analyP.CrossSectionP;
490     'Max Height', analyP.MaxHeight; 'Min Height', analyP.MinHeight; 'Max Size',
analyP.MaxSize; 'Min Size', analyP.MinSize; 'Margin', analyP.margin};
491
492     xlswrite(filep, d, 2, 'C1');
493
494
495     % Functions below are used to set all parameters used in analysis. More
496     % information about the parameters can be found in the manual.
497
498     function density_Callback(hObject, eventdata, handles)
499     % hObject    handle to density (see GCBO)
500     % eventdata  reserved - to be defined in a future version of MATLAB
501     % handles    structure with handles and user data (see GUIDATA)
502     global imageP
503     str=get(hObject, 'string');
504     imageP.ImageSize=str2double(str);
505
506     % Hints: get(hObject,'String') returns contents of density as text
507     %        str2double(get(hObject,'String')) returns contents of density as a double
508
509
510     % --- Executes during object creation, after setting all properties.
511     function density_CreateFcn(hObject, eventdata, handles)
512     % hObject    handle to density (see GCBO)
513     % eventdata  reserved - to be defined in a future version of MATLAB
514     % handles    empty - handles not created until after all CreateFcns called
515     global imageP
516     % Hint: edit controls usually have a white background on Windows.
517     %        See ISPC and COMPUTER.
518     if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
519         set(hObject,'BackgroundColor','white');
520     end
521
522     str=get(hObject, 'string');
523     imageP.ImageSize=str2double(str);
524
525
526
527     function edit7_Callback(hObject, eventdata, handles)
528     % hObject    handle to edit7 (see GCBO)
529     % eventdata  reserved - to be defined in a future version of MATLAB
530     % handles    structure with handles and user data (see GUIDATA)
531     global imageP
532
533     str=get(hObject, 'string');
534     imageP.xyScale=str2double(str);
535
536     % Hints: get(hObject,'String') returns contents of edit7 as text
537     %        str2double(get(hObject,'String')) returns contents of edit7 as a double
538
539

```

```

540 % --- Executes during object creation, after setting all properties.
541 function edit7_CreateFcn(hObject, eventdata, handles)
542 % hObject    handle to edit7 (see GCBO)
543 % eventdata  reserved - to be defined in a future version of MATLAB
544 % handles    empty - handles not created until after all CreateFcns called
545 global imageP
546 % Hint: edit controls usually have a white background on Windows.
547 %    See ISPC and COMPUTER.
548 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
549     set(hObject,'BackgroundColor','white');
550 end
551 str=get(hObject, 'string');
552 imageP.xyScale=str2double(str);
553
554
555 function edit8_Callback(hObject, eventdata, handles)
556 % hObject    handle to edit8 (see GCBO)
557 % eventdata  reserved - to be defined in a future version of MATLAB
558 % handles    structure with handles and user data (see GUIDATA)
559 global imageP
560
561 str=get(hObject, 'string');
562 imageP.zRange=str2double(str);
563
564 % Hints: get(hObject,'String') returns contents of edit8 as text
565 %    str2double(get(hObject,'String')) returns contents of edit8 as a double
566
567
568 % --- Executes during object creation, after setting all properties.
569 function edit8_CreateFcn(hObject, eventdata, handles)
570 % hObject    handle to edit8 (see GCBO)
571 % eventdata  reserved - to be defined in a future version of MATLAB
572 % handles    empty - handles not created until after all CreateFcns called
573 global imageP
574 % Hint: edit controls usually have a white background on Windows.
575 %    See ISPC and COMPUTER.
576 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
577     set(hObject,'BackgroundColor','white');
578 end
579 str=get(hObject, 'string');
580 imageP.zRange=str2double(str);
581
582
583 function edit9_Callback(hObject, eventdata, handles)
584 % hObject    handle to edit9 (see GCBO)
585 % eventdata  reserved - to be defined in a future version of MATLAB
586 % handles    structure with handles and user data (see GUIDATA)
587 global imageP
588
589
590 str=get(hObject, 'string');
591 imageP.zScale=str2double(str);
592
593 % Hints: get(hObject,'String') returns contents of edit9 as text

```

```

594 %     str2double(get(hObject,'String')) returns contents of edit9 as a double
595
596
597 % --- Executes during object creation, after setting all properties.
598 function edit9_CreateFcn(hObject, eventdata, handles)
599 % hObject    handle to edit9 (see GCBO)
600 % eventdata  reserved - to be defined in a future version of MATLAB
601 % handles    empty - handles not created until after all CreateFcns called
602
603 global imageP
604
605 % Hint: edit controls usually have a white background on Windows.
606 %     See ISPC and COMPUTER.
607 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
608     set(hObject,'BackgroundColor','white');
609 end
610
611 str=get(hObject, 'string');
612 imageP.zScale=str2double(str);
613
614
615 function edit10_Callback(hObject, eventdata, handles)
616 % hObject    handle to edit10 (see GCBO)
617 % eventdata  reserved - to be defined in a future version of MATLAB
618 % handles    structure with handles and user data (see GUIDATA)
619 global analyP
620
621
622 str=get(hObject, 'string');
623 analyP.threshold=str2double(str);
624
625 % Hints: get(hObject,'String') returns contents of edit10 as text
626 %     str2double(get(hObject,'String')) returns contents of edit10 as a double
627
628
629 % --- Executes during object creation, after setting all properties.
630 function edit10_CreateFcn(hObject, eventdata, handles)
631 % hObject    handle to edit10 (see GCBO)
632 % eventdata  reserved - to be defined in a future version of MATLAB
633 % handles    empty - handles not created until after all CreateFcns called
634 global analyP
635 % Hint: edit controls usually have a white background on Windows.
636 %     See ISPC and COMPUTER.
637 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
638     set(hObject,'BackgroundColor','white');
639 end
640
641 str=get(hObject, 'string');
642 analyP.threshold=str2double(str);
643
644
645 function edit11_Callback(hObject, eventdata, handles)
646 % hObject    handle to edit11 (see GCBO)
647 % eventdata  reserved - to be defined in a future version of MATLAB

```

```

648 % handles structure with handles and user data (see GUIDATA)
649 global analyP
650
651 str=get(hObject, 'string');
652 analyP.CrossSection=str2double(str);
653
654 % Hints: get(hObject,'String') returns contents of edit11 as text
655 % str2double(get(hObject,'String')) returns contents of edit11 as a double
656
657
658 % --- Executes during object creation, after setting all properties.
659 function edit11_CreateFcn(hObject, eventdata, handles)
660 % hObject handle to edit11 (see GCBO)
661 % eventdata reserved - to be defined in a future version of MATLAB
662 % handles empty - handles not created until after all CreateFcns called
663
664 global analyP
665 % Hint: edit controls usually have a white background on Windows.
666 % See ISPC and COMPUTER.
667 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
668 set(hObject,'BackgroundColor','white');
669 end
670
671 str=get(hObject, 'string');
672 analyP.CrossSection=str2double(str);
673
674
675 function edit12_Callback(hObject, eventdata, handles)
676 % hObject handle to edit12 (see GCBO)
677 % eventdata reserved - to be defined in a future version of MATLAB
678 % handles structure with handles and user data (see GUIDATA)
679 global analyP
680
681 str=get(hObject, 'string');
682 analyP.MaxHeight=str2double(str);
683 % Hints: get(hObject,'String') returns contents of edit12 as text
684 % str2double(get(hObject,'String')) returns contents of edit12 as a double
685
686
687 % --- Executes during object creation, after setting all properties.
688 function edit12_CreateFcn(hObject, eventdata, handles)
689 % hObject handle to edit12 (see GCBO)
690 % eventdata reserved - to be defined in a future version of MATLAB
691 % handles empty - handles not created until after all CreateFcns called
692 global analyP
693 % Hint: edit controls usually have a white background on Windows.
694 % See ISPC and COMPUTER.
695 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
696 set(hObject,'BackgroundColor','white');
697 end
698
699 str=get(hObject, 'string');
700 analyP.MaxHeight=str2double(str);
701

```

```

702
703
704 function edit13_Callback(hObject, eventdata, handles)
705 % hObject   handle to edit13 (see GCBO)
706 % eventdata reserved - to be defined in a future version of MATLAB
707 % handles   structure with handles and user data (see GUIDATA)
708 global analyP
709
710 str=get(hObject, 'string');
711 analyP.MinHeight=str2double(str);
712
713
714 % Hints: get(hObject,'String') returns contents of edit13 as text
715 %       str2double(get(hObject,'String')) returns contents of edit13 as a double
716
717
718 % --- Executes during object creation, after setting all properties.
719 function edit13_CreateFcn(hObject, eventdata, handles)
720 % hObject   handle to edit13 (see GCBO)
721 % eventdata reserved - to be defined in a future version of MATLAB
722 % handles   empty - handles not created until after all CreateFcns called
723 global analyP
724
725 % Hint: edit controls usually have a white background on Windows.
726 %       See ISPC and COMPUTER.
727 if ispc && isequal(get(hObject,'BackgroundColor'),
728 get(0,'defaultUicontrolBackgroundColor'))
729     set(hObject,'BackgroundColor','white');
730 end
731
732 str=get(hObject, 'string');
733 analyP.MinHeight=str2double(str);
734
735
736 function edit14_Callback(hObject, eventdata, handles)
737 % hObject   handle to edit14 (see GCBO)
738 % eventdata reserved - to be defined in a future version of MATLAB
739 % handles   structure with handles and user data (see GUIDATA)
740 global analyP
741
742 str=get(hObject, 'string');
743 analyP.MaxSize=str2double(str);
744
745 % Hints: get(hObject,'String') returns contents of edit14 as text
746 %       str2double(get(hObject,'String')) returns contents of edit14 as a double
747
748
749 % --- Executes during object creation, after setting all properties.
750 function edit14_CreateFcn(hObject, eventdata, handles)
751 % hObject   handle to edit14 (see GCBO)
752 % eventdata reserved - to be defined in a future version of MATLAB
753 % handles   empty - handles not created until after all CreateFcns called
754 global analyP
755
756 % Hint: edit controls usually have a white background on Windows.

```

```

757 % See ISPC and COMPUTER.
758 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
759     set(hObject,'BackgroundColor','white');
760 end
761
762 str=get(hObject, 'string');
763 analyP.MaxSize=str2double(str);
764
765
766
767
768 function edit15_Callback(hObject, eventdata, handles)
769 % hObject handle to edit15 (see GCBO)
770 % eventdata reserved - to be defined in a future version of MATLAB
771 % handles structure with handles and user data (see GUIDATA)
772 global analyP
773
774 str=get(hObject, 'string');
775 analyP.MinSize=str2double(str);
776
777 % Hints: get(hObject,'String') returns contents of edit15 as text
778 % str2double(get(hObject,'String')) returns contents of edit15 as a double
779
780
781 % --- Executes during object creation, after setting all properties.
782 function edit15_CreateFcn(hObject, eventdata, handles)
783 % hObject handle to edit15 (see GCBO)
784 % eventdata reserved - to be defined in a future version of MATLAB
785 % handles empty - handles not created until after all CreateFcns called
786 global analyP
787
788 % Hint: edit controls usually have a white background on Windows.
789 % See ISPC and COMPUTER.
790 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
791     set(hObject,'BackgroundColor','white');
792 end
793
794 str=get(hObject, 'string');
795 analyP.MinSize=str2double(str);
796
797
798
799
800 function edit16_Callback(hObject, eventdata, handles)
801 % hObject handle to edit16 (see GCBO)
802 % eventdata reserved - to be defined in a future version of MATLAB
803 % handles structure with handles and user data (see GUIDATA)
804 global analyP
805
806 str=get(hObject, 'string');
807 analyP.margin=str2double(str);
808
809
810 % Hints: get(hObject,'String') returns contents of edit16 as text

```

```

811 %     str2double(get(hObject,'String')) returns contents of edit16 as a double
812
813
814 % --- Executes during object creation, after setting all properties.
815 function edit16_CreateFcn(hObject, eventdata, handles)
816 % hObject    handle to edit16 (see GCBO)
817 % eventdata  reserved - to be defined in a future version of MATLAB
818 % handles    empty - handles not created until after all CreateFcns called
819
820 global analyP
821
822 % Hint: edit controls usually have a white background on Windows.
823 %     See ISPC and COMPUTER.
824 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
825     set(hObject,'BackgroundColor','white');
826 end
827
828 str=get(hObject, 'string');
829 analyP.margin=str2double(str);
830
831
832 % --- If Enable == 'on', executes on mouse press in 5 pixel border.
833 % --- Otherwise, executes on mouse press in 5 pixel border or over reset.
834 function reset_ButtonDownFcn(hObject, eventdata, handles)
835 % hObject    handle to reset (see GCBO)
836 % eventdata  reserved - to be defined in a future version of MATLAB
837 % handles    structure with handles and user data (see GUIDATA)
838
839 % Display analyzing parameters.
840
841 global imageP analyP
842
843 imageP.ImageSize
844 imageP.xyScale
845 imageP.zRange
846 imageP.zScale
847
848 analyP.threshold
849 analyP.CrossSection
850 analyP.MaxHeight
851 analyP.MinHeight
852 analyP.MaxSize
853 analyP.MinSize
854 analyP.margin
855
856
857 % --- Executes on button press in english.
858 function english_Callback(hObject, eventdata, handles)
859 % hObject    handle to english (see GCBO)
860 % eventdata  reserved - to be defined in a future version of MATLAB
861 % handles    structure with handles and user data (see GUIDATA)
862 global analyT
863 analyT=get(hObject, 'string');
864
865 % Hint: get(hObject,'Value') returns toggle state of english

```



```

866
867
868 % --- Executes during object creation, after setting all properties.
869 function unitgroup_CreateFcn(hObject, eventdata, handles)
870 % hObject    handle to unitgroup (see GCBO)
871 % eventdata  reserved - to be defined in a future version of MATLAB
872 % handles    empty - handles not created until after all CreateFcns called
873
874
875 % --- Executes during object creation, after setting all properties.
876 function english_CreateFcn(hObject, eventdata, handles)
877 % hObject    handle to english (see GCBO)
878 % eventdata  reserved - to be defined in a future version of MATLAB
879 % handles    empty - handles not created until after all CreateFcns called
880 global analyT
881 analyT=get(hObject, 'string');
882
883
884 % --- Executes on button press in si.
885 function si_Callback(hObject, eventdata, handles)
886 % hObject    handle to si (see GCBO)
887 % eventdata  reserved - to be defined in a future version of MATLAB
888 % handles    structure with handles and user data (see GUIDATA)
889 global analyT
890 analyT=get(hObject, 'string');
891 % Hint: get(hObject,'Value') returns toggle state of si
892
893
894
895 function edit17_Callback(hObject, eventdata, handles)
896 % hObject    handle to edit17 (see GCBO)
897 % eventdata  reserved - to be defined in a future version of MATLAB
898 % handles    structure with handles and user data (see GUIDATA)
899 global analyP
900
901 str=get(hObject, 'string');
902 analyP.CrossSectionP=str2double(str);
903
904 % Hints: get(hObject,'String') returns contents of edit17 as text
905 %        str2double(get(hObject,'String')) returns contents of edit17 as a double
906
907
908 % --- Executes during object creation, after setting all properties.
909 function edit17_CreateFcn(hObject, eventdata, handles)
910 % hObject    handle to edit17 (see GCBO)
911 % eventdata  reserved - to be defined in a future version of MATLAB
912 % handles    empty - handles not created until after all CreateFcns called
913 global analyP
914
915 % Hint: edit controls usually have a white background on Windows.
916 %       See ISPC and COMPUTER.
917 if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
918     set(hObject,'BackgroundColor','white');
919 end
920

```

```
921 str=get(hObject, 'string');
922 analyP.CrossSectionP=str2double(str);
```

### Addit.m

```
001 % This function look for all pixels belong to one particle. The function
002 % start with a coordinate of one pixel (normally the pixel in the left top
003 % site of the particle) and find out coordinates of all pixels belong to
004 % the particle.
005
006 % Authur: Haowei Wang (hwang23@emory.edu)
007 % Last updated Sep. 10th, 2011
008
009 function data=addit(m, n)
010
011 global biColor
012 test=[m n];
013 blobsT=[m n];
014 bn=1;
015 biColor(m,n)=0;
016 testT=[0 0];
017 i=1;
018
019
020 while test(1,1)>0
021     for j=1:size(test,1)
022         x=test(j,1);
023         y=test(j,2);
024         %     if x==0
025         %         test
026         %         j
027         %         testT
028         %     end
029
030 % Put all neighbor pixels to the array and label them.
031 % If all neighbor of a pixels have been looked at, the pixel itself will be
032 % moved out from the test array.
033
034         if biColor(x-1, y-1)==1
035             testT(i,:)=[x-1, y-1];
036             i=i+1;
037             biColor(x-1, y-1)=0;
038             blobsT(bn,:)=[x-1,y-1];
039             bn=bn+1;
040         end
041
042         if biColor(x, y-1)==1
043             testT(i,:)=[x, y-1];
044             i=i+1;
045             biColor(x, y-1)=0;
046             blobsT(bn,:)=[x,y-1];
047             bn=bn+1;
048         end
049
050         if biColor(x+1, y-1)==1
```

```

051         testT(i,:)= [x+1, y-1];
052         i=i+1;
053         biColor(x+1, y-1)=0;
054         blobsT(bn,:)= [x+1,y-1];
055         bn=bn+1;
056     end
057
058     if biColor(x-1, y)==1
059         testT(i,:)= [x-1, y];
060         i=i+1;
061         biColor(x-1, y)=0;
062         blobsT(bn,:)= [x-1,y];
063         bn=bn+1;
064     end
065
066     if biColor(x+1, y)==1
067         testT(i,:)= [x+1, y];
068         i=i+1;
069         biColor(x+1, y)=0;
070         blobsT(bn,:)= [x+1,y];
071         bn=bn+1;
072     end
073
074     if biColor(x-1, y+1)==1
075         testT(i,:)= [x-1, y+1];
076         i=i+1;
077         biColor(x-1, y+1)=0;
078         blobsT(bn,:)= [x-1,y+1];
079         bn=bn+1;
080     end
081
082     if biColor(x, y+1)==1
083         testT(i,:)= [x, y+1];
084         i=i+1;
085         biColor(x, y+1)=0;
086         blobsT(bn,:)= [x,y+1];
087         bn=bn+1;
088     end
089
090     if biColor(x+1, y+1)==1
091         testT(i,:)= [x+1, y+1];
092         i=i+1;
093         biColor(x+1, y+1)=0;
094         blobsT(bn,:)= [x+1,y+1];
095         bn=bn+1;
096     end
097 end
098 test=testT;
099 i=1;
100 testT=[0 0];
101 end
102 data=blobsT;
103 % if mask(m,n)==0;
104 % blobsT(bn,:)= [m,n];
105 % mask(m,n)=-1;
106 % bn=bn+1;

```

```

107 % around=[m-1 n-1; m n-1; m+1 n-1; m-1 n; m+1 n; m-1 n+1; m n+1; m+1 n+1];
108 % for i=1:8
109 %     if biColor(around(i,1), around(i,2))==1 && mask(around(i,1), around(i,2))==0
110 %         addit(around(i,1), around(i,2));
111 %     end
112 % end
113 % end

```

### Findtr.m

```

01 % This function looking for trace segments according to trace ID.
02
03 % Authur: Haowei Wang (hwang23@emory.edu)
04 % Last updated Sep. 10th, 2011
05
06 function data=findTr(traces, trID)
07
08 traceP=zeros(2);
09
10 for i=1:size(traces,1)
11     if traces(i,1)==-1 && traces(i,2)==trID
12         j=1;
13         p=i+1;
14         while traces(p,2)~=0
15             traceP(j,:)=traces(p,:);
16             j=j+1;
17             p=p+1;
18         end
19         break;
20     end
21 end
22 data=traceP;

```

### Readtr.m

```

01 % This function read coordinates of traced molecules from tracing file and
02 % return a matrix of coordinates of tracing.
03
04 % Authur: Haowei Wang (hwang23@emory.edu)
05 % Last updated Sep. 10th, 2011
06
07 function data = readtr(filename)
08 fd = fopen(filename);
09 data.tr=zeros(2,2);
10 i=1;
11
12 while feof(fd)==0
13     DataIn=fscanf(fd, '%d %d', 2);
14
15     if feof(fd)==1
16         break;
17     end
18     if i==1 && DataIn(1)>0
19         data.BASE=DataIn(1);
20         data.OVERWHELM=DataIn(2);
21     continue;

```

```

22     end
23     data.tr(i,1)=DataIn(1);
24     data.tr(i,2)=DataIn(2);
25     i=i+1;
26
27     end
28
29     fclose(fd);

```

### GroupAnalysis.m

```

001 % This program group all contact particles and DNA traces together.
002 % User can define the rules to analys those groups.
003
004 % The Group structure in each images contain three parts: Group.NoBlob is
005 % an array of trace IDs that do not interact with any blobs. Group.tr and
006 % Group.blobs are two 2*2 array. Each row of the arrays represents one
007 % group of blobs and tracings. For example, Group.tr(i,:) includes all the
008 % trace ID belong to i-th group; Group.blobs(i,:) includes all the blobs
009 % belong to i-th group.
010
011 % The structure of traces contains all traces of each image. traces.BASE is
012 % the basel of the image calculated by tracing program. traces.OVERWHELM
013 % is the overwhelming value which mostly represent the DNA height.
014 % traces.tr is an array of all traces. In traces.tr, the first row of each
015 % segment is [-1 traceID]; the last row is [-1 0]. Trace ID is a postive
016 % number generated by tracing program to identify each segments in one
017 % image.
018
019 % Blob data are saved in an array named blobs. The data structure is: each
020 % segment of data start with raw is [-1 blobID], the following two raw are
021 % [lowerx, lowery; upperx, uppery] which corresponding to the range of the
022 % blob. The following data are coordinates of every pixels blong to this
023 % blob.
024
025 % This program needs subfunctions below: readimage.m, readtr.m,
026 % relateData.m, addit.m, SegLength.m, findTr.m.
027 % Please keep them in the same folder of the program.
028
029 % Authur: Haowei Wang (hwang23@emory.edu)
030 % Last updated Sep. 10th, 2011
031
032 clear all
033 close all
034
035 global biColor mask ConDataT Group;
036
037 % data structure of results
038 SegTail=zeros(2);
039 segTN=1;
040 % end of data structure
041
042 THRESHOLD=1.4; % Set the threshold for elimination of DNA.
043 AreaSize=10; % Blobs contain less than this number of pixles will not be counted.
044
045 % Prepare the filenames of images need to be analyzed.

```

```

046
047 [filename, pathname, filterindex]=uigetfile('*.txt', 'pick a file', 'Multiselect', 'on');
048
049 CurrentP=pwd;
050 path(path,CurrentP);
051 cd(pathname);
052
053 if iscell(filename)
054     fileNum=size(filename, 2);
055 else
056     fileNum=1;
057 end
058 step=2;
059 Cstep=0;
060 final=zeros(2,1);
061 fn=1;
062
063 for i=1:fileNum
064     if fileNum==1
065         traceFile=filename;
066     else
067         traceFile=char(filename(i));
068     end
069     traces=readtr(traceFile);
070
071 % Display filename of current image.
072
073     traceFile
074
075     NameLength=size(traceFile, 2)-6;
076     imageFile=traceFile(1:NameLength);
077     image=readimage(imageFile);
078     ImageSize=size(image,1);
079
080
081     biColor=image;
082     biColor(biColor<(traces.OVERWHELM-traces.BASE)*THRESHOLD+traces.BASE)=0;
083     biColor(biColor>0)=1;
084     biColor(1,:)=0;
085     biColor(ImageSize,:)=0;
086     biColor(:,1)=0;
087     biColor(:,ImageSize)=0;
088
089     Modify=1;
090
091     blobs=zeros(2);
092     blobsI=1;
093     bID=1;
094     blobsT=zeros(2);
095     mask=zeros(ImageSize);
096
097 % This part looks for the area of each blobs
098
099     for m=1:ImageSize
100         for n=1:ImageSize
101             if biColor(m,n)==1

```

```

102         blobsT=addit(m,n);
103         if size(blobsT,1)>AreaSize
104             blobs(blobsI,:)=[-1 bID];
105             lowerx=min(blobsT(:,1));
106             lowery=min(blobsT(:,2));
107             upperx=max(blobsT(:,1));
108             uppery=max(blobsT(:,2));
109             blobs(blobsI+1,:)=[lowerx lowery];
110             blobs(blobsI+2,:)=[upperx uppery];
111             blobs(blobsI+3,:)=[-1 0];
112             blobsI=blobsI+3;
113             nextI=blobsI+size(blobsT,1);
114             blobs(blobsI+1:nextI,:)=blobsT;
115             blobsI=nextI+1;
116             bID=bID+1;
117             blobsT=zeros(2);
118         end
119     end
120 end
121 end
122
123 % pcolor(image)
124 % shading flat
125 % hold on
126 % plot(blobs(:,2), blobs(:,1)', 'LineStyle', 'none', 'Color', 'red', 'Marker', '.',
'MarkerSize',2);
127 % figure
128 % pcolor(image)
129 % shading flat
130
131 % The function relateData will generate a table of connection. The first
132 % line is trace ID, the other cells are blob ID that connect to the trace
133 % in same line.
134
135     ConData=relateData(traces.tr, blobs);
136     ConDataT=ConData;
137
138 % Put traces into different groups
139
140     Group.NoBlob=zeros(2,1);
141
142     gni=1;
143
144     for j=1:size(ConData,1)
145         if ConDataT(j,1)~=0
146             if ConDataT(j,2)==0
147                 Group.NoBlob(gni)=ConDataT(j,1);
148                 gni=gni+1;
149                 ConDataT(j,1)=0;
150             end
151         end
152     end
153     gni=1;
154
155     Group.tr=zeros(2);
156     Group.blobs=zeros(2);

```

```

157     groupN=1;
158
159     % Put all related traces and blobs together. If trace a connect to blob b;
160     % blob b connect to trace a & c; blob d connect to trace c, the ID of trace
161     % a, c and blob b, d will appear in same row of Group.tr and Group.blobs.
162
163     while sum(ConDataT(:,1))>0
164         test=1;
165         trN=1;
166         blobN=1;
167         for j=1:size(ConData,1)
168             if ConDataT(j,1)~=0
169                 Group.tr(groupN,1)=ConDataT(j,1);
170                 trN=trN+1;
171                 ConDataT(j,1)=0;
172                 for k=2:size(ConData, 2)
173                     if ConDataT(j,k)==0
174                         ConDataT(j,:)=0;
175                         break;
176                     end
177                     Group.blobs(groupN,blobN)=ConDataT(j,k);
178                     ConDataT(j,k)=0;
179                     blobN=blobN+1;
180                 end
181                 break;
182             end
183         end
184
185     while test==1
186         test=0;
187         for j=1:size(ConData,1)
188             if ConDataT(j,1)~=0
189                 for k=2:size(ConData,2)
190                     for l=1:size(Group.blobs, 2)
191                         if ConDataT(j,k)==Group.blobs(groupN,l)
192                             Group.tr(groupN,trN)=ConDataT(j,1);
193                             trN=trN+1;
194                             blobTMP=ConDataT(j,:);
195                             for m=2:size(blobTMP,2)
196                                 for n=1:blobN-1
197                                     addB=blobTMP(m);
198                                     if addB==Group.blobs(groupN,n)
199                                         addB=0;
200                                         break;
201                                     end
202                                 end
203                                 if addB>0
204                                     Group.blobs(groupN, blobN)=addB;
205                                     blobN=blobN+1;
206                                 end
207                             end
208                             ConDataT(j,:)=0;
209                             test=1;
210                             break;
211                         end
212                     end

```



```

213             if test==1
214                 break;
215             end
216         end
217     end
218     if test==1
219         break;
220     end
221 end
222
223     end
224     groupN=groupN+1;
225 end
226
227 % Generate a mask of blobs
228
229     maskB=zeros(512);
230     blobsM=blobs;
231
232     bID=0;
233     for j=1:size(blobs,1)
234         if blobsM(j,1)==-1
235             if blobsM(j,2)>0
236                 bID=blobsM(j,2);
237                 blobsM(j+1,1)=0;
238                 blobsM(j+2,1)=0;
239             end
240         else
241             if blobsM(j,1)>0
242                 maskB(blobsM(j,1), blobsM(j,2))=bID;
243             end
244         end
245     end
246
247 % analysis DNA contain only one blobs without loop
248     for j=1:size(Group.tr,1)
249         trJ=Group.tr(j,:);
250         trJN=trJ(trJ>0);
251         if size(trJN,2)>2
252             continue;
253         end
254
255         if size(trJN,2)==1
256             traceP=findTr(traces.tr, trJN);
257             if traceP(1,1)==0
258                 continue;
259             end
260             if maskB(traceP(1,1), traceP(1,2))>0
261                 continue;
262             end
263             Seg=zeros(2);
264             segN=1;
265             for k=1:size(traceP,1)
266                 if maskB(traceP(k,1), traceP(k,2))>0
267                     break;
268                 end

```

```

269         Seg(segN,:)=traceP(k,:);
270         segN=segN+1;
271     end
272     if size(Seg,1)<3
273         continue;
274     end
275     length1=SegLength(Seg);
276
277     Seg=zeros(2);
278     segN=1;
279     for l=k:size(traceP,1)
280         if maskB(traceP(1,1), traceP(1,2))>0
281             continue;
282         end
283         Seg(segN,:)=traceP(l,:);
284         segN=segN+1;
285     end
286     length2=SegLength(Seg);
287     if size(Seg,1)<3
288         continue;
289     end
290
291     if length1<2 || length2<2
292         continue;
293     end
294
295     if length1<length2
296         SegTail(segTN,:)=[length1 length2];
297         segTN=segTN+1;
298     else
299         SegTail(segTN,:)=[length2 length1];
300         segTN=segTN+1;
301     end
302 end
303
304 if size(trJN,2)==2
305     traceP1=findTr(traces.tr, trJN(1));
306     traceP2=findTr(traces.tr, trJN(2));
307     if traceP1(1,1)==0 || traceP2(1,1)==0
308         continue;
309     end
310     if maskB(traceP1(1,1), traceP1(1,2))>0
311         traceP=flipud(traceP1);
312         if maskB(traceP1(1,1), traceP1(1,2))>0
313             continue;
314         end
315     end
316
317     if maskB(traceP2(1,1), traceP2(1,2))>0
318         traceP=flipud(traceP2);
319         if maskB(traceP2(1,1), traceP2(1,2))>0
320             continue;
321         end
322     end
323
324     Seg=zeros(2);

```

```

325     segN=1;
326     traceP=traceP1;
327     for k=1:size(traceP,1)
328         if maskB(traceP(k,1), traceP(k,2))>0
329             break;
330         end
331         Seg(segN,:)=traceP(k,:);
332         segN=segN+1;
333     end
334     if size(Seg,1)<3
335         continue;
336     end
337     length1=SegLength(Seg);
338
339     Seg=zeros(2);
340     segN=1;
341     traceP=traceP2;
342     for k=1:size(traceP,1)
343         if maskB(traceP(k,1), traceP(k,2))>0
344             break;
345         end
346         Seg(segN,:)=traceP(k,:);
347         segN=segN+1;
348     end
349     if size(Seg,1)<3
350         continue;
351     end
352     length2=SegLength(Seg);
353
354     if length1<2 || length2<2
355         continue;
356     end
357
358     if length1<length2
359         SegTail(segTN,:)=length1 length2];
360         segTN=segTN+1;
361     else
362         SegTail(segTN,:)=length2 length1];
363         segTN=segTN+1;
364     end
365 end
366 end
367 end
368
369 % Measure the segment length of the shorter tail and put them into a
370 % histogram.
371
372 SegTail=SegTail*1000/512;
373 Clength=SegTail(:,1)+SegTail(:,2);
374 for i=1:size(Clength,1);
375     if Clength(i)<116.8
376         SegTail(i,:)=0;
377     end
378 end
379 pr=SegTail(:,1);
380 pr_V=pr(pr>0);

```

```

381 xout=[0:2:max(pr_V)*1.2];
382 n=histc(pr_V,xout);
383 bar(xout,n)
384 hold on
385 %ft=fitype('gauss4');
386 %opt=fitoptions();
387 cfun=fit(xout',n,'gauss1','Lower',[0 0 2],'Upper',[1000 100 500])
388 %cfun=fit(xout',n,'gauss2','Lower',[0 1 0 0 2.5 0],'Upper',[8000 2.5 500 8000 4 500])
389 %cfun=fit(xout',n,'gauss5','Lower',[0 0 20 0 280 20 0 490 20 0 780 20 0 1100
20],'Upper',[30 280 500 30 490 500 30 780 500 30 1100 500 30 1400 500])
390 %x2=[0:1:max(x1)'];
391 xout2=[0:0.5:max(pr_V)*1.2];
392 f2=feval(cfun,xout2');
393 plot(xout2,(f2),'r-','Linewidth',2);

```

### RelateData.m

```

01 % This function is written to find the relationship between particles and
02 % DNA traces. It will generate a connection table between traces and blobs.
03 % The first line of the table is trace ID. The other cells of the table are
04 % blob IDs. All blobs connect to the trace in the first cell of each row.
05 % If one blob connect to more than one trace, it will appear in different
06 % row.
07
08 % Authur: Haowei Wang (hwang23@emory.edu)
09 % Last updated Sep. 10th, 2011
10
11 function data = relateData(traces, blobs)
12
13 trToB=zeros(2);
14 maskB=zeros(512);
15
16 bID=0;
17 for i=1:size(blobs,1)
18     if blobs(i,1)==-1
19         if blobs(i,2)>0
20             bID=blobs(i,2);
21             blobs(i+1,1)=0;
22             blobs(i+2,1)=0;
23         end
24     else
25         if blobs(i,1)>0
26             maskB(blobs(i,1), blobs(i,2))=bID;
27         end
28     end
29 end
30
31 trN=1;
32 for i=1:size(traces,1)
33     if traces(i,1)==-1
34         if traces(i,2)>0
35             trToB(trN,1)=traces(i,2);
36             trN=trN+1;
37         end
38     else
39         if maskB(traces(i,1), traces(i,2))>0

```

```

40         bID=maskB(traces(i,1), traces(i,2));
41         j=2;
42         while j<size(trToB,2)
43             if trToB(trN-1,j)==0
44                 break;
45             end
46             if trToB(trN-1,j)==bID
47                 bID=0;
48             end
49             j=j+1;
50         end
51         if bID>0
52             trToB(trN-1,j)=bID;
53         end
54     end
55 end
56 end
57 data=trToB;

```

### ConvertJ.m

```

001 % This function convert DNA traces into NeuronJ version.
002
003 % Authur: Haowei Wang (hwang23@emory.edu)
004 % Last updated Sep. 10th, 2011
005
006 clear all
007 close all
008
009 dim=512;
010 dim=dim+1;
011
012 % Get filenames of DNA traces.
013
014 [filename, pathname, filterindex]=uigetfile('*.txt', 'pick a file', 'Multiselect', 'on');
015
016 targetDir=uigetdir(pathname, 'Select target folder.');
```

```

017
018 CurrentP=pwd;
019 path(path,CurrentP);
020
021 if iscell(filename)
022     fileNum=size(filename, 2);
023 else
024     fileNum=1;
025 end
026 step=2;
027 Cstep=0;
028 final=zeros(2,1);
029 fn=1;
030
031 for i=1:fileNum
032     if fileNum==1
033         traceFile=filename;
034     else
035         traceFile=char(filename(i));

```

```

036     end
037
038     cd(pathname);
039
040     % Read traces.
041
042     traces=readtr(traceFile);
043
044     % Display filename of tracing.
045
046     traceFile
047
048     length=0;
049     prevP=[0 0];
050     newTr=zeros(2);
051     Lthres=45;
052
053     n=size(traces.tr,1);
054
055     start=1;
056     sEnd=1;
057
058     for m=2:n
059         if traces.tr(m,1)==-1
060             if traces.tr(m,2)>0
061                 start=m;
062                 prevP=traces.tr(m+1,:);
063             end
064             if traces.tr(m,2)==0
065                 sEnd=m;
066                 sNum=sEnd-start+1;
067
068             % Very short traces will not be disregarded.
069
070             if length>Lthres
071                 pointN=size(newTr,1);
072                 newTr(pointN+1:pointN+sNum,:)=traces.tr(start:sEnd,:);
073             end
074
075             length=0;
076
077         end
078         else
079             length=length+sqrt((traces.tr(m,1)-prevP(1))^2+(traces.tr(m,2)-prevP(2))^2);
080             prevP=traces.tr(m,:);
081         end
082     end
083
084
085     n=size(newTr,1);
086
087     % Prepare head of NeuroJ file.
088
089     if n>4
090         l=size(traceFile,2);
091         traceJ=traceFile(1:l-6);

```

```

092     traceJ(l-9)='_';
093
094     cd(targetDir);
095     fd=fopen(strcat(traceJ,'.ndf'), 'w');
096
097     fprintf(fd, '// NeuronJ Data File - DO NOT CHANGE\r');
098     fprintf(fd, '1.4.0\r');
099     fprintf(fd, '// Parameters\r');
100     fprintf(fd, '0\r');
101     fprintf(fd, '2.0\r');
102     fprintf(fd, '0.7\r');
103     fprintf(fd, '0\r');
104     fprintf(fd, '800\r');
105     fprintf(fd, '5\r');
106     fprintf(fd, '5\r');
107     fprintf(fd, '1\r');
108     fprintf(fd, '// Type names and colors\r');
109     fprintf(fd, 'Default\r');
110     fprintf(fd, '4\r');
111     fprintf(fd, 'Axon\r');
112     fprintf(fd, '7\r');
113     fprintf(fd, 'Dendrite\r');
114     fprintf(fd, '1\r');
115     fprintf(fd, 'Primary\r');
116     fprintf(fd, '7\r');
117     fprintf(fd, 'Secondary\r');
118     fprintf(fd, '1\r');
119     fprintf(fd, 'Tertiary\r');
120     fprintf(fd, '8\r');
121     fprintf(fd, 'Type 06\r');
122     fprintf(fd, '4\r');
123     fprintf(fd, 'Type 07\r');
124     fprintf(fd, '4\r');
125     fprintf(fd, 'Type 08\r');
126     fprintf(fd, '4\r');
127     fprintf(fd, 'Type 09\r');
128     fprintf(fd, '4\r');
129     fprintf(fd, 'Type 10\r');
130     fprintf(fd, '4\r');
131     fprintf(fd, '// Cluster names\r');
132     fprintf(fd, 'Default \r');
133     fprintf(fd, 'Cluster 01\r');
134     fprintf(fd, 'Cluster 02\r');
135     fprintf(fd, 'Cluster 03\r');
136     fprintf(fd, 'Cluster 04\r');
137     fprintf(fd, 'Cluster 05\r');
138     fprintf(fd, 'Cluster 06\r');
139     fprintf(fd, 'Cluster 07\r');
140     fprintf(fd, 'Cluster 08\r');
141     fprintf(fd, 'Cluster 09\r');
142     fprintf(fd, 'Cluster 10\r');
143
144     trN=1;
145     iCount=0;
146     iSeg=3;
147     segNum=1;

```

```

148
149 % Put in trace data
150
151     for j=3:n
152         if newTr(j,1)==-1
153             if newTr(j,2)==0
154                 if iCount~=0
155                     fprintf(fd, '%d\r', newTr(j-1,2));
156                     fprintf(fd, '%d\r', dim-newTr(j-1,1));
157                     iCount=0;
158                 end
159                 iSeg=3;
160                 segNum=1;
161             else
162                 sstr=strcat('// Tracing N', num2str(trN), '\r');
163                 fprintf(fd, sstr);
164                 fprintf(fd, [num2str(trN) '\r']);
165                 fprintf(fd, '0\r');
166                 fprintf(fd, '0\r');
167                 fprintf(fd, 'Default\r');
168                 trN=trN+1;
169             end
170         else
171             if iSeg==3
172                 sstr=strcat('// Segment_', num2str(segNum), ' of Tracing N', num2str(trN-1),
173 '\r');
174                 sstr(11)=' ';
175                 fprintf(fd, sstr);
176                 segNum=segNum+1;
177                 iSeg=0;
178             end
179             if iCount==0
180                 fprintf(fd, '%d\r', newTr(j,2));
181                 fprintf(fd, '%d\r', dim-newTr(j,1));
182                 iSeg=iSeg+1;
183             end
184
185             iCount=iCount+1;
186
187             if iCount==4
188                 iCount=0;
189             end
190
191         end
192     end
193     fprintf(fd, '// End of NeuronJ Data File\r');
194     fclose(fd);
195 end
196 end

```

### Imreverse.m

```

01 % This program reverse color of TIF image and zoom out it to fit six images
02 % into one page of word document.
03 % The modified image will be saved as JPE files.

```



```

04
05 % Authur: Haowei Wang (hwang23@emory.edu)
06 % Last updated Sep. 10th, 2011
07
08 clear all
09 close all
10
11 try
12     [filename, pathname, filterindex]=uigetfile('.tif', 'pick a file', 'Multiselect', 'on');
13     if ~iscell(filename)
14         if filename(1)==0
15             clear all
16             close all
17             error('Cannot find any files.');
```

```

18         end
19     end
20 catch
21     clear all
22     close all
23     error('Cannot find any files.');
```

```

24 end
25 currentP=pwd;
26 path(path,currentP);
27 cd(pathname);
28 mkdir('reverse');
```

```

29
30 if iscell(filename)
31     fileNum=size(filename, 2);
32 else
33     fileNum=1;
34 end
35
36 for traceN=1:fileNum
37     if fileNum==1
38         file=filename;
39     else
40         file=char(filename(traceN));
41     end
42     if ~isequal(file, 0)
43         try
44             a=imread(file);
45         catch
46             continue;
47         end
48         writeout=imresize(255-a, 0.55);
49         fileR=file(1:size(file,2)-4);
50         cd('reverse');
```

```

51         imwrite(writeout, strcat('r_', fileR, '.jpg'));
52         cd(pathname);
53     end
54 end

```