

**COMPOSITIONAL COMPLEXITY MEASURES  
OF DNA SEQUENCE  
USING MULTIFRACTAL TECHNIQUES**

by

HONG ZHANG

A Thesis

Submitted to the Faculty of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

University of Manitoba

Winnipeg, Manitoba, Canada

Thesis Advisor: Prof. W. Kinsner, Ph. D., P. Eng.

© Hong Zhang; October, 2001

(xv + 111 + A2 + B27) = 155 pp.



National Library  
of Canada

Acquisitions and  
Bibliographic Services

395 Wellington Street  
Ottawa ON K1A 0N4  
Canada

Bibliothèque nationale  
du Canada

Acquisitions et  
services bibliographiques

395, rue Wellington  
Ottawa ON K1A 0N4  
Canada

*Your file Votre référence*

*Our file Notre référence*

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-80099-7

Canada

**THE UNIVERSITY OF MANITOBA  
FACULTY OF GRADUATE STUDIES  
\*\*\*\*\*  
COPYRIGHT PERMISSION PAGE**

**Compositional Complexity Measures of DNA Sequence Using Multifractal Techniques**

**BY**

**Hong Zhang**

**A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University  
of Manitoba in partial fulfillment of the requirements of the degree**

**of**

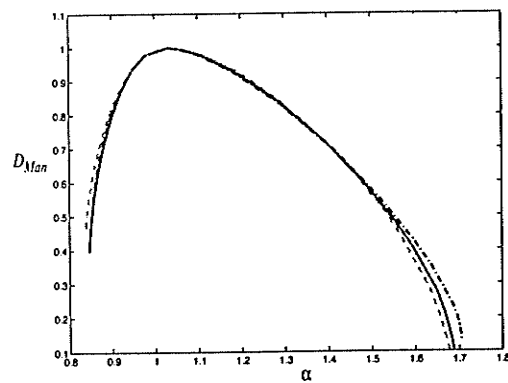
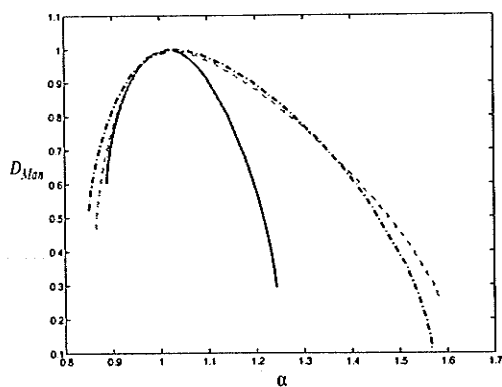
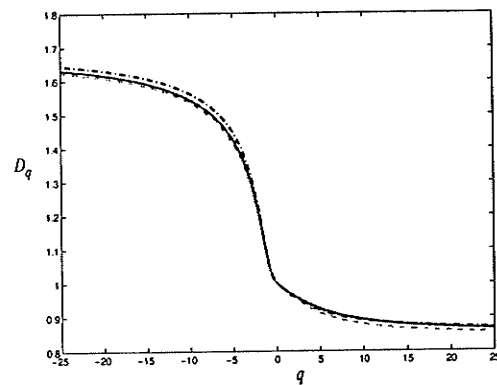
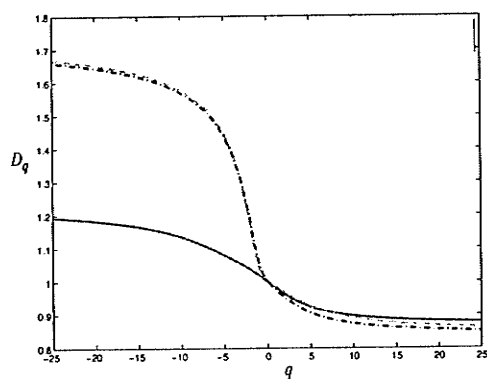
**MASTER OF SCIENCE**

**HONG ZHANG ©2001**

**Permission has been granted to the Library of The University of Manitoba to lend or sell copies of this thesis/practicum, to the National Library of Canada to microfilm this thesis and to lend or sell copies of the film, and to University Microfilm Inc. to publish an abstract of this thesis/practicum.**

**The author reserves other publication rights, and neither this thesis/practicum nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.**

GATCCATCCGCCTCGGCCTCCCAAAGTGCTGGGATTACAGGCGTGAGCCACCG  
 CGCCCGGCCCCCAACCTTGGGACATTTTCATCCATTCATTCATCCTTTTTTTT  
 TTTTTTTTTTTGAGACGGAGTCTTGCTCTGTCACCCAGGCTGGAGTGCAGGGG  
 CAAGATCTCAGCTCCTGCACCCTCCACCTTCCGGATTCAAGTGATTCTCCTGC  
 CTCAGCCTCCCAAGTAGTTGGGATTACAGGCATGCCATCAACATGTCTGGCTA  
 ATTTTTGTATTTTAGTAGAAATGGGGTTTCACCATGTTGGCCAGGCTGGTCT  
 CGAACTCCTGACTTCAGGTGATCCTCCCACCTCAGCCTCCCAAAGTGCTGGGA  
 TTACAGGTATGAGCCACCGCGCCTGGCGCATGGGCACATCCATTGAGTGTGCA  
 CTTGGTGCCAAGTTCTGTGCCAGGCACAGGCAATTCAACATTTATTGGAATGA  
 TGAGTCCCTGTCTGCATGGAATTCATAGGCTAGAGGAGGAAGCAGTTTGCCT  
 CTGGTCCCATGGCCAGAGCAGCCCCAGGTGAAGGTTATGAATTTATTGTCCCA  
 TCTAATGGTGTTCAGCAGTCTGCCACATGGTGGGAAGGAGGCCCCACAGAGC  
 TGTGCTGTCTCCTTCCCAGGATGAGCTGGAGCACAGCCTGGGGGAGAGTGCGG



## ABSTRACT

This thesis presents a study of the chaotic property of DNA sequences and an approach for characterization of DNA sequences based on multifractal techniques. The DNA sequence analysis provided in the thesis is motivated by the possibility of identifying biological functionality using information contained within the DNA sequences.

Numerical mapping is the basis of DNA sequence analysis. Improper mapping may introduce artifacts into the resulting DNA signals. To resolve this problem, a new numerical mapping method based on the organism's codon usage is introduced. Using mutual information and false nearest neighbourhood analysis, it has been shown that there is a strong correlation among the three bases within the codons, and that DNA sequences have a high-dimensional structure. A novel model of multifractal measures for an  $m$ -dimensional object is proposed in the thesis. With this model, it is shown that the three-dimensional Lorenz attractor and the x-variable time series of the Lorenz system have a similar structure of the Rényi dimension spectrum in general. Rather than using DNA signals, an approach based on frame signals for Rényi and Mandelbrot dimension spectra analysis is developed. The experimental results of Rényi and Mandelbrot dimension spectra demonstrate that there is a significant difference between the open reading frames and the other non-coding reading frames. Furthermore, local Rényi dimension analysis with different resolution reveals the interior structure of the genes and the genomic DNA sequences. Therefore, it opens up a possible way for coding prediction as well as for extracting the higher order structure information stored in the genomic DNA sequences.

## ACKNOWLEDGMENTS

First of all, I would like to thank my advisor, Dr. W. Kinsner, for his time and support during my studies and for his encouragement and patience throughout the course of this thesis.

I would also like to thank my colleagues in the Delta Research Group, Richard Dansereau, Bin Huang, Jin Chen, Alexis Denis, Tina Ehtiati, Jonathan Greenberg, Luotao Sun, Steven Miller, Reza Fazel, Abdelhakim El-Boustani, Jizong Li, Kalen Brunham, Sharjeel Siddiqui, Robert Barry, Michael Potter, and Martin Stadler. Your help, one way or another, benefit me a lot in my development as a graduate student.

Finally, my special thank goes to my entire family. Their constant support and patience supported me throughout this research. This thesis is devoted to my wife, Cheena, and my son, Stephen.

Partial financial support is acknowledged from the Natural Sciences and Engineering Research Council (NSERC) of Canada and Show Cable, Winnipeg.

---

## TABLE OF CONTENTS

ABSTRACT .....	v
ACKNOWLEDGEMENTS .....	vi
TABLE OF CONTENTS .....	vii
LIST OF FIGURES .....	xi
LIST OF TABLES .....	xiv
LIST OF ABBREVIATIONS AND SYMBOLS .....	xv
<b>CHAPTER I</b>	
<b>INTRODUCTION .....</b>	<b>1</b>
Problem Definition .....	1
Goal and Objectives .....	3
Thesis Organization .....	4
<b>CHAPTER II</b>	
<b>BACKGROUND ON GENOMICS .....</b>	<b>6</b>
Structural Genomics .....	6
DNA Molecule .....	6
Gene .....	9
Genome .....	11
Chromosome .....	11
Genome Structure .....	14
DNA Organization .....	16
Functional Genomics .....	19
RNA .....	19
Protein .....	21
Gene Expression .....	21
Transcription Process .....	21
Translation Process .....	23

---

Codon Usage .....	25
Computational Analysis of DNA Sequences .....	26
Gene Prediction .....	27
Complexity Analysis of DNA Sequence .....	30
Summary .....	32
<b>CHAPTER III</b>	
<b>BACKGROUND ON FRACTALS AND CHAOS .....</b>	<b>33</b>
Fractals .....	33
Fractal Sets .....	33
Fractal Dimensions .....	34
Multifractal Dimensions .....	38
Rényi Generalized Entropy and Dimension Spectrum .....	38
Mandelbrot Spectrum .....	40
Chaos and Strange Attractors .....	41
Chaotic Dynamical Systems .....	41
Strange Attractors .....	43
Characterization of Dynamical Systems .....	45
Time Delay Approach .....	46
Mutual Information Method for Choosing Lag .....	48
False Nearest Neighbourhood Method for Choosing Embedding Dimension .....	50
Distinguishing Stochastic System From Deterministic Chaos .....	52
Summary .....	55
<b>CHAPTER IV</b>	
<b>CHARACTERIZATION OF THE DNA SIGNALS .....</b>	<b>56</b>
Numerical Mapping of the Symbolic DNA Sequences .....	56
The Size of a Point of an Object in Multifractal Analysis .....	58
Multifractal Model of Numerical Series .....	62
Approximations of the Multifractal Measures on the Numerical Series .....	63

---



---

Multifractal Measures on the Single Fractal Numerical Series .....	64
Multifractal Measures on the Multifractal Numerical Series .....	66
Local Rényi Dimensions of the DNA Signals .....	69
Window Size .....	70
Sliding Step .....	71
Summary .....	71
<b>CHAPTER V</b>	
<b>EXPERIMENT DESIGN .....</b>	<b>72</b>
The Experimental DNA Sequences .....	72
The Generated random and Cantor DNA Sequences .....	72
The Human DNA Sequences .....	73
Numerical Mapping of the DNA Symbolic Sequences .....	75
Chaotic Characterization of the DNA Sequences .....	76
Feature Extraction of the DNA Signals .....	77
Local Fractal Dimension Analysis .....	77
Summary .....	78
<b>CHAPTER VI</b>	
<b>EXPERIMENTAL RESULTS AND DISCUSSION .....</b>	<b>79</b>
Chaotic Properties of the DNA Sequences .....	79
Mutual Information Analysis of the DNA Sequences .....	79
False Nearest Neighbourhood Analysis of the DNA Sequences .....	82
Multifractal Dimension Analysis of the DNA Sequences .....	84
Rényi Dimension Spectrum of the DNA Sequences .....	84
Mandelbrot dimension Spectrum of the DNA Sequences .....	90
Local Rényi dimension Analysis of the DNA sequences .....	96
Analysis With High Resolution Reveals Gene Structures .....	96
Analysis With Low Resolution Reveals Genomic DNA Structures .....	98
Summary .....	100

---

**CHAPTER VII**  
**CONCLUSIONS.....101**

    Conclusions .....101  
    Contributions .....102  
    Recommendations .....103

**REFERENCES .....104**

**APPENDICES**

    A: Generated Samples Sequence ..... A1-2  
    B: Source Code .....B1-27

---

## LIST OF FIGURES

Fig. 2.1	The structure of a DNA molecule .....	7
Fig. 2.2	The gene structure of eukaryotes .....	10
Fig. 2.3	The Structure of a higher organism genome .....	11
Fig. 2.4	Human karyotype and chromosome banding .....	13
Fig. 2.5	Basic functional elements in chromosomes of higher organisms .....	14
Fig. 2.6	The packaging of DNA in eukaryotic cells .....	17
Fig. 2.7	The hierarchy of chromosomal structure in metaphase .....	18
Fig. 2.8	Overview of RNA processing in eukaryotes, using $\beta$ -globin gene as an example .....	23
Fig. 2.9	Reading frames and mRNA .....	24
Fig. 3.1	Generation of the Cantor set .....	34
Fig. 3.2.	The illustration of multifractal dimension spectrum for a single fractal, the Cantor set, and a multifractal object .....	39
Fig. 3.3	The one dimensional trajectory of the Lorenz system for $\sigma = 10$ , $\eta = 8/3$ , and $\gamma = 28$ with the initial values at $x(0) = 0$ , $y(0) = 1$ , and $z(0) = 0$ .....	42
Fig. 3.4	The strange attractor of the Lorenz system .....	43
Fig. 3.5	Lorenz attractor reconstructed from the x component trajectory .....	47
Fig. 3.6	The mutual information function of the (a) white noise and (b) Lorenz attractor .....	49
Fig. 3.7	The percentage of false nearest neighbourhoods for the Lorenz attractor .....	53
Fig. 3.8	The percentage of false nearest neighbourhoods for white noise .....	53
Fig. 3.9	The reconstructed white noise system with a lag of 1 for (a) and 10 for (b) .....	54
Fig. 4.1	The illustration of fractal dimensions of a plane and a cube .....	59
Fig. 4.2	The Rényi dimension spectrum of the Cantor numerical series .....	65
Fig. 4.3	The Mandelbrot Spectrum of The Cantor numerical series .....	65
Fig. 4.4	The Rényi dimension spectrum of the x-variable trajectory of the Lorenz system .....	67
Fig. 4.5	The Mandelbrot Spectrum of the x-variable trajectory of the Lorenz system .....	68
Fig. 4.6	The Rényi dimension spectrum of the Lorenz system .....	68

---

---

Fig. 4.7	The Mandelbrot spectrum of the Lorenz system .....	69
Fig. 5.1	An illustration of numerical mapping of the DNA sequence .....	75
Fig. 6.1	The mutual information analysis of the DNA signal and frame signals associated to a random generated white noise DNA sequence with a size of 20 kb .....	80
Fig. 6.2	The mutual information analysis of the human HD gene cDNA sequence .....	80
Fig. 6.3	The mutual information analysis of the human NEB gene cDNA sequence .....	81
Fig. 6.4	The mutual information analysis of x-variable trajectory of the Lorenz system .....	81
Fig. 6.5	The false nearest neighborhood analysis of the human HD gene cDNA sequence.....	83
Fig. 6.6	The false nearest neighbourhood analysis of the human NEB gene cDNA sequence.....	83
Fig. 6.7	The Rényi dimension spectra of the random DNA sequence .....	85
Fig. 6.8	The Rényi dimension spectra of the Cantor DNA sequence.....	85
Fig. 6.9	The Rényi dimension spectra of the HYAL2 cDNA coding sequence .....	86
Fig. 6.10	The Rényi dimension spectra of the first exon of the HYAL1 gene .....	86
Fig. 6.11	The Rényi dimension spectra of the third exon of the HYAL2 gene .....	87
Fig. 6.12	The Rényi dimension spectra of the first intron of the HYAL2 gene.....	87
Fig. 6.13	The Rényi dimension spectra of the 5' flank region of the HYAL2 gene .....	88
Fig. 6.14	The Rényi dimension spectra of the 3' flank region of the HYAL2 gene .....	88
Fig. 6.15	The Rényi dimension spectra of the ph-20, a human genomic DNA sequence which contains the entire HYAL2 gene and the 5' flank region, the first exon and intron of the HYA11 gene .....	89
Fig. 6.16	The Rényi dimension spectra of the H19 gene .....	89
Fig. 6.17	The Mandelbrot spectra of the random DNA sequence .....	91
Fig. 6.18	The Mandelbrot spectra of the Cantor DNA sequence .....	92
Fig. 6.19	The Mandelbrot Spectra of the HYAL2 cDNA sequence .....	92
Fig. 6.20	The Mandelbrot Spectra of the third exon of the HYAL1 gene .....	93
Fig. 6.21	The Mandelbrot Spectra of the first exon of the HYAL2 gene .....	94
Fig. 6.22	The Mandelbrot Spectra of the 5' flank sequence of the HYAL2 gene .....	94

---

---

Fig. 6.23	The Mandelbrot Spectra of the first intron sequence of the HYAL2 gene .....	95
Fig. 6.24	The Mandelbrot Spectra of the 3' flank sequence of the HYAL2 gene .....	95
Fig. 6.25	The Mandelbrot Spectra of the genomic DNA sequence, ph-20 .....	96
Fig. 6.26	High resolution local fractal dimension analysis of the $\beta$ -globin gene .....	97
Fig. 6.27	Low resolution local fractal dimension analysis of the human genomic DNA sequence .....	99

## LIST OF TABLES

Table 2.1	Genetic code and human codon usage .....	20
Table 5.1	The information of the experimental human DNA sequences .....	74

**LIST OF ABBREVIATIONS AND SYMBOLS**

3'	The downstream end of a DNA sequence
5'	The upstream end of a DNA sequence
$\alpha$	Hölder exponent
A	Adenine
bp	Base pair
C	Cytosine
cDNA	Complementary DNA
D*	Composite dimension
DNA	Deoxyribonucleic acid
$D_\beta$	Spectral dimension
$D_H$	Hausdorff dimension
$D_I$	Information dimension
$D_{Man}$	Mandelbrot dimension
$D_q$	Rényi dimension
$D_s$	Self-similarity dimension
G	Guanine
HGP	Human Genome Project
$H_s$	Shannon entropy
mRNA	Messenger RNA
ORF	Open reading frame
RNA	Ribonucleic acid
T	Thymine
tRNA	Transfer RNA
vel	Volume element
U	Uracile

# CHAPTER I

## INTRODUCTION

### 1.1 Problem Definition

Better understanding of our life and other living systems may benefit our society. It may help to raise better plants and animals, create enhanced pharmaceuticals for our health and improve living level, develop new sources of energy, as well as mitigate the long-term impacts of climate change and clean up the environment. An ambitious program, Genomes to Life, has been launched by the Department of Energy of the USA in 2001. The goals of this program are to achieve a fundamental, comprehensive, and systematic understanding of life [DOE01].

Understanding genomes is a fundamental step to understanding life. A genome stores a complete and complex set of instructions, which is embedded in the deoxyribonucleic acid (DNA) sequence, necessary for building and maintaining the life of an organism. Found in many trillions of cells in our body, the human genome contains the information for all cellular structures and lifetime activities of the cells in our body, as well as for body growth, development, and its functions. This information is mostly stored in the genes. A gene is a piece of DNA sequence which is composed of exons and introns in higher eukaryotic organisms. It has been long known that the exons, called coding regions, of the genome carry information which instructs the cellular processes in the way of leading the events from DNA sequences to amino acid sequences or proteins, while the introns of the genes and the intergenic regions, which are called non-coding regions con-



---

tains no information for making the proteins in the organism. The proteins in the organism do essentially all the work of the cells.

Launched in 1986, the Human Genome Project (HGP) is currently being completed with a great success. It has recently published a draft sequence of the human genome, which covers 96% of the human genome containing  $3 \times 10^9$  base pairs of DNA. However, obtaining the DNA sequences of the entire genomes of the human and other organisms is just the beginning of understanding our and other organisms' genomes. The immediate challenge is characterization of the genome structures, including the mapping and packing structure of the total set of genes and their regulatory elements.

A vast amount of genomic DNA sequences has been sequenced to date with an exponential growth rate due to the tremendous improvement of the sequencing techniques. For example, new sequences averaging about 30 million DNA bases (which represents approximately 105 genes and their respective proteins) were sequenced every day in 2000 [NCBI01] [Uber01]. However, the progress of characterizing genome structure and the pace of gene discovery actually is rather slow because of the limitations of the traditional biological techniques.

So far, only a small fraction of genes that cause human genetic disease have been identified. Each new gene revealed by genome sequence analysis has the potential to significantly affect human health. Within the human genome, it is estimated that total of 6,000 genes have a direct impact on the diagnosis and treatment of human genetic diseases. The timely development of diagnostic techniques and treatment for these diseases

has immeasurable value for the world. Computational analysis is a key component that can contribute significantly to the knowledge to effect such developments. In addition, new computational methods will provide complementary information which can be of benefit for gene prediction by the traditional experimental methods.

Most of the current research in deciphering the meaning of DNA sequences is approached from the low base-pair level. Its main objective is to search for patterns or correlations existing in the DNA sequence related to codons, amino acids, and proteins. A number of gene prediction systems have been developed in recent years. These systems use a variety of sophisticated computational techniques, including neural network [UbMu91], dynamic programming [SnSt93], rule-based methods [SoSL94], decision trees [HuHa92], probability reasoning [GKDS92] and hidden Markov chains [HeSF97]. Most of these techniques rely on the statistical qualities of exons in the genome and, therefore, the fundamental limitation of them is the use of a known gene data pool as a training set for their classification. Consequently, they are capable of finding only the genes that are homologous with those in the training data set.

It has been demonstrated that fractal techniques [Kins94] can be useful in the classification of stationary and nonstationary signals such as speech, image, human fingerprints, biological signals, and radio transmitter transients [Lang96] [Chen97] [Shaw97] [Jang97] [Grie96] [Ehti99] [Dans01].

## **1.2 Goal and Objectives**

The general goal of this thesis is to develop techniques for structural characteriza-

---

---

tion of DNA sequences, with the main interest in coding region prediction for genomic DNA sequences. More specifically, the following objectives have to be achieved:

- (i) A method of characterizing the chaotic property of DNA sequences;
- (ii) A technique for extracting features from the structural information in DNA sequences in order to be able to distinguish the coding regions from the non-coding pool in a given genomic DNA sequence; and
- (iii) A technique for on-line coding region prediction of genomic DNA sequences.

To address these objectives, this thesis focuses mainly on fractal and multifractal techniques [Kins94] for feature extraction. The established methods such as mutual information criterion [FrSw86] are also used for chaotic characterization of the DNA sequences.

### **1.3 Thesis Organization**

This thesis is organized in seven chapters. Chapter 1 states the motivation, objectives and goal for this thesis. Chapter 2 contains the background information on genomics. Chapter 3 gives the background knowledge on fractal, multifractal, as well as chaos and stranger attractors. The theoretical basis for reconstructing strange attractors from a single variable time series and chaos characterizing criteria are also provided in this chapter. In Chapter 4, experimental algorithms for DNA sequences analysis are described. Chapter 5 provides experimental design and experimental parameter choosing. The experimental

results of characterization and classification of the DNA sequences are given in Chapter 6.

Conclusions, recommendations and contributions are presented in Chapter 7.

## CHAPTER II

# BACKGROUND ON GENOMICS

This chapter provides the basic concepts of molecular biology as well as an overview of the research in DNA sequence analysis and the techniques of gene prediction. First, a background knowledge of DNA, gene, genome, as well as gene expression is provided. A brief review of codon usage is then described. The chapter finishes by discussing the problems and techniques of gene prediction and DNA sequence structural analysis from the point of view of bioinformatics.

### 2.1 Structural Genomics

A *genome* contains a full set of genetic instructions for the organism and allows a sharing of the knowledge with offspring, from simple bacteria to remarkably complex human beings. A genome is made of DNA (deoxyribonucleic acid). Understanding how a genome functions requires the knowledge of its structure and organization.

#### 2.1.1 DNA Molecule

A *DNA* molecule is composed of smaller units called *nucleotides*. Tens of thousands of nucleotides link together in a polynucleotide chain. The upstream end of a DNA chain is called the 5' end of the chain and the downstream end is called the 3' end of the chain. Two DNA chains wrap around each other to resemble a twisted helical ladder (Fig. 2.1).

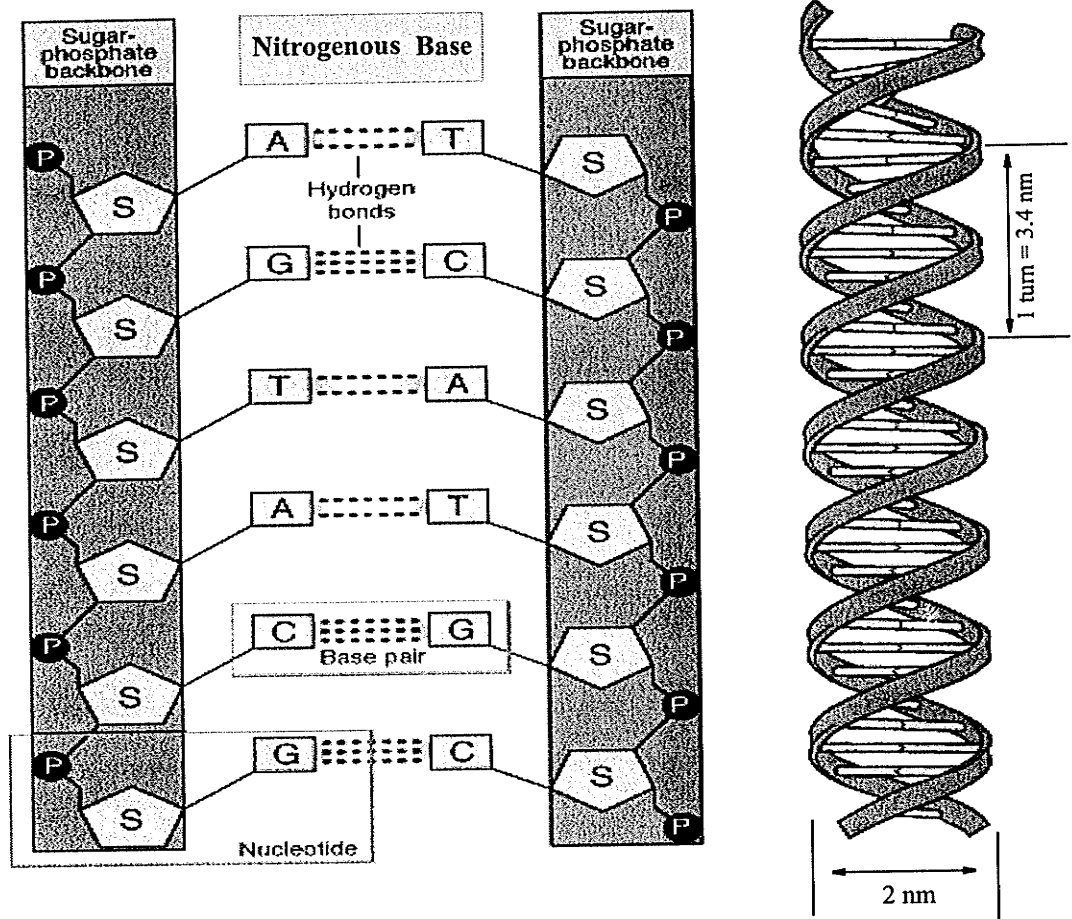


Fig. 2.1. The structure of a DNA molecule (after [NHGI01]).

As shown in Fig. 2.1, a nucleotide consists of three parts: a *deoxyribose sugar residue*, a *phosphate group*, and a *nitrogenous base*. The sugar of one nucleotide links to the phosphate group of the next. The sugar and phosphate are often called the “backbone” of the DNA. The nitrogenous base is the part of the nucleotide that carries hereditary information, so the words “nucleotide” and “base” are often used interchangeably. There are four main nitrogenous bases found in DNA: *adenine*, *thymine*, *cytosine*, and *guanine*, abbreviated as A, T, C, and G, respectively. Adenine and guanine are double-ringed

*purines*. Thymine and cytosine are single-ringed *pyrimidines*. The two polynucleotide chains are held together by van der Waals forces, weak hydrogen bonds between the nitrogenous bases on each chain, forming *base pairs* (bp). The hydrogen bonding between complementary base pairs is such that the most energetically stable DNA configuration is achieved when adenine pairs with thymine and guanine pairs with cytosine. Although the spatial requirements of DNA potentially allow four complementary base pairs to be formed (i.e., G-T, G-C, A-T, and A-C), only the G-C and A-T base pairs are normally found in DNA. In other words, the order of bases on one DNA strand, or a side of the ladder, determines the bases on the complementary DNA strand, or the other side of the ladder. Three hydrogen bonds stabilize G-C base pairs and two hydrogen bonds stabilize A-T base pairs. Because hydrogen bonding between base pairs contributes to the stability of the DNA double helix, base sequence affects the stability of DNA. This means that the regions of the DNA with an abundance in G-C base pairs are more stable than A-T rich regions of the DNA.

The helical structure, described in Fig. 2.1, is called *B-Form* DNA. It was found by James D. Watson and Francis Crick in 1953. B-DNA is only one of several possible configurations. Other DNA conformations use the same nucleotides and molecular bonds, but the three-dimensional structure of the helix is different. At least six different DNA configurations (designated A, B, C, D, E, and Z) have been identified, but only the A, Z, and B conformations are found in nature. B-Form DNA is the most common form of DNA found in living organisms. It has an average diameter 2.0 nm and approximate 10.1 to 10.6 bp per turn. *A-Form* DNA, which is present in RNA, has a diameter of 2.3 nm and 11 bp per turn. *Z-Form* DNA has a diameter of 1.8 nm and 12 bp per turn. Unlike B-Form and A-

Form DNA, Z-Form is a left-hand helix. Only a very restricted set of DNA sequences appear able to adopt the Z-Form structure. The biological significance of the range of structures accessible to particular DNA sequences is not fully understood [Sind94].

Composed of four letters (A, T, C, and G), DNA sequences are often used to represent the order of DNA nucleotides in DNA molecules since (1) only the nitrogenous bases contain genetic information, (2) the sugar-phosphate backbone mainly maintains the same structure along the DNA molecules, and (3) one DNA strand determines the complementary strand.

### 2.1.2 Gene

A *gene* is a specific small piece of DNA and is the basic physical and functional unit of heredity. Roughly speaking, each gene carries a set of instructions required for the constructing of one specific protein. *Proteins* are a diverse group of large, complex molecules that determine, among other things, how the organism looks, how well its body metabolizes food or fights infection, mediate much of the information flow within a cell, and sometimes even how it behaves. Genes also contain the information that help to control where, when, and in what amount proteins are produced.

A gene is composed of several parts. As shown in Fig. 2.2, in higher organisms, the protein-making instructions are broken up into relatively short sections named *exons*. The exons are separated by longer sections of “nonsense” DNA, called *introns*. For genes in higher organisms, the size of exons is small (on the average, 145 bp for humans) but long in introns (some exceeding 10 kb) [HGP01]. A gene also contains regulatory sequences, named *regulatory elements*. Most of the regulatory elements of a gene are located in the 5'



flank region of the gene, while some may be located in the 3' flank region, or buried in the middle of the gene.

The regulatory elements are crucial to how a living system works. With the binding of specific proteins onto its regulatory elements, the gene is turned on or off. Most higher organisms are composed of different kinds of cells, each type of cell performs a particular function different from others. A liver cell, for example, does not have the same structure and biochemical duties as a brain cell, but both of them contain the same set of genes. Consequently, different groups of genes are turned on and off between the liver cell and the brain cell to produce different sets of proteins which perform different biochemical functions.

Genes vary widely in length. For human, the average size of a "typical" gene is about 27.9 kb long and contains an average of 8.8 exons. The average length of a human exon is 145 bp and 3365 bp for a human intron. To date, the largest human gene found has a length of 2.4 Mb, the largest number of exons in a human gene is 178, and the longest single human exon is approximately 17 kb [HGP01] [VAML01].

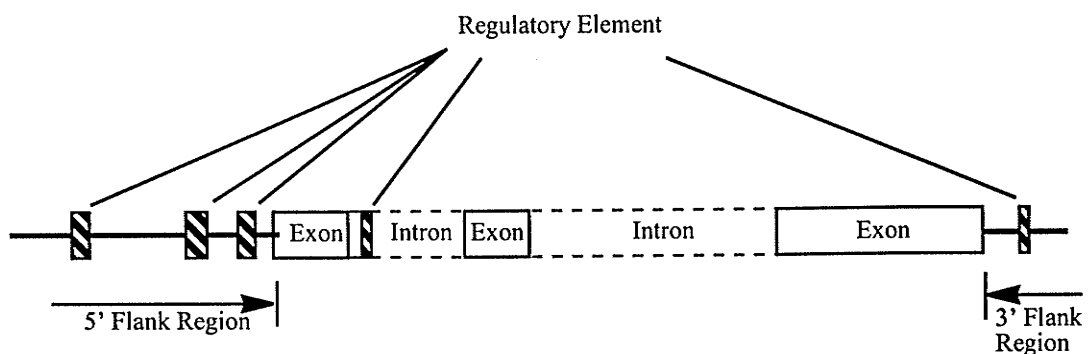


Fig. 2.2. The gene structure of eukaryotes.

### 2.1.3 Genome

A genome is all of the genetic material of an organism. It is the entire set of hereditary instructions for building, running, and maintaining the organism, as well as passing life on to the next generation. A genome consists of genes, which are packaged in chromosomes and affect specific characteristics of the organism.

As shown in Fig. 2.3, in most organisms, a genome is made of DNA and composed of intergenic regions and genes. In general, the exons of genes are called the *coding regions* of the genome while the introns and the intergenic regions are called the *non-coding regions*.

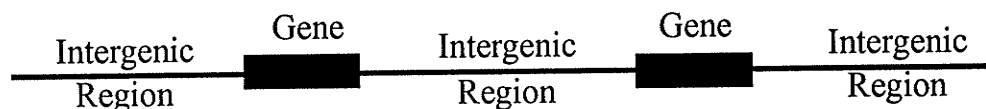


Fig. 2.3. The Structure of a higher organism genome.

#### 2.1.3.1 Chromosome

Genomes are organized into a number of physically separated parts, called *chromosomes*. Composed of DNA and protein, chromosomes are stored in the nucleus of cells. It helps a cell to keep the large amount of genetic information neat, organized, and compact. Chromosomes are the fundamental elements of inheritance, since they are passed from parent cell to daughter cell, from parent to progeny.

The number of chromosomes in an organism varies with species. In a lower organism like bacteria, the entire genome is packaged into a single chromosome. A mosquito

has six chromosomes, a sunflower 34, a goldfish 94, and a cat 38. A normal human being has 46 chromosomes, 22 pairs of autosomes and one pair of sex chromosomes.

Different chromosomes contain different genes. For example, in humans, the gene HBB encodes the  $\beta$ -globin amino-acid polypeptide, a part of the haemoglobin protein that carries oxygen in red blood cells. The gene is found in chromosome 11. The HYAL1 and HYAL2 genes, which involve glycosaminoglycan catabolism and cell migration, are located in chromosome 3.

Chromosomes can be seen under a light microscope. Stained with certain dyes, the chromosomes demonstrate a pattern of light and dark bands reflecting regional variations in the amounts of A-T and G-C. Differences in size and banding pattern allow the chromosomes to be distinguished from one another with a *karyotype* technique, a tool in the diagnosis of genetic diseases (Fig. 2.4).

Every chromosome contains a single molecule of DNA with an average of 150 million bases. The DNA molecules in a human chromosome, when stretched out to their full length, would be between 1.7 cm and 8.5 cm long, depending on the specific chromosome. But the diameter of the DNA molecules are less than a millionth of a centimetre across. If such a long, thin DNA molecule floats free in a cell, it could easily be broken up or tangled up with itself. To deal with this, the DNA molecules are folded into an orderly, compact shape in a cell by winding around protein spools and fastening into the loops, coils, and fibres of other proteins.

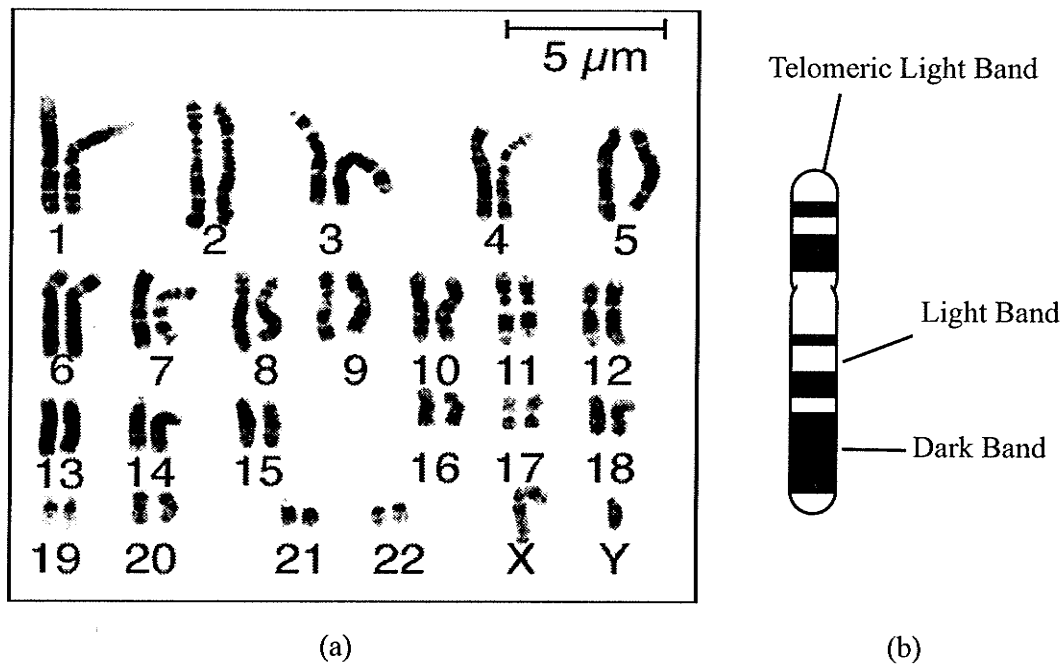


Fig. 2.4. Human karyotype and chromosome banding. (a) Microscopic examination of chromosome size and banding patterns allows medical laboratories to identify and arrange each of the 24 human different chromosomes (22 pairs of autosomes and one pair of sex chromosomes) into a karyotype, which then serves as a tool in the diagnosis of the genetic diseases. The particular individual in this case is a male because there is one X and one Y chromosome. Experimental data have shown that chromosome 22 has a higher density of genes and chromosome 21 has a lower density of genes. (b) Typical properties of bands. Light bands usually have a higher GC content (GC rich) than that of dark bands. The GC rich regions in a genome usually have a greater density of genes. Certain light bands, located adjacent to telomeres, are extremely rich in genes and have an unusually high GC content (after [NHGI01]).

In its most tightly condensed situation, a chromosome, which contains several centimetres of DNA, is only a few ten-thousandths of a centimetre long. In general, chromosomes are fully condensed only in preparation for cell division. Otherwise, some of the loops and coils are unfastened so that the DNA molecule can perform some biological processes.

In higher organisms, chromosomal DNA molecules are usually linear. As shown in Fig. 2.5, linear chromosomal DNA molecules have at least three functional units. *Telomeres* are specialized structures at the ends of the chromosome. Telomeres provide a mechanism by which the ends of the linear chromosomes can be replicated. They also stabilize the ends of the chromosomes. *Centromeres* are DNA regions necessary for precise segregation of chromosomes to daughter cells during cell division. They are the binding site for proteins that make up the skeleton, which in turn serves as the attachment site for microtubules, the cellular organelles that pull the chromosomes apart during cell division. *Replication origins* are the locations of the start of DNA synthesis. A replicating chromosome may have many active replication origins. The presence of multiple replication origins, for example, allows for complete replication of the entire human genome in only eight hours.

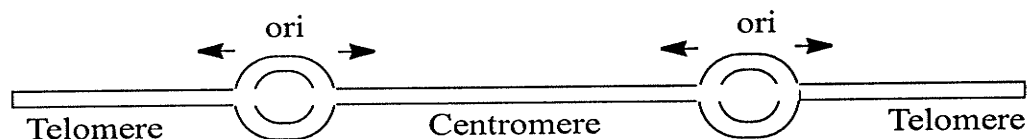


Fig. 2.5. Basic functional elements in chromosomes of higher organisms. ori, replication origin.

### 2.1.3.2 Genome Structure

The size of genomes vary widely. For example, the genome sizes of bacterium *E. coli*, yeast, fly, human, and some amphibians is  $4.6 \times 10^6$ ,  $1.2 \times 10^7$ ,  $1.0 \times 10^8$ ,  $3.0 \times 10^9$ , and  $8.0 \times 10^{11}$  bp, respectively. In humans, less than 2% of the genome are coding regions, and more than 64% of the genome are filled by intergenic DNA

[VAML01]. The genome size does not correlate well with organismal complexity. The human genome, for instance, is 200 times larger than that of the yeast *S. cerevisiae*, but 200 times smaller than that of *Amoeba dubia* [GrHe99]. This is because genomes also contain a large quantity of repetitive sequence [Hart00] [GrHe99].

The genes in mammalian genomes are not uniformly distributed over the various chromosomes. There are *gene-rich* and *gene-poor* regions in their genomes. About 20% of the entire human genome are composed of large gene-poor regions or desert. These regions have a length longer than 500 kb without a gene. The lack of genes in the gene-poor regions does not mean that they are devoid of biological function [VAML01].

In higher organisms, the GC content distribution is not even throughout the entire genomes. There are higher GC content (*GC-rich*) and lower GC content (*GC-poor*) regions. In humans, the GC content is from about 30% to over 65% with a window size of 20 kb. The GC content of the entire human genome is 38%. It has been confirmed that there are strong correlations between GC content regions and gene density. The density of genes is greater in the GC-rich regions than in the GC-poor regions. Why do GC-rich regions correlate with high gene density? One possible explanation is that a considerable fraction of the nucleotide G and C contribute to coding regions and regulatory elements [CaSm99].

For a stained metaphase chromosome, a distinct pattern of banding can be seen under microscope (Fig. 2.4). In general, dark bands correspond to the GC-poor regions and light bands are related with the GC-rich regions. It is not clear how these base composition differences can yield physically such dramatic staining differences. However, as dis-

cussed above, GC content correlates strongly with the gene density. In humans, chromosomes 17, 19, and 22, which have more light bands than the others, had the highest gene density. Conversely, chromosomes X, 4, 13, 18, and Y, which correspond to the few light bands, had the lowest gene density. Certain light bands, located adjacent to telomeres, are extremely rich in genes and have an unusually high GC content. An example is the Huntington's disease region at the tip of the short arm of human chromosome 4 [HGP01] [VAML01].

#### 2.1.4 DNA Organization

As shown previously, the DNA molecules in higher organisms are very thin, about 2 nm in diameter (Fig. 2.1), but are very long in length. Human genome, if lined up, is nearly 1 m long. Hence, the cell faces an enormous packaging problem. The DNA molecules not only have to fit into the cell but must be packaged properly so the information contained in the DNA molecules can be accessed efficiently by other biological molecules.

In higher organisms, only a small portion of a genome is associated with some ongoing biological processes and therefore, are unpacked. In humans, over 50% of the genome are *repetitive sequences*, called "junk" DNA, which contain no functional information and do not relate with the biological processing in the cells. Moreover, in a multicellular organism with complex developmental regulatory schemes, there are large portions of genetic information that are not used in particular cell types. Certain genes may be utilized only within one short period during development. Hence, eukaryotic cells have mechanisms for packaging regions of their chromosomes into configurations that do not become involved in other biological process in cells.

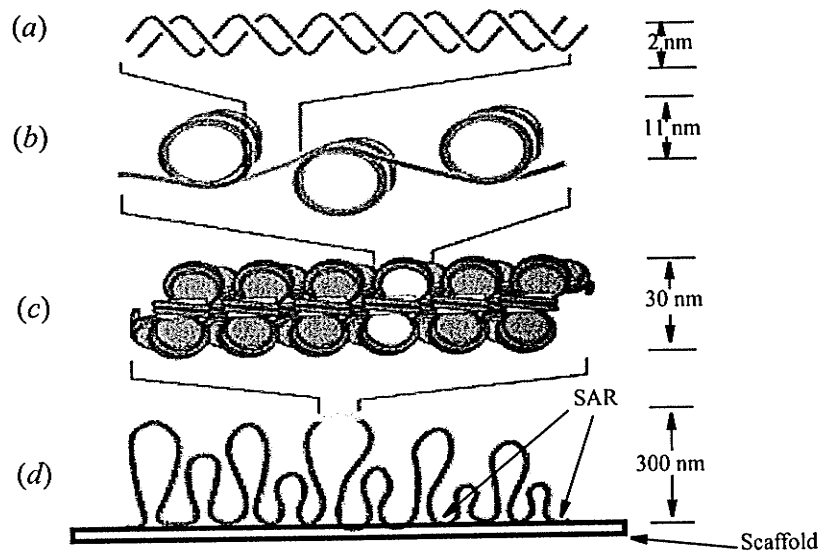


Fig. 2.6. The packaging of DNA in eukaryotic cells. (a) short region of DNA double helix; (b) section of 11-nm filament composed of nucleosomes; (c) section of a 30-nm fiber of packed nucleosomes, and (d) section of chromosome loops anchored to a protein scaffold. The scaffold is not straight. The attachment region called SAR.

In eukaryotic cells, the first order of DNA packing in chromosomes is the formation of a string of nucleosomes along the DNA. A *nucleosome* is a coiled structure of about 145 bp long DNA wrapped into two left-handed coils around a histone octamer (Fig. 2.6). DNA is not randomly wrapped into nucleosomes. Only some regions of the DNA helix are associated with nucleosomes [ArMo93]. Nucleosomes organize themselves together to form a filament of about 11 nm in diameter. The 11-nm filament is coiled upon itself to make a thicker solenoidal structure of fiber with a diameter of 30 nm. Stretches of solenoid containing on an average of 50-100 kb long DNA are attached to a protein scaffold. About 200 bp long with the repeats of the sequence AATATATTT, the specific AT-rich regions of DNA are associated with the regions in the nuclear matrix, called SAR (*scaffold attachment region*). The highest order of chromosome packing occurs in met-



aphase. During this period, the metaphase chromosomes appear to consist of stacks of packed 30-nm fiber loops (Fig. 2.7) [Sind94].

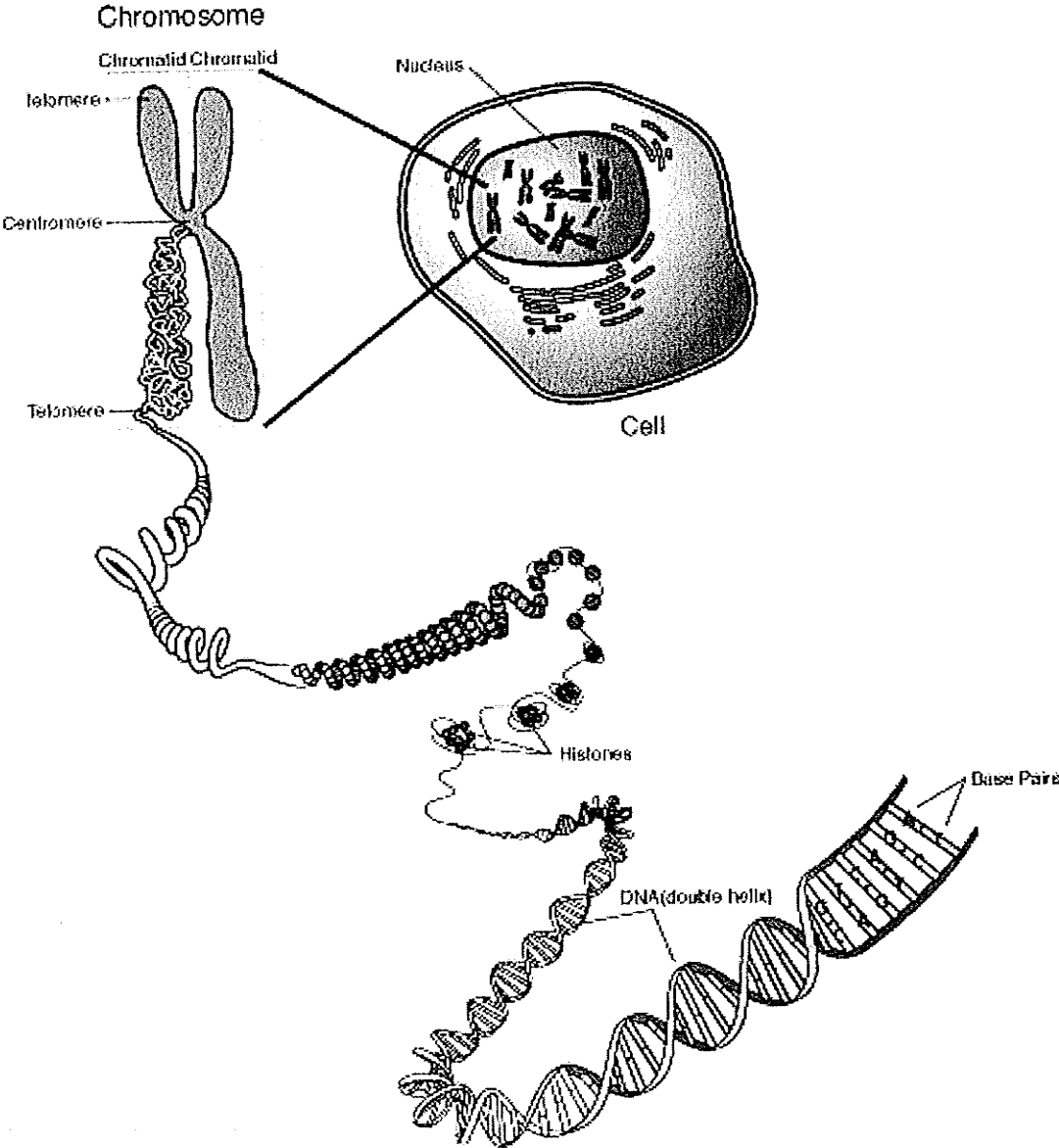


Fig. 2.7. The hierarchy of chromosomal structure in the metaphase (from [NHGI01]).

The higher-order structures of chromosomes pose an extraordinary challenge for structural biology society because they are so complex and the structures are so large. It would greatly benefit our life if we completely uncover the complicated structure and the sophisticated mechanisms that allow DNA packing and unpacking to be used to modulate DNA function.

## 2.2 Functional Genomics

For its survival, an organism must encode and store all the instructions needed to build, operate, maintain, and reproduce itself and to respond to varied environmental conditions. The organism also has to read out the instructions of its genome in the proper order, time, and amount for each gene product. The overall aim of functional genomics is to understand how an organism deals with these issues.

### 2.2.1 RNA

*RNA* (ribonucleic acid) is the direct molecular instruction for the synthesis of a specific protein. It is a single-stranded polynucleotide molecule and is made by transcription from a DNA template. Like DNA, RNA consists of three parts: a sugar residue, a nitrogenous base, and a phosphate group. The RNA molecules are composed of adenine, guanine, cytosine, and *uracil* (U) instead of thymine. Most of the organisms have DNA as their genetic materials. However, some bacterial viruses, some animal viruses, and some plant viruses have RNA as their genetic materials.

There are four major classes of RNA molecules or transcripts, *messenger RNA* (mRNA), *transfer RNA* (tRNA), *ribosomal RNA* (rRNA), and *small nuclear RNA*

(snRNA). The mRNAs, tRNAs, and rRNAs are found in both prokaryotes and eukaryotes, while snRNAs are found only in eukaryotes. The mRNA molecules carry the genetic information out of the nucleus for protein synthesis (Fig 2.8). Together with the numerous ribosomal proteins, the rRNA molecules are assembled to form the ribosomes. The ribosomes are cellular organelles involved in protein synthesis. The specific tRNA molecules bring the specific amino acids to the ribosomes and recognize the specific encoded sequences of the mRNAs to allow correct protein synthesis. The snRNAs are involved in processing of mRNA precursor in eukaryotes.

Table 2.1. Genetic code and human codon usage.

First letter	Second letter												Third letter
	U			C			A			G			
U	17.0	UUU	Phe	14.8	UCU	Ser	12.1	UAU	Tyr	10.0	UGU	Cys	U
	20.5	UUC		17.5	UCC		15.8	UAC		12.3	UGC		C
	7.3	UUA	Leu	11.9	UCA		0.7	UAA	Stop	1.3	UGA	Stop	A
	12.5	UUG		4.5	UCG		0.5	UAG		12.9	UGG		Trp
C	12.8	CUU	Leu	17.3	CCU	Pro	10.5	CAU	His	4.6	CGU	Arg	U
	19.3	CUC		20.0	CCC		14.9	CAC		10.8	CGC		C
	7.0	CUA		16.7	CCA		12.0	CAA	6.3	CGA	A		
	39.7	CUG		7.0	CCG		34.5	CAG	11.6	CGG	G		
A	15.8	AUU	Ile	12.9	ACU	Thr	17.0	AAU	Asn	12.1	AGU	Ser	U
	21.6	AUC		19.3	ACC		19.8	AAC		19.3	AGC		C
	7.2	AUA	14.9	ACA	24.0		AAA	11.5	AGA	Arg	A		
	22.3	AUG	Met	6.3	ACG		32.6	AAG	11.3		AGG	G	
G	10.9	GUU	Val	18.5	GCU	Ala	22.4	GAU	Asp	10.8	GGU	Gly	U
	14.6	GUC		28.3	GCC		26.1	GAC		22.7	GGC		C
	7.0	GUA		15.9	GCA		29.1	GAA	16.4	GGA	A		
	28.8	GUG		7.5	GCG		40.2	GAG	16.4	GGG	G		

The numbers represent the occurrence of specific codons per thousand codons (based on 12,816,923 codons, 27,143 coding DNA sequences).

## 2.2.2 Protein

Proteins do most of the work in a cell. Instead of just four nucleotides, proteins are the chains of 20 different amino acids held together. A linear chain of amino acids is called a *polypeptide*. A protein is one or more properly folded polypeptide complex. Every three-base mRNA sequence (a triplet), called a *codon*, specifies an amino acid in a polypeptide chain. As shown in Table 2.1, the codons are degenerate since there are  $4^3 = 64$  possible codons encoding for only 20 amino acids and one stop signal. The genetic code (each mRNA codon and its corresponding amino acid or stop signal) is nearly universal for all forms of life.

## 2.2.3 Gene Expression

### 2.2.3.1 Transcription Process

*Transcription* is the mechanism by which a template strand of DNA is utilized by specific proteins, called *RNA polymerases*, to generate an RNA chain. The process is initialized by an RNA polymerase, with the help of several proteins called *initiation factors*, recognizes and binds to a specific region called *promoter* upstream of a gene on the DNA. The double-stranded DNA then unwinds in the promoter region. For mRNA genes in particular, a number of regulatory proteins participate in initiation by indicating which protein-encoding genes are to be copied. Specific *regulatory proteins* are located at specific regulatory elements in the DNA molecules. Of the unwounded double-stranded DNA, only the one containing the correct promoter sequence acts as the template and the RNA polymerase then synthesizes RNA by an orderly copying of the DNA template into a RNA chain, using four nucleotides A, G, C, and U. The template is not always the same chain of

the double-stranded DNA. Different genes may have their template chains on either chain of the double-stranded DNA. However, for a given gene, the template chain remains the same within the boundaries of a gene.

Unlike bacterial RNAs, the eukaryotic RNAs undergo *post-transcriptional* processes. After adding a 5' end cap and a 3' end poly(A) tail to it, the resulting RNA chain is called *precursor mRNA* (pre-mRNA). The sequences correlated with the intronic DNA must be removed from the primary transcript prior to the RNAs being biologically active. The process dealing with intron removal is called *RNA splicing*. The illustration of RNA processing in eukaryotes, using  $\beta$ -globin gene as an example, is shown in Fig. 2.8.

As discussed earlier, a large number of nucleotides incorporated in the primary transcript as introns are removed later in the splicing process. Energy is utilized in the synthesis of the primary transcript and the splicing processing. At first glance, it seems that the presence of introns in eukaryotic genes is an extreme waste of cellular energy. However, the presence of introns can protect the genetic damage by environmental influences. Another function of introns is to allow alternative splicing to occur. By altering the pattern of exon organization, from a single primary transcript, different proteins can arise from the processed mRNA from a single gene. Therefore, this allows an increase in the number of proteins without increasing the overall number of genes.

A mRNA is a template for a specific protein. Studies on the mRNA reveal the amino acid information of a protein. In practice, scientists study the DNA copies, which are the copies from the mRNAs, instead of the mRNAs itself, since the DNA copies are more stable than the mRNAs. The DNA copies are called *complementary DNA* (cDNA).

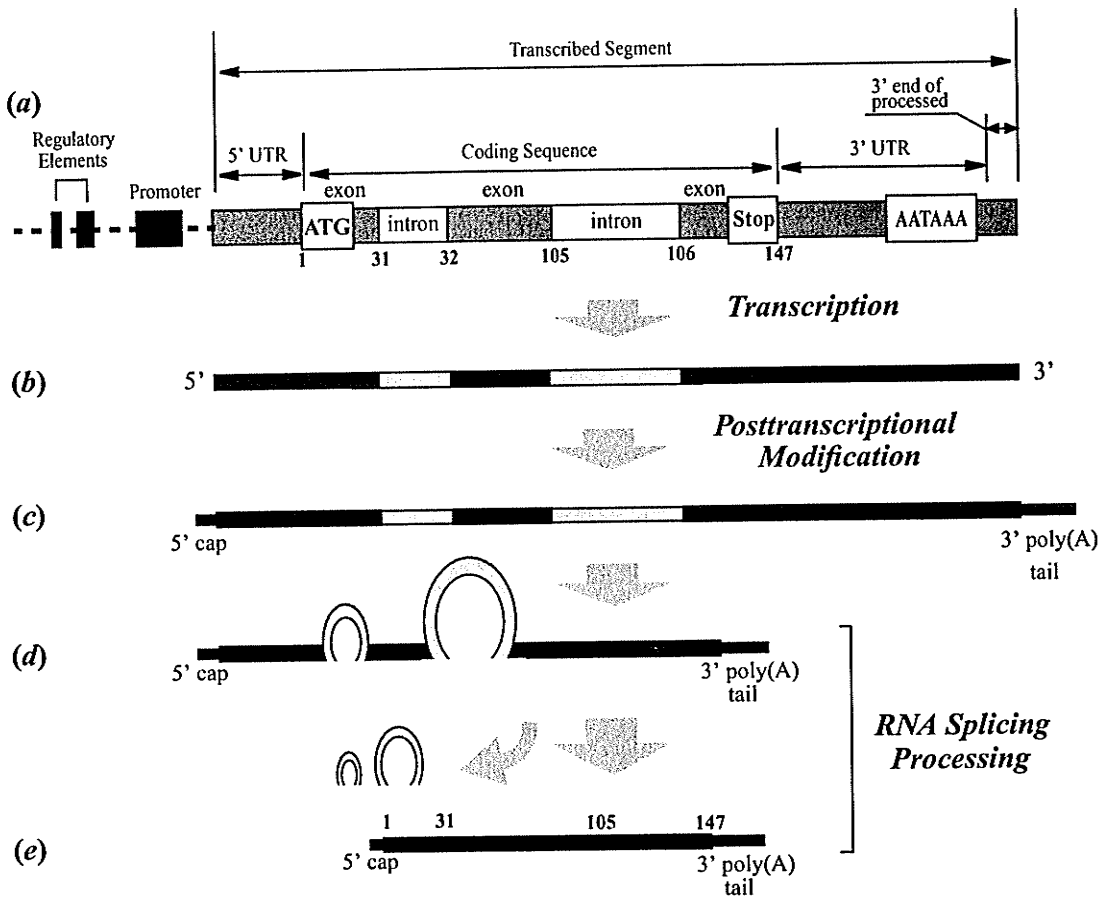


Fig. 2.8. Overview of RNA processing in eukaryotes, using  $\beta$ -globin gene as an example. The  $\beta$ -globin gene contains three exons and two introns. After addition of the 5' cap and 3' poly(A) tail, the introns are removed during splicing process. The small numbers refer to positions in the sequence of  $\beta$ -globin, which contains 147 amino acids. (a)  $\beta$ -globin genomic DNA, (b) primary RNA transcript, (c)  $\beta$ -globin pre-mRNA, (d) undergoing splicing, and (e) mature mRNA.

### 2.2.3.2 Translation Process

*Translations* are the processes that produce proteins based on the information embedded in mRNA sequences. Translation is carried out by ribosomes.

As described in Table 2.1, each three-base nucleotide sequence, or codon, in the

mRNA encodes a specific amino acid. Because there are three nucleotides in each codon, an mRNA can be translated in three different *reading frames* in each region (Fig. 2.9). These reading frames shift forward or backward from one another by one base. Only a part of one out of the three reading frames, called *open reading frame (ORF)*, represents the corresponding protein sequence.

A translation usually begins at a AUG initiator codon in an ORF, which serves as the template for protein synthesis. Once translation has started, the ribosome, moves three bases at a time along the mRNA. The translation is stopped when the ribosome reaches a terminating codon, or stop codon.

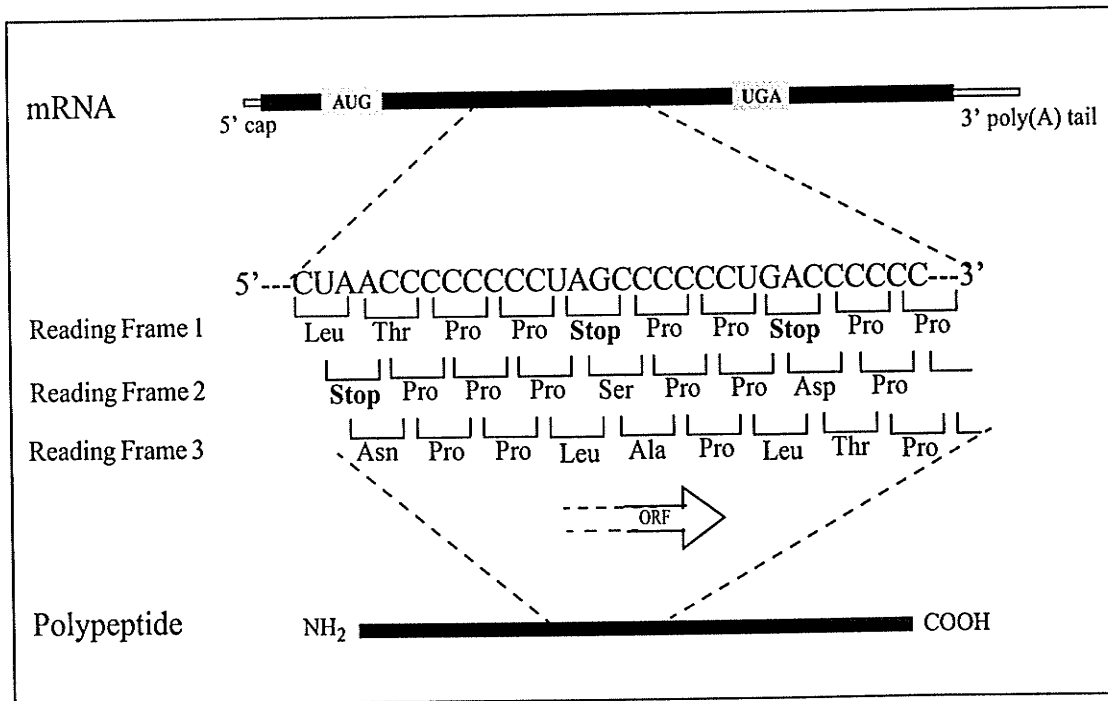


Fig. 2.9. Reading frames and mRNA. There are three reading frames correlated with an mRNA. They are inferred by shifting one base forward or backward from the mRNA sequence. Only a portion of one of the three reading frames represents the corresponding protein sequence.

Figure 2.9 also shows a part of an open reading frame (ORF). An ORF represents a potential amino acid sequence with an initiator codon and a chain terminating codon for a possible protein-coding region in a DNA sequence. This sequence is not necessary in nature but just a string of codons for amino acids that is not broken by stop codons. The ORF is often used by computer programs for possible protein-coding region seeking, homology searching, and restriction site locating.

### 2.3 Codon Usage

The unequal usage of codons in the coding regions appears to be a universal feature of the genomes across species. The bias is mainly due to the uneven usage of the amino acids in the existing proteins and the uneven usage of synonymous codons [GGGP80].

*Codon usage* can be strongly biased in different species. For example, six different codons specify the amino acid leucine (UUA, UUG, CUU, CUC, CUA, and CUG), but 60 percent of the leucine codons in bacteria are CUG, and 80 percent are UUC in yeast [CUTG01]. Table 2.1 shows the human codon usage based on the human coding sequences (12,816,923 codons, 27,143 DNA coding sequences) published before June 2001 [CUTG01].

The bias of codon usage is associated with a wide variety of factors including tRNA abundance [Ikem85] [Bulm87], gene expression level [Holm86], local compositional biased [KaMr96], protein composition and structure [DMAB91] [GMBG00], translation optimization [Xia98], gene length [Eyre96], and mRNA secondary structure



[HuKH92]. Studies have also shown that the genome GC content is correlated with cross-species differences in frequencies of codons [OOYU88] and amino acids [Lobr97] [WiVa99]. Further research by Knight *et al.* [KnFL01] demonstrates that it is the genome GC content which drives codon usage. These studies suggest that genes and genomes at mutation/selection equilibrium reproduce a unique relationship between nucleic acid and protein composition or in other words the structure of the genome determines the codon usage in the organism [KnFL01].

## 2.4 Computational Analysis of DNA Sequences

Biological computing has a long history in structural molecular biology. The recent boom is mainly a consequence of the sequencing of the human and other genomes. These projects yield an enormous amount of DNA and protein sequence data with an exponential growing rate. It has been estimated that, for example, new sequences of approximately 11 billion DNA bases were published in 2000, three times as many as that in 1999 [NCBI01]. This amount of data is shifting research in molecular biology and genetics from a purely experimental approach to the one in which experiments can be planned in front of a computer. *Bioinformatics* is the field as a consequence of dealing with this by the combination of information science and biology.

One of the most active areas in bioinformatics involves the analysis of DNA sequence information. The interests of DNA analysis are mainly in gene prediction in the various organisms and characterization of the higher structure of the genomes.

### 2.4.1 Gene Prediction

Gene prediction faces a number of challenges. First of all, genes of the most eukaryotic organisms are neither continuous nor contiguous. They are separated by a large size of the intergenic DNAs. Their exons are further interrupted by introns. Second, the coding regions of a higher organism occupy only a small fraction of its entire genome. The length of the exons varies widely and sometimes is even as small as 10 to 20 bp. As a result of this, a trivial exon signal may be submerged into the non-coding pool of a genome. Third, the arrangement of genes in a genome makes things even more complicated by some exceptions. There are examples of genes nested within each other, such as one gene located in an intron of another gene or overlapping genes on the same or opposite DNA strands [DSRC99].

Two classes of computational approach are used commonly for gene prediction in genomic sequences. Sequence similarity search, such as the BLAST family of programs, is a well-established computational approach for gene prediction which has been used extensively with considerable success [Fick96]. It is used to detect sequence similarity between an uncharacterized sequence of interest and the known sequences of the genes, the proteins, or the mRNAs. It suggests that they are homologous and share common evolutionary origin if there is a significant similarity shown between them. Therefore, the information from the known sequences can be used to infer a gene structure or function of the uncharacterized sequence.

The approach of sequence similarity search has limitations although it has been proven useful in many cases. It has been shown that only a fraction of newly discovered

sequences have identifiable homologies in the current databases [Clav97] [WAAB94] [DSRC99]. Therefore, this approach will give little or no useful information for the uncharacterized sequences which have no homology with the known genes.

The other computational approach, called *ab initio* prediction, integrates coding statistics and sequence signal detection into one framework. Coding statistics are different between coding and non-coding regions. A number of measures, including codon [StMc82], hexamer [ClSB90], amino acid usage [FiTu92], position asymmetry [FiTu92], codon preference [GrDB84], dependence between nucleotide positions [BoMc93], mutual information [HeGr95], entropy [SnSt93], and Fourier analysis [TRBR97], have been introduced to coding statistics [Clav97]. Most of gene prediction programs integrate the output of a number of coding statistics.

Sequence signal detection methods attempt to recognize the genes by following the interaction of the gene expression machinery with the nucleic acid. Sequence signals, usually only several base-long subsequences, are recognized by the cell machinery and are the signals for certain processes. The signals that are modeled by current gene prediction programs are promoter elements, start and stop codons, splice sites, and poly(A) tail sites.

Many pattern recognition techniques are used for detecting sequence signals and integrating several coding statistics. For example, the popular GrailEXP program uses a neural network [UbMu91]. FGENEH program uses linear discriminant analysis [SoSL94]. Dong and Searls [DoSe94] in GENLANG use linguistic methods. Decision tree is used in SORFIND [HuHa92] and dynamic programming is used in GENEPARSER [SnSt93]. Currently more powerful programs are entirely built with hidden Markov mod-

els (GENSCAN [BuKa97], GENIE [KHRE96], GENEMARK.HMM [LuBo98], HMMGENE [Krog97], and VEIL [HeSF97]). Both codon statistics and signal detection models need to be trained by training sets.

Although they have a remarkable success for exon prediction, the accuracy of the current gene prediction programs remains rather low when facing the large anonymous sequences generated by HGP. For instance, only 20% of annotated genes have all exons predicted exactly for the human chromosome 22 [DSRC99]. The predicting accuracy of exon prediction is also dependent on the length of the exons. The accuracy decreased for the exons longer than 200 bp or shorter than 70 bp. It still remains a problem for nested and overlapping genes or alternative splicing. Most of the programs can not deal with the case of multiple genes or partial gene in a sequence [YeLB01].

There are fundamental limitations for this class of approaches. First of all, high accuracy prediction should not only predict genes positively where there are genes, but predict genes negatively where there is no gene. The current data of training sets are originally from the public databases. These sequences are characterized by anomalously high gene density, since the regions containing the genes have been preferentially sequenced and published [Guig97]. The gene density of the popular ALLSEQ data set, for example, is 15%, while the gene density of the human genome is less than 1.5% averagely. Most of the programs even give a positive prediction for 12-36% of random generated DNA sequences, although they demonstrated a good success on the known sequences in the public databases [Guig97].

The second limitation is that these programs introduce a bias in favor of the detec-

tion of genes similar to those known sequences [Clav00]. Hence, these programs, which are developed based on the atypical gene sample available in current databases, may not perform as well on genes more typical of the biological universe as a whole. In other words, these programs may not predict successfully for the sequences containing the genes that are not similar or have no similar property to the known genes. The typical example of this is that these programs fail in prediction of non-protein coding genes (such as Xist and H19) [Clav00].

#### 2.4.2 Complexity Analysis of DNA Sequence

In the past decade or so the search for the statistical features of the genomic DNA sequences is an area with rapid growth in the amount of the obtained results as well as in the range of their functional and evolutionary implications for organisms. The development of this area is directly influenced by the exponential growth of the data in DNA and protein sequences.

During the past few years, there has been intense discussion about the existence and the nature of long-range correlations in DNA sequences. The correlation properties of DNA sequences was first studied by Peng *et al.* [PGHS92] in 1992. In their publication, they have demonstrated a long-range correlation in the non-coding regions but not in the coding regions of higher eukaryotic DNA sequences, using the Lévy walk method to map the DNA alphabet sequences into numerical sequences. These conclusions were supported by examining the size distribution of purine and pyrimidine clusters in pure coding and non-coding regions of different organisms [MBGS95] [PrA197]. Others have reported that both coding and non-coding regions of the DNA sequences present long-range correlation

[ChLa93] [PrCl92]. Voss [Voss92], based on the self-similar characteristics in DNA sequences, showed that DNA sequences exhibit a power-law relationship with  $1/f$  (pink) noise behaviour in DNA sequences. The author also noted a strong periodicity at a frequency of three bp in DNA sequences. Yu *et al.* [YuAW01] [YuAn01] proposed a time series model based on the global structure of the complete genome, and have shown long-range correlations in the bacteria DNA sequences. Further studies by Audit *et al.* [ATVA01] suggested that the long-range correlations, observed in both coding and non-coding regions, are the signature of the higher-order structural organization of chromatin. The presence of small-scale correlations only in eukaryotic genomes are related to the machinery underlying the wrapping of DNA in the nucleosomal structure.

A major problem of these analysis algorithms applied to DNA sequences is that errors are introduced by giving artificial values to the sequence at each base pair position. When mapping DNA alphabet sequences to numerical sequences, most of the algorithms published use the Lévy walk or modified Lévy walk models [PBGS92]. Briefly, to map a DNA sequence using the Lévy walk model, a walker either descends or rises one step at the position  $i$  along a DNA sequence chain if a pyrimidine (C/T) or a purine (A/G) occurs, respectively.

To avoid this and the critical limitation of “prior training” problems, entropy and mutual information algorithms are used for measuring the complexity of DNA sequences. Ebeling *et al.* [EMKS00] reported that correlations on many scales, including those of long range, are found in the DNA sequences. Using the average mutual information (AMI), Grosse *et al.* [GHBS00] showed that the probability distribution functions of the

AMI are significantly different in coding and non-coding regions of DNA sequences and there are persistent period-three oscillations of AMI functions for the coding sequences. By calculating Rényi dimension through Rényi entropy, Rifaat proved that the genomic DNA sequences demonstrate species independent multifractality [Rifa98].

## **2.5 Summary**

This chapter presented the background of the molecular biology and genomics involved in this thesis. Codon statistics was discussed. The bioinformatics techniques and analysis methods for DNA sequences were outlined. The next chapter will provide a background on fractal and chaos since the goal of this thesis is characterization of DNA sequences through multifractal analysis.

## CHAPTER III

### BACKGROUND ON FRACTALS AND CHAOS

This chapter presents the background on fractals and chaos involved in this thesis. The basic concepts of fractal, fractal dimensions, as well as some typical fractal sets and their properties are first described. Multifractal and multifractal dimensions are discussed with more detail. Chaotic dynamics and strange attractor are then introduced. In the last part, the methods of attractor reconstruction are described.

#### 3.1 Fractals

##### 3.1.1 Fractal Sets

The concept of *fractals* was first introduced by Mandelbrot for describing complex objects which are difficult to deal with by topological geometry. Since then, fractals have been studied extensively in physics and mathematics. A fractal object is self-similar, which means that its parts are similar, in some extent, to itself as a whole. In other words, a fractal has self-similarity in the structure and complexity at all scales.

The Cantor set [Cant83] is one of the mathematically self-similar fractals. It is composed of an infinite set of points. These points distribute uncontiguously on a one-dimensional line. As illustrated in Fig. 3.1, the Cantor set is built by beginning with a straight line as an initiator. The middle one third segment of the initiator is then eliminated, resulting in the generator. The process is repeated on each truncated line segments. As the process iterated infinitely, a set of infinite points but not line segments are left. The set of these infinite points is called the Cantor set. Clearly, as we can see, the Cantor set is



not a line segment although the points of the set can fill up a line. It is an object with a self-similar structure at any scale.






		Length of Segment	Number of Segment
Initiator		$L$	1
Generator: Step 1		$(1/3)^1 L$	$2^1$
Step 2		$(1/3)^2 L$	$2^2$
Step 3		$(1/3)^3 L$	$2^3$
Step 4		$(1/3)^4 L$	$2^4$
	⋮		⋮

Fig. 3.1. Generation of the Cantor set.  $L$  is the length of the initiator.

### 3.1.2 Fractal Dimensions

Fractal dimensions measure the degree of complexity (or roughness, brokenness, and irregularity) of an object. The introduction of fractal dimension is motivated by the fact that it is not sufficient to describe the complexity of fractal objects with the use of traditional geometry. That the fractal dimension strictly exceeds the topological dimension is the essential feature for all fractals [Mand83].

Let's consider again the example of the Cantor set in Fig. 3.1. The length of the each line segment, in the unit of the original initiator, at each iterated step is  $(1/3)^1$ ,  $(1/3)^2$ ,  $(1/3)^3$ , and  $(1/3)^n$  (at step  $n$ ), respectively. The number of the line segments is  $2^1$ ,  $2^2$ ,  $2^3$ , and  $2^n$  (at step  $n$ ), respectively. Hence, the total length of the Cantor set at each iteration in

the unit of the original initiator is  $(2/3)^1$ ,  $(2/3)^2$ ,  $(2/3)^3$ , and  $(2/3)^n$  (at step  $n$ ). When the generation of the Cantor set continues to infinite iteration, the total length of the Cantor set is  $(2/3)^\infty = 0$  and the total number of the line segments, with a length of 0, is  $2^\infty = \infty$ . It is clear that the Cantor set can not be measured with the traditional dimension since it is not a line, a point, or any other curves which can be described in normal one, two, or three dimensions.

However, fractal dimensions, which take into account the relationship between the scale and some measurements, can be used to describe such complex objects. As shown in Fig. 3.1, the length of each self-similar line segment,  $r$ , and the number,  $N$ , of self-similar segments are  $(1/3)^n$  and  $2^n$  at iteration  $n$ , respectively. The  $D_s$ , called self-similarity dimension, is

$$D_s = \frac{\log(N)}{\log(1/r)} = \frac{\log(2^n)}{\log(1/(1/3)^n)} = \frac{\log(2)}{\log(3)} \approx 0.6309 \quad (3.1)$$

Thus the  $D_s$  of Cantor set is approximately 0.63 and is invariant no matter how many iterations have passed. This also shows that the notion of traditional integer dimensions needs to be expanded to fractal dimensions to describe fractal objects such as the Cantor set.

There are many distinct definitions of fractal dimensions in order to reflect the different properties of the self-similar and self-affine objects. The fractal dimensions can be catalogued into four basic classes: morphological, entropy, spectral, and variance dimensions [Kins94].

The Hausdorff dimension,  $D_H$ , is widely used since it is easy to understand and compute. Briefly, the object to be measured is covered by a set of volume elements (vels) with a vel size of  $r$  (called scale factor). The number of vels,  $N(r)$ , which covers the object completely is taken as the measurement for the power-law relationship,

$$N(r) \sim r^{-D_H} \quad (3.2)$$

Thus, the hausdorff dimension becomes

$$D_H = \lim_{r \rightarrow 0} \frac{\log N(r)}{\log(1/r)} \quad (3.3)$$

The Hausdorff dimension is one of morphological dimensions [Kins94]. The limitation of this dimension is that it does not provide detailed information related to the non-uniform property of a fractal but an estimation of the outline complexity of the fractal.

The information dimension,  $D_I$ , avoids this problem by taking into account the information contained in a fractal object as well as the geometrical properties of the object.

The well known Shannon entropy is the amount of the information for specifying a state of a system at a certain resolution  $r$ . It is defined as

$$H_s = - \sum_{i=1}^{N(r)} p_i \log p_i \quad (3.4)$$

where  $N(r)$  is the number of vels and  $r$  is the vels size, a scale factor. The  $p_i$  is the relative frequency  $n_i$  with which the object intersects the  $i$ th vel of the total number  $N$  of intersects of the fractal with all the vels

$$p_i = \lim_{N \rightarrow \infty} \frac{n_i}{N} \quad (3.5)$$

where

$$N = \sum_{i=1}^{N(r)} n_i \quad (3.6)$$

Hence, the information dimension,  $D_I$ , is defined as

$$D_I = \lim_{r \rightarrow 0} \frac{-\sum_{i=1}^{N(r)} p_i \log p_i}{\log(1/r)} \quad (3.7)$$

A time series can be transformed into its power spectrum, using Fourier, wavelet, or other spectral analysis techniques [Kins91]. The power spectrum  $P(f)$  of the relative frequency interval  $f$  between successive notes (seminotes) can be approximated by a homogeneous power function with an exponent  $\beta$  [Kins94], i.e.

$$P(f) = cf^\beta \quad (3.8)$$

where  $c$  is a constant value and white noise has a  $\beta$  value of 0, pink noise has a  $\beta$  value of -1, brown noise has a  $\beta$  value of -2, and for black noise,  $\beta$  is -3.

The spectral dimension,  $D_\beta$ , is then defined as

$$D_\beta = E + \frac{3 - \beta}{2} \quad (3.9)$$

where  $E$  is embedding Euclidean dimension. The spectral dimension analysis of a complex physiological signal, such as the heart rate (ECG), brain waves (EEG), muscle waves

(EMG), blood pressure, and gastric noises, may reveal a better understanding of the underlying process responsible for the signal.

The self-similarity, Hausdorff, information, and spectral dimensions are single dimension. There has been an argument that a single dimension only reflects an aspect of a fractal object [Chen97]. A multifractal spectrum is more appropriate for describing fractal objects since it reveals more information of a fractal object [Kins94].

## 3.2 Multifractal Dimensions

### 3.2.1 Rényi Generalized Entropy and Dimension Spectrum

Knowing the limitation of the Shannon entropy for describing the non-uniform distribution in the fractal sets, Alfréd Rényi introduced the concept of the Rényi entropy in 1955. Rényi entropy,  $H_q$ , is a generalized form of the Shannon entropy [Rény55]. It is defined as

$$H_q = \frac{1}{q-1} \log \sum_{j=1}^{N(r)} p_j^q \quad -\infty < q < \infty \quad (3.10)$$

where  $q$  is the moment-order. Hence, according to the Eq. 3.7, the Rényi dimension,  $D_q$ , is given by

$$D_q = \lim_{r \rightarrow 0} \frac{H_q}{\log(r)} = \lim_{r \rightarrow 0} \frac{1}{q-1} \frac{\log \left( \sum_{j=1}^{N(r)} p_j^q \right)}{\log(r)} \quad (3.11)$$

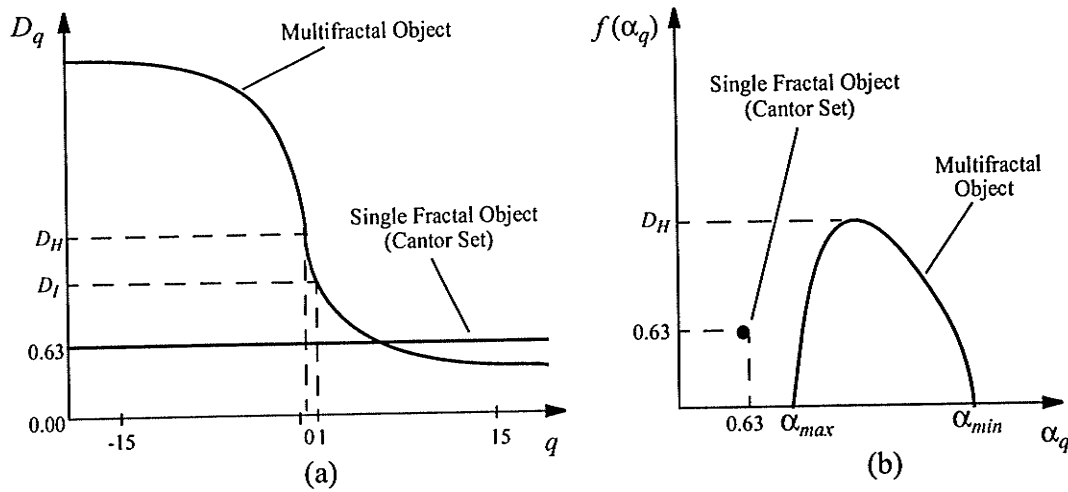


Fig. 3.2. The illustration of multifractal spectrum for a single fractal, the Cantor set, and a multifractal object. (a) Rényi dimension spectrum, (b) Mandelbrot dimension spectrum.

As illustrated in Fig. 3.2, the Rényi dimension is a monotonic and nonincreasing function of the moment order  $q$ . For the case of  $q = 0$ , the Rényi dimension is equivalent to the Hausdorff dimension  $D_H$ , and for  $q = 1$ , the Rényi dimension reduces to information dimension  $D_I$ . Thus, the Rényi dimension covers the Hausdorff dimension, the information dimension, and many other fractal dimensions as special cases.

For strictly self-similar fractal objects such as the Cantor set, the fractal dimension is single-valued (0.63 for Cantor set) for all values of  $q$  and is called *single fractal*. If there are multiple corresponding values of Rényi dimension for an object, the fractal object is called *multifractal*. The multiple-value of the Rényi dimension is mainly due to the non-uniform distribution of the multifractals. The difference between dimensions of different order  $q$  measures the degree of inhomogeneity of a multifractal in the sense of whether its different subsets are visited with equal frequency. The Rényi dimension is a multifractal

dimension measure. The spread of the Rényi dimension spectrum indicates the complexity of the fractal. The multifractal dimension measure is a very useful tool for complex fractal objects since it provides an infinite number of different (and relevant) dimensions for describing a fractal object.

### 3.2.2 Mandelbrot Spectrum

Another multifractal dimension measure is the Mandelbrot dimension. In single fractal dimension measure, a power-law relationship

$$p \sim r^D \quad (3.12)$$

can be obtained between the distribution of the probabilities  $p$  with a single vels covering and the size  $r$  for a homologous fractal.

For a fractal with an inhomogenous probabilities  $p_j$ , it is useful to consider a nonuniform set of vels with size  $r_j$ . The local power-law relationship between the  $p_j$  in  $j$ th vel and the  $r_j$  follows

$$p_j(r_j) \sim r_j^{\alpha_j} \quad (3.13)$$

where  $\alpha_j$  ( $j=1, 2, \dots, N(r)$ ) is a noninteger, called the Hölder exponent [Kins94]. The Hölder exponent depends on the selected region of the measure. For a particular nonuniform fractal object with inhomogenous measures, the Hölder exponent varies within a limited range  $\alpha \in [\alpha_{min}, \alpha_{max}]$ . The value of the range represents the complexity of nonuniformity.

A power-law relationship exists between the number of the vels  $N_\alpha(r)$  and a particular  $\alpha$  as follow

$$N_{\alpha}(r) \sim r^{-f(\alpha)} \quad (3.14)$$

where  $f(\alpha)$  is the Mandelbrot dimension at the specific  $\alpha$  subset.

It has been shown [HJKS86] [AtSV88] [Kins94] that the Mandelbrot dimension  $f(\alpha)$  and the Hölder exponent  $\alpha$  are related with the Rényi dimension  $D_q$  by

$$\alpha_q = \frac{d}{dq}[(q-1)D_q] \quad (3.15)$$

and

$$f(\alpha_q) = q\alpha_q - (q-1)D_q \quad (3.16)$$

The Mandelbrot dimension  $f(\alpha)$ , also called  $D_{Man}$ , provides an object with an alternative perspective view of multifractality. An example of Mandelbrot, *i.e.*  $f(\alpha)$  versus  $\alpha$ , is shown in Fig. 3.2. It is noticed that a single fractal object such as the Cantor set reduces to a point in the Mandelbrot spectrum since it has single-value of fractal dimension for all values of  $q$ .

### 3.3 Chaos and Strange Attractors

#### 3.3.1 Chaotic Dynamical Systems

Most natural phenomena can be categorized as nonlinear dynamical systems. Nonlinear dynamical systems can be summed up into three classes: stable, unstable, and chaotic [JoSm87]. Stable dynamical systems settle either into a periodic motion or a steady state after some transient period. Unstable dynamical systems are aperiodic and unbounded.



However, the chaotic systems are neither stable nor unstable but appear as random behaviour [Kins95]. They are extremely sensitive to the initial conditions and thus are unpredictable on their long-term behaviour. On the other hand, the chaotic systems are deterministic and their behaviour has a sense of order and pattern with limited boundary.

As far back as in the nineteenth century, the aspect of sensitivity to initial conditions and long-term unpredictability in chaos had been studied [HuYo93]. The keen interest in this area is largely due to the Lorenz's works in weather prediction [Lore63]. The concept of chaos in regard to deterministic nonlinear behaviour was first introduced by Li and Yorke [LiYo75]. Since then, chaos theory has dramatically developed.

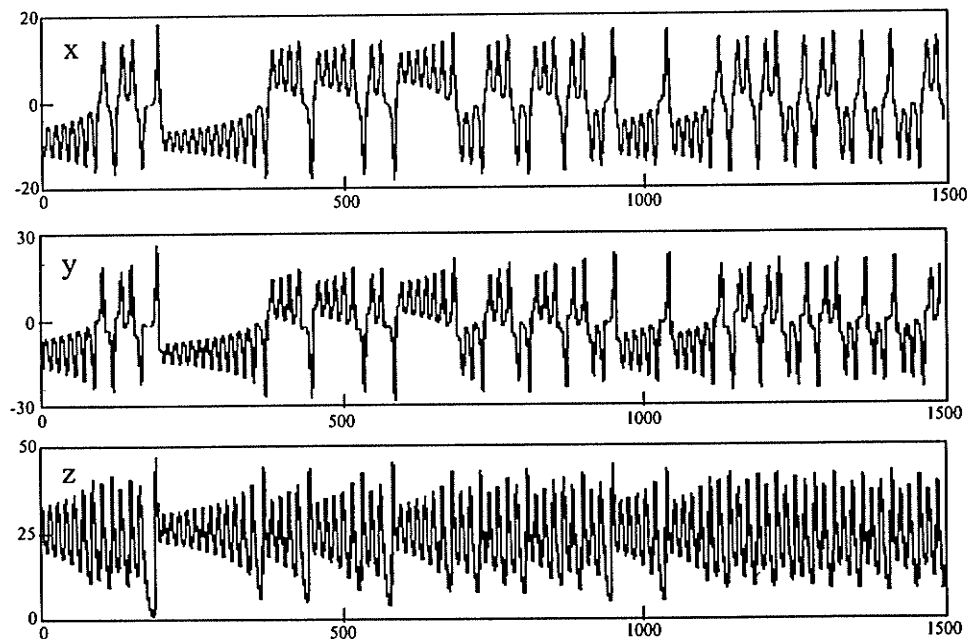


Fig. 3.3. The one dimensional trajectory of the Lorenz system for  $\sigma = 10$ ,  $\eta = 8/3$ , and  $\gamma = 28$  with the initial values at  $x(0) = 0$ ,  $y(0) = 1$ , and  $z(0) = 0$ .

### 3.3.2 Strange Attractors

First let us consider the example of Lorenz dynamical system. The system is governed by the Lorenz equations

$$\begin{cases} \frac{d}{dt}x(t) = -\sigma[x(t) - y(t)] \\ \frac{d}{dt}y(t) = -x(t)z(t) + \gamma x(t) - y(t) \\ \frac{d}{dt}z(t) = x(t)y(t) - \eta z(t) \end{cases} \quad (3.17)$$

The critical parameter is  $\gamma$ . It determines the stability of the solutions. The range of  $\gamma \in [27.74, 100.5]$  determines that the equations exhibit a chaotic behaviour. Here, the values of the parameters are selected as  $\sigma = 10$ ,  $\eta = 8/3$ , and  $\gamma = 28$ . The initial values are  $x(0) = z(0) = 0$  and  $y(0) = 1$ . Figure 3.3 shows the  $x$ -component,  $y$ -component, and the  $z$ -

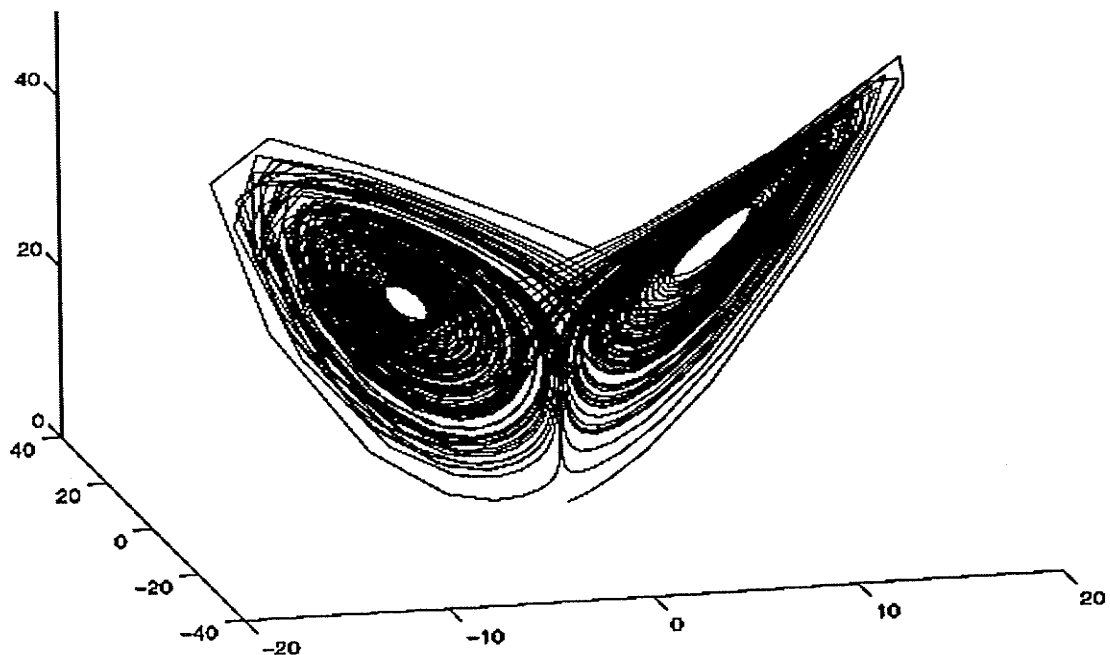


Fig. 3.4. The strange attractor of the Lorenz system displayed at low resolution.

---

component of the system versus time  $t$ , respectively. Figure 3.4 exhibits the three-dimensional phase space of a chaotic solution to the Lorenz system.

For a dynamical system, the set of all dependent variables constitutes a *phase space*, *i.e.* a Euclidean space whose coordinates are these variables. In the case of the Lorenz system, the phase space is three-dimensional. Each point in this phase space represents a possible instantaneous state of the Lorenz system. A solution of the Lorenz equations is represented by a particle travelling along an orbit or *trajectory* in this phase space. As exhibited in Figs. 3.3 and 3.4, the trajectory of the Lorenz system in phase space demonstrates the existence of a bounded object, although the time waveform shows the random or unpredictable behaviour. This trajectory is called *strange attractor*. The name of strange attractor refers to its unusual properties of sensitivity to initial conditions. It is noticed that a single point in the phase space determines the entire future trajectory since such a point represents a complete set of initial conditions for the Lorenz equations. This means that a trajectory in its phase space can never cross. However, a trajectory may intersect if the projective space has fewer dimensions than the embedding dimension.

Chaos and fractals are different fields. Chaos deals with time evolution and its underlying or distinguishing characteristics, while fractals deal with geometric patterns and quantitative ways of characterizing those patterns. However, chaos and fractals are closely intertwined and often occur together. Most chaotic attractors, for example, have a fractal striated texture [PeJS92]. Because of their close relationships, studies in one field may help in the other.

---

Modelling a natural nonlinear chaotic phenomenon is extremely difficult due to the chaotic features of sensitivity to initial condition and unpredictability. One way to characterize the phenomenon is through multifractal analysis, using the multifractal features of its strange attractor, if there is a strange attractor for this natural nonlinear chaotic system. In order to do so, it has to be determined whether the underlying system is a deterministic or a stochastic system and more important, to have the strange attractor. But in practice, it is difficult to obtain enough information on all variables of the underlying system and usually we can only access the trajectory of one measured variable due to technical limitations. This leads to the issue, to be discussed in the following section, reconstructing its strange attractor using one component variable of the all variables in a chaotic system.

### 3.3.3 Characterization of Dynamical Systems

For a dynamical system characterization, the majority difficulty is that we usually have only incomplete information of the system. The measured time series does not typically cover all the degrees of freedom of the system. However, a time series of a single variable can carry the information about the dynamics of the entire multivariable system and this allows the attractor of the chaotic system to be reconstructed [PCDS80]. Several techniques for attractor reconstruction are currently employed, such as time delay [Take81], derivative coordinates [PCDS80], and singular-value decomposition [BrKi86]. The method of time delay is the most widespread approach since it is the most straightforward and the noise level is constant for each delay component.

### 3.3.3.1 Time Delay Approach

Basically, delay coordinates are used to form  $M$ -dimensional state-space vectors,  $\mathbf{X}_i$ . That is, the reconstructed trajectory,  $\mathbf{X}$ , is given by

$$\mathbf{X} = [\mathbf{X}_1 \mathbf{X}_2 \mathbf{X}_3 \cdots \mathbf{X}_M]^T \quad (3.18)$$

For a single-value time series,  $\{x_1, x_2, x_3, \dots, x_N\}$ , the reconstructed state of the  $m$ -dimensional system at each discrete time  $i$  is

$$\begin{aligned} \mathbf{X}_i &= [x_i \ x_{i+\tau} \ x_{i+2\tau} \ \cdots \ x_{i+(m-1)\tau}] \\ i &= 1, 2, 3, \dots, N - (m-1)\tau \end{aligned} \quad (3.19)$$

where  $m$  is the embedding dimension,  $N$  is the length of the time series, and  $\tau$  is called the *lag* of the time series.

According to Takens' theorem, a faithful reconstruction is guaranteed as long as the relationship of the embedding dimension  $m$  and the topological dimension  $n$  are

$$m > 2n \quad (3.20)$$

In theory, the time delay  $\tau$  can be chosen arbitrarily if an infinite amount of noise-free data is used [Take81]. However, the quality of the analysis depends on the value chosen for  $\tau$  if only a limited amount of noisy data is available. In practice, if a value of lag is too small, then there is little information between the successive delay coordinates and the constructed trajectory becomes compressed along the main diagonal of the embedding space. On the other hand, large values of  $\tau$  usually results in contiguous delay coordinates becoming uncorrelated. The reconstruction is no longer representative of the true dynamics.

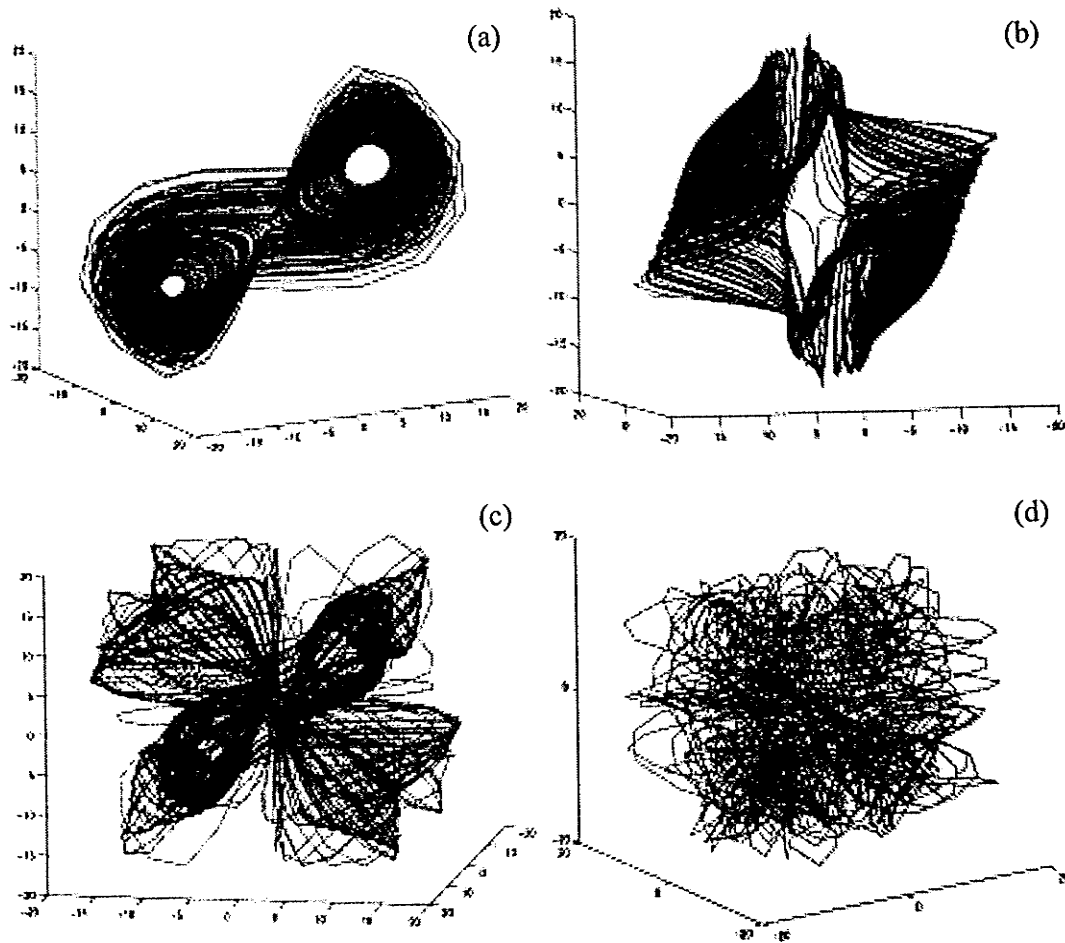


Fig. 3.5. Lorenz attractor reconstructed from the  $x$  component trajectory.  $\sigma = 10$ ,  $\eta = 8/3$ , and  $\gamma = 18$ ,  $x(0) = z(0) = 0$ ,  $y(0) = 1$ , embedding dimension is 2 and 3000 points are used for the reconstruction. (a)  $\tau = 2$ , (b)  $\tau = 8$ , (c)  $\tau = 17$ , and (d)  $\tau = 100$ .

Figure 3.5 shows the reconstructed Lorenz attractor based on the  $x$ -variable trajectory of the Lorenz system with a lag  $\tau$  of 2, 8, 17, and 100. As can be seen, the reconstructed Lorenz attractor shares the basic dynamical properties with the original Lorenz attractor for the low lag values. For large lag values, the attractor starts to lose its structure.

### 3.3.3.2 Mutual Information Method for Choosing Lag

There are two methods of choosing the best lag. The *autocorrelation function* measures the linear dependence of the points in an attractor, while the *mutual information* measures the general dependence of the points. Therefore, it is expected that using the mutual information method would give a better measure of the shift from redundancy to irrelevance [FrSw86].

The mutual information is defined as

$$M = \sum_{i=1}^{N-\tau(m-1)} P[x_i, x_{i+\tau}, \dots, x_{i+\tau(m-1)}] \times \log \left( \frac{P[x_i, x_{i+\tau}, \dots, x_{i+\tau(m-1)}]}{P[x_i]P[x_{i+\tau}] \dots P[x_{i+\tau(m-1)}]} \right) \quad (3.21)$$

where  $P(x_i)$  is the probability of the occurrence of the time series variable  $x_i$ ,  $P[x_i, x_{i+\tau}, \dots, x_{i+\tau(m-1)}]$  is the joint probability of occurrence of the attractor coordinates  $X_i = [x_i, x_{i+\tau}, x_{i+2\tau}, \dots, x_{i+(m-1)\tau}]$ , and  $m$  is the embedding dimension.  $M$  is a measure of the general statistical dependence of the reconstruction variables on each other. If the coordinates are independent statistically, such as white noise, then

$$P[x_i, x_{i+\tau}, \dots, x_{i+\tau(m-1)}] = P[x_i]P[x_{i+\tau}] \dots P[x_{i+\tau(m-1)}] \quad (3.22)$$

In this case,  $M = 0$ . Therefore, the mutual information of white noise is zero.

In practice, the coordinate at the first local minimum of the mutual information is selected as the lag value. For a special case of a two-dimensional reconstruction of an attractor in  $x_i$  and  $x_{i+\tau}$  planes, the mutual information,  $M_2$ , is deduced as

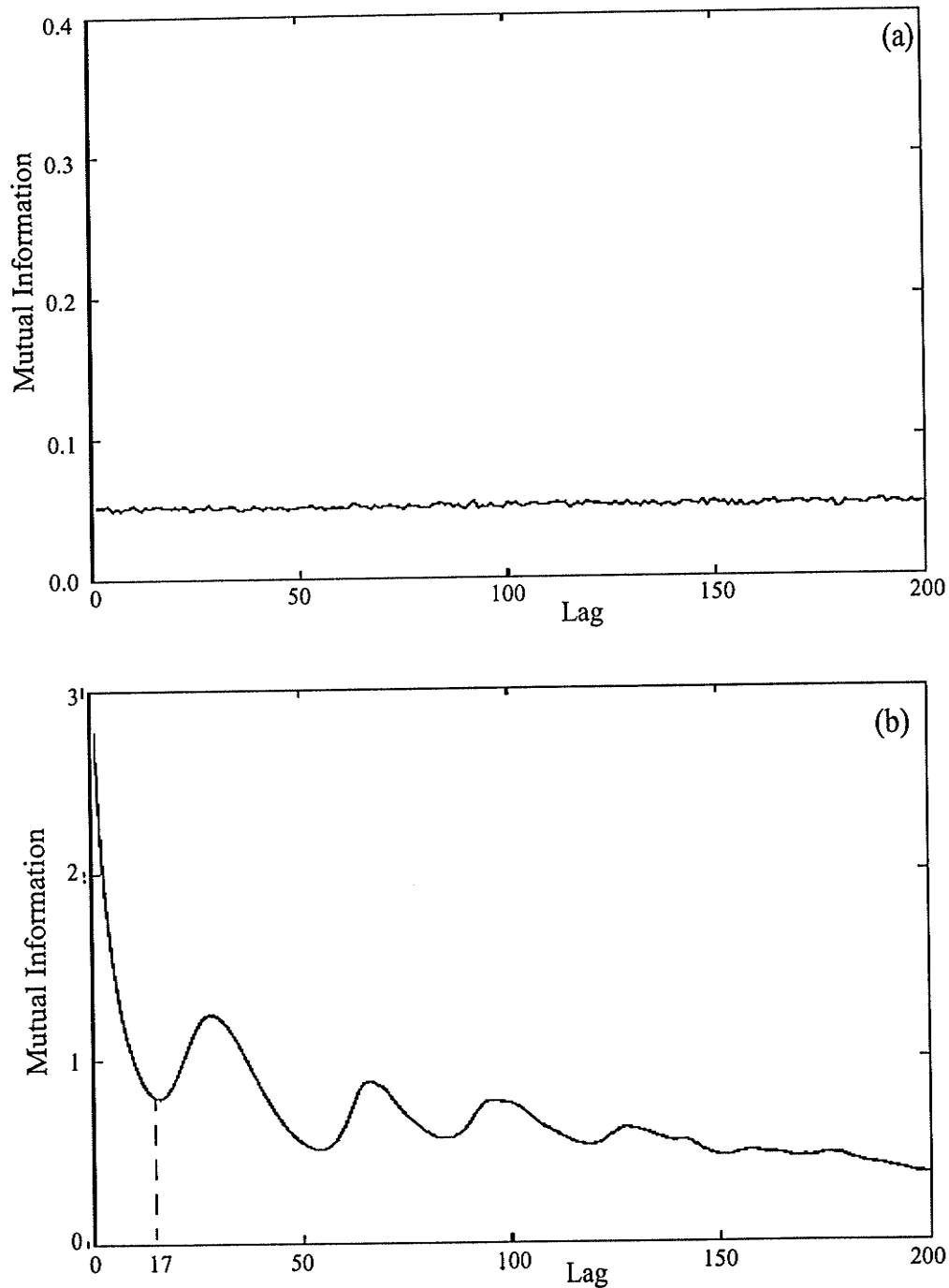


Fig. 3.6. The mutual information function of the (a) white noise and (b) Lorenz attractor. The lag value corresponding to the first local minimum of the mutual information function is selected as the best lag. For the white noise, the value of mutual information is very small due to the fact that there is no correlation among the points. For Lorenz attractor, a lag of 17 at the first local minimum mutual information is chosen.



$$M_2 = \sum_{j=1}^{N_c} \sum_{k=1}^{N_r} P(j, k) \log \left[ \frac{P(j, k)}{P(j)P(k)} \right] \quad (3.23)$$

where  $N_c$  and  $N_r$  are the number of vels along a row and a column, respectively.  $P(j)$  and  $P(k)$  are the probability of the occurrence of the attractor in column  $j$  and row  $k$  respectively.  $P(j, k)$  is the joint probability of the attractor in the column  $j$  and row  $k$  vel. The relationship of  $M_2$  versus, using Lorenz attractor as an example, is plotted in Fig. 3.6. As shown in the figure, the best lag is selected as the first minimum of  $M_2$ .

It is noted that, as can be seen in Fig. 3.6, the mutual information of the stochastic system such as white noise is significantly small and varies in a very trivial range, indicating there is no correlation in general among these points.

### 3.3.3.3 False Nearest Neighbourhood Method for Choosing

#### Embedding Dimension

There are mainly two methods to determine the dimensionality of a dynamical system. Correlation dimension analysis is the most often used method. The usual problem associated with this method is that often the dimensionality of a system is too high to find a clear indication of a finite dimension.

Another method of determining the dimensionality of a system proposed by Kennel *et al.* [KeBA92] is to estimate the percentage of the *false nearest neighbours*. The main idea is that for a deterministic system, the closed points that are true neighbours in the  $n$ -dimensional embedding space stay close in the  $n+1$  embedding dimension space. On the other hand, the points may appear as close neighbours by the projection effects in the  $n$ -

dimension space, if  $n$  is too small. These points are mapped randomly onto the entire attractor in the  $n+1$  dimensions space.

Based on this idea, for a point  $\vec{x}_i$  of an attractor reconstructed from the time series  $x$  in  $n$  dimensions, the Euclidean distance,  $R_n^2(i, r)$ , between the point and its  $r$ th nearest neighbour,  $\vec{x}_i^{(r)}$  is given by

$$R_n^2(i, r) = \sum_{k=0}^{n-1} [x_{i+k\tau} - x_{i+k\tau}^{(r)}]^2 \quad (3.24)$$

where  $\tau$  is the lag.

In  $n+1$  dimensions, the Euclidean distance is

$$R_{n+1}^2(i, r) = R_n^2(i, r) + [x_{i+n\tau} - x_{i+n\tau}^{(r)}]^2 \quad (3.25)$$

Thus, the first Kennel's criterion for false neighbours is given by

$$\left[ \frac{R_{n+1}^2(i, r) - R_n^2(i, r)}{R_n^2(i, r)} \right]^{1/2} = \frac{|x_{i+n\tau} - x_{i+n\tau}^{(r)}|}{R_n^2(i, r)} > R_{tol} \quad (3.26)$$

where  $R_{tol}$  is a threshold for the criterion. The value of  $R_{tol}$  is selected experimentally by checking the sensitivity on different values of  $R_{tol}$ . In general, it is enough to consider the nearest neighbour, *i.e.*  $r = 1$ .

This criterion by itself is not sufficient for judgment because it is unnecessarily close to  $\vec{x}_i$ , even though  $\vec{x}_i^{(r)}$  is the nearest neighbour of  $\vec{x}_i$ . So the additional criterion, which discards those nearest neighbours located on the boundary of the attractor, given by Kennel *et al.* is

$$\frac{R_n(i, r)}{R_A} > A_{tol} \quad (3.27)$$

where  $A_{tol}$  is a threshold for the second criterion and  $R_A$  is the standard deviation of the data and represents the attractor size. Therefore, a nearest neighbour is declared as false if either criterion test fails.

Figure 3.7 shows the percentage of calculated false nearest neighbours for Lorenz attractor with  $A_{tol} = 2$  and  $R_{tol}$  from 0.5 to 30. As can be seen, the percentage of nearest neighbours falls to zero when the embedding dimensions are not less than three, indicating that the Lorenz attractor has a low embedding dimension structure. This method can also be used to distinguish between deterministic chaos and a stochastic system. As shown in Fig. 3.8, the ratio of false nearest neighbours does not drop to zero for a white noise as the embedding dimension increases. This suggests that white noise has a higher dimensional structure and, therefore, is not a low-dimensional deterministic chaotic system.

#### 3.3.3.4 Distinguishing Stochastic System From Deterministic Chaos

It is noticed that distinguishing deterministic chaos from noise is also an important issue. A good algorithm is able not only to characterize chaotic systems accurately but to identify between noise and chaos. In this section, a further utility of our approach is established by comparing the performance of the approach on white noise with deterministic chaos.

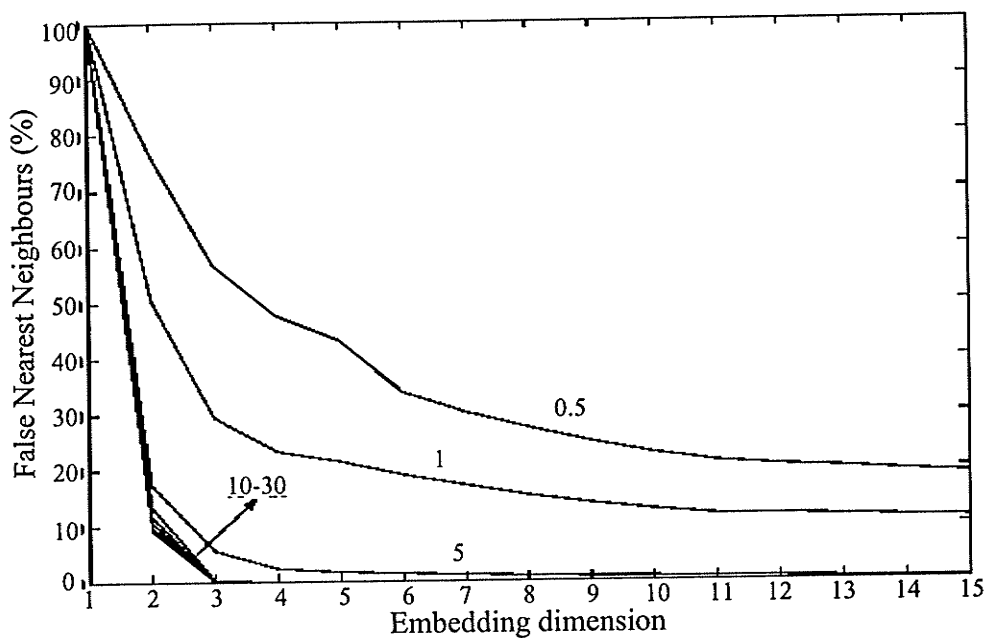


Fig. 3.7. The percentage of false nearest neighbourhoods for the Lorenz attractor. The values on the curves represent the different  $R_{tol}$ .

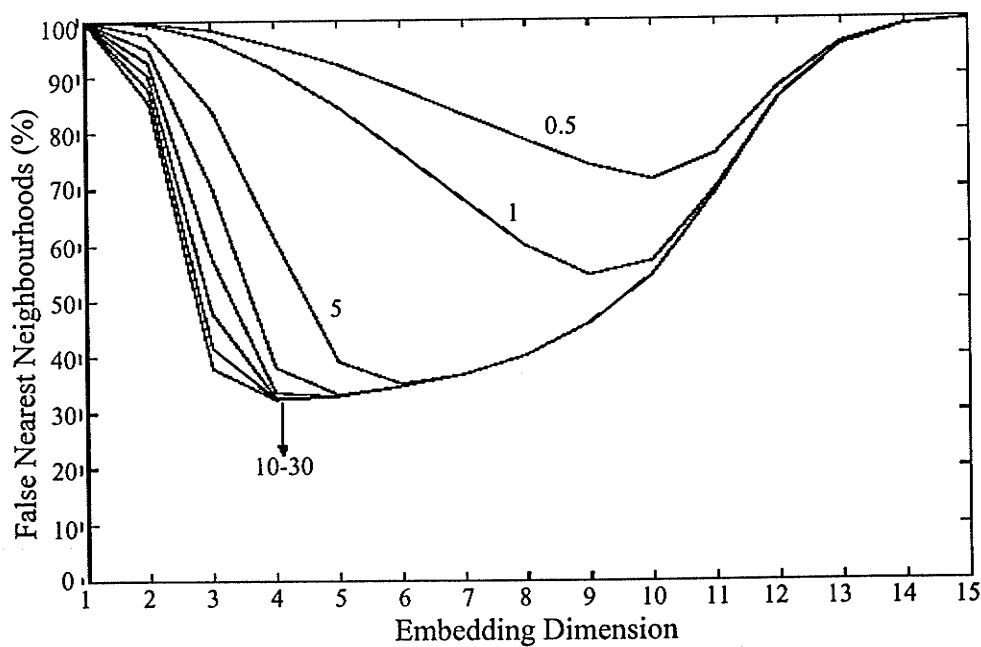


Fig. 3.8. The percentage of false nearest neighbourhoods for white noise. The value on each curve represents the  $R_{tol}$  value.

As shown in Fig. 3.6 (a), the values of the mutual information of the white noise are small in comparison to a deterministic chaos, Lorenz system (Fig. 3.6 (b)), indicating that there is no dependency between the points in the white noise system. Further characterization of white noise, as shown in Fig. 3.8, using false nearest neighbourhood method exhibits that the ratio of false nearest neighbours do not fall to zero as the embedding dimension increases. This suggests that white noise has a higher dimensional structure. The results also demonstrate that the method of the false nearest neighbourhood is able to distinguish high-dimensional systems from low-dimensional chaos. In general, a high-dimensional system is considered as random noise.

The reconstructed white noise is shown in Fig. 3.9. As can be seen, a high-dimensional noise distributes in the entire phase space without any boundary since there is no correlation between the points.

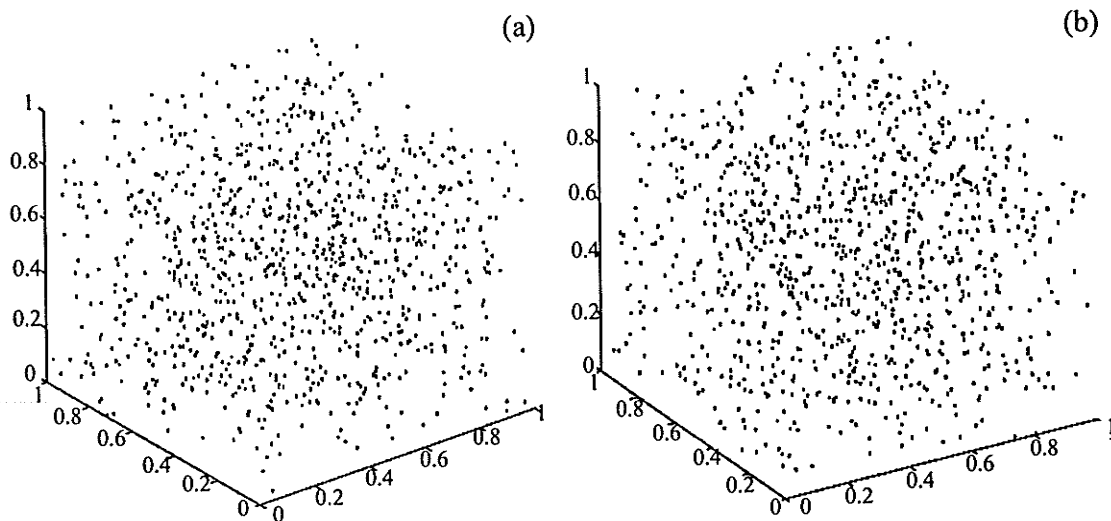


Fig. 3.9. The reconstructed white noise system with a lag of 1 for (a) and 10 for (b).

### 3.4 Summary

In this chapter, the background knowledge about fractal and chaos are provided. The algorithms of fractal and multifractal analysis are described. Strange attractors and their reconstruction are reviewed. The methods for determining appropriated embedding dimension and lag for reconstruction are discussed. Our approach of distinguishing between the high-dimensional noise and the low-dimensional deterministic chaos is further explained. In the next chapter, we will present the methods and experimental design for characterizing DNA sequences

## CHAPTER IV

### CHARACTERIZATION OF DNA SEQUENCES

Fractals are used mainly to model the highly nonstationary signals. In general, fractal-dimension calculations result in the signal's topology dimension, thus the fractal modelling does not exhibit much advantage. However, this approach gives good performance for nonstationary signals. Signals can be considered as measures. If the measure has self-similarity, fractal dimensions can be applied. This chapter introduces a novel approach for compositional complexity measure of DNA sequences, which has been developed for this thesis. The approach is based on the fact that the outlines of most natural objects are multifractals. In this thesis, a DNA sequence will be considered as a one-dimensional strange attractor. Hence, the measure of the complexity of DNA sequences is not on a point by point basis, but, instead to consider the DNA sequence as a whole object. Before applying multifractal analysis to DNA sequences, the problems of how to map the DNA alphabet sequences to numerical sequences and how to define the measure for points of a numerical sequence which are used in the multifractal analysis for strange attractor have to be solved.

#### 4.1 Numerical Mapping of DNA Sequences

For most analysis methods applied to DNA sequences, the first problem is how to map DNA alphabet sequences into numerical sequences. Currently, the DNA walk representation, or so called Lévy walk model [PBGS92], is mainly used in the DNA complexity analysis. The Lévy walk model, which maps a DNA alphabet sequence to a one-dimensional numerical sequence, defined in [PBGS92] is that the walker steps "down" when a

purine (A or G) occurs at position  $i$ , while the walker steps “up” when a pyrimidine (C or T) occurs at position  $i$ . Two-dimensional Lévy walk models have been also reported [ZhKi98].

The disadvantage of this model is that the artificial relationships between purine and pyrimidine are introduced by walking “up” a step for a purine and “down” a step for a pyrimidine at position  $i$ . Therefore, an analysis of the resulting numerical DNA sequence may have interference due to the artificial information.

With the success of the HGP, 96% of the entire human genome has been sequenced and published. Within the human genome, over twenty six thousand genes, or more than 65% of the entire human genes, have been annotated before the early part of in 2001. Therefore, it is possible to estimate the human codon usage using these large data sets. An example of human codon usage, which is a statistical result from a data set of approximately 12.8 million codons, available at web site (ref. [CUTG01]), is shown as Table 2.1 in Section 2.2.1 and will be used through this thesis.

To use multifractal techniques to measure the DNA alphabet sequence, we have first to map a DNA alphabet sequence into a numerical series. In order to do so, a codon, a three-base sequence, is considered to be the smallest unit in the DNA sequence instead of a base pair. Each point in the numerical sequence represents a codon in the DNA symbolic sequence. A measure of each point is assigned based on the statistical usage of the corresponding codon in the DNA alphabet sequence. Thus, a relationship between the values of the points are established by the statistical codon usage of the organism.



---

Since a DNA sequence represents three frames, a numerical mapping of the DNA sequence will result in three DNA numerical series. Only one series of a coding DNA sequence represents the actual amino acid sequence.

## 4.2 The Size of a Point of an Object in Multifractal Analysis

As discussed in Section 3.1, a particular measure is used for each fractal dimension. For example, the measure of the Hausdorff dimension is the number of vels  $N(r)$ , which cover the entire fractal object, and the Shannon entropy  $H_s$  is the measure for the information dimension. The fractal dimensions actually reflect the rate of change of the measures as the vel sizes are changed [Chen97].

Prior to applying multifractal analysis to an object, there are two basic questions that have to be solved. How to measure a “point” and how to determine the size of a “point” of a fractal object. In general, a “point” in a  $m$ -dimensional space is defined as the smallest and undividable unit in the  $m$ -dimensional space. Chen [Chen97] proposed that the measure of a pixel, the smallest unit of an image, is the amount of the grey level of the pixel. According to the theory of the Rényi dimension, it is, therefore, deduced by Chen that the size of a pixel in the fractal analysis is the inverse of the image size. The limitation of this model is that it is suitable only for the square images.

Therefore, we extend Chen’s idea to a  $m$ -dimensional fractal object. Consider a “point” of a fractal object as a uniformed unit block of the  $m$ -dimension. The value of the “point” can be represented by the amount of “mass” of the unit block or the density of the block in the  $m$ -dimensional space. Thus, in order to apply multifractal analysis to measure

a fractal object, a measure is assigned to each "point" in the  $m$ -dimensional space, *i.e.*, the measure of a point of a fractal object is the value of the point in the  $m$ -dimensional space.

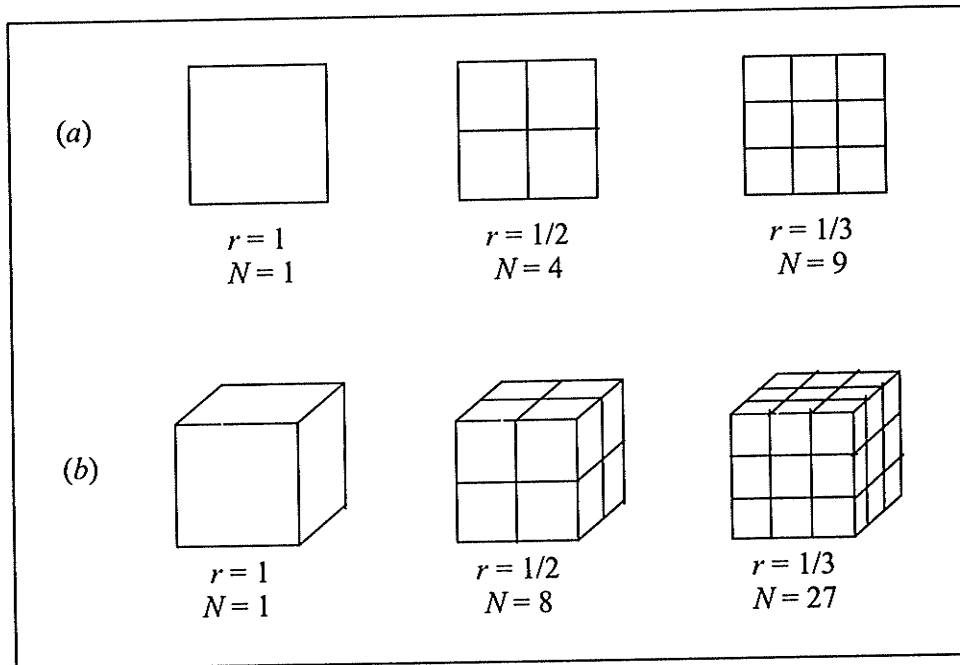


Fig. 4.1. The illustration of fractal dimensions of a plane and a cube. (a) The dimension of a plane is 2. (b) The dimension of a cube is 3.

To determine the size of a  $m$ -dimensional point, let's consider the uniformed regular geometric objects. According to the definition of the fractal dimension (Fig. 4.1), we know that the dimension values of a plane and a cube are 2 and 3, respectively. If considering a uniformed cube in the three-dimensional space, each point in the cube has the same value. When applying the multifractal measure to the uniformed cube, the values of the Rényi dimension should be 3 no matter what the value of  $q$  is.

Suppose the value of each point in a uniformed cube, which has a size of  $s_1 \times s_2 \times s_3$ , is  $C$ . The measure of each point then is denoted by  $n_j = C$  ( $j=1, 2, \dots, s_1 \times s_2 \times s_3$ ). The sum  $N$  of  $n_j$  is  $s_1 \times s_2 \times s_3 \times C$ . If we define the size of each point is  $r$ , the total number of non-overlapping vels  $N(r)$  for covering the object is  $s_1 \times s_2 \times s_3$ . Then based on the definition of the Rényi dimension (Eq. (3.11)), the estimation of  $D_q$  for the object can be obtained by

$$D_q = \frac{1}{q-1} \frac{\log \sum_{j=1}^{s_1 \times s_2 \times s_3} \left(\frac{n_j}{N}\right)^q}{\log r} \quad (4.1)$$

Therefore,  $r$  is represented by

$$r = \exp \left( \frac{\log \sum_{j=1}^{s_1 \times s_2 \times s_3} \left(\frac{n_j}{N}\right)^q}{D_q(q-1)} \right) \quad (4.2)$$

Since  $D_q = 3$ ,  $N = s_1 \times s_2 \times s_3 \times C$ , and  $n_j = C$ , thus

$$r = \exp \left( \frac{\log \sum_{j=1}^{s_1 \times s_2 \times s_3} \left(\frac{C}{s_1 \times s_2 \times s_3 \times C}\right)^q}{3(q-1)} \right) = \prod_{k=1}^3 \left(\frac{1}{s_k}\right)^{1/3} \quad (4.3)$$

Now we consider the case of an object of  $s_1 \times s_2$  uniformed plane in the three-dimensional space. Only the points on the plane have the value  $C$  and others have the value of zero. It is noted that  $D_q = 2$ ,  $N = s_1 \times s_2 \times C$ , and  $n_j = C$  in this case. In the same way as discussed above, the size of each point is given by

$$r = \exp \left( \frac{\log \sum_{j=1}^{s_1 \times s_2} \left( \frac{n_j}{N} \right)^q}{D_q(q-1)} \right) = \prod_{k=1}^2 \left( \frac{1}{s_k} \right)^{1/2} \quad (4.4)$$

If we extend to the  $m$ -dimensional space, similarly we can obtain that

$$r = \exp \left( \frac{\log \sum_{j=1}^{s_1 \times s_2 \times \dots \times s_m} \left( \frac{C}{C \prod_{k=1}^m s_k} \right)^q}{m(q-1)} \right) = \prod_{k=1}^m \left( \frac{1}{s_k} \right)^{1/m} \quad (4.5)$$

where  $s_k$ , ( $k = 1, 2, \dots, m$ ), is the topological size of the object along the  $k$ -th axis.

As in the special cases, the size of a point in the three-dimensional space is

$$r_3 = \left( \frac{1}{s_1 s_2 s_3} \right)^{1/3} \quad (4.6)$$

where  $s_1, s_2, s_3$  are the topological size of the fractal object in the three-dimensional space.

In a two-dimensional space, the size of a point of a fractal object is given by

$$r_2 = \frac{1}{\sqrt{s_1 s_2}} \quad (4.7)$$

where  $s_1$  and  $s_2$  are the size of a two-dimensional object. As a special case of a square image,  $s_1 = s_2$ . Hence, the size of a pixel is  $r_2 = \frac{1}{s}$ , which is the case discussed by Chen [Chen97]. For an one-dimensional line, the size of a point in the line is given by

$$r_1 = \frac{1}{s} \quad (4.8)$$

where  $s$  is the length of the line object.

Eq. (4.8) suggests that *the size of a point in a numerical series or a sequence is the inverse of the length of the series in the fractal analysis.*

### 4.3 Multifractal Model of Numerical Series

First for a numerical series, we define that the length of the series is  $s$  if the series is composed of  $s$  points. As described in Chapter 3, the scaling issue, or in other words, *vels and how to choose the vel size, is the critical issue when applying multifractal measures.* Based on the discussion in Section 4.1 and 4.2, a measure of a point, in an one-dimensional series is related to the value of the point, *i.e.*

$$n_j = C_j \quad j = 1, 2, \dots, s \quad (4.9)$$

where  $n_j$  is the measure value, or “mass” of the  $j$ -th point,  $C_j$  is the value of the  $j$ -th point, and  $s$  is the length of the series. Therefore, the total “mass” of the fractal object is

$$N = \sum_{j=1}^s C_j \quad (4.10)$$

It has been discussed that for a numerical series with a length of  $s$ , the size of each point in the series is

$$r = \frac{1}{s}. \quad (4.11)$$

Based on this, the multifractal measures of the numerical series are approximately as follows.

#### 4.3.1 Approximations of the Multifractal Measures on the Numerical Series

According to Eq. (3.11), an estimation of the Rényi dimension,  $\hat{D}_q$ , for a numerical series is given by

$$\hat{D}_q = \frac{1}{q-1} \frac{\log \sum_{j=1}^s C_j^q - \log \left( \sum_{j=1}^s C_j \right)^q}{\log \frac{1}{s}} \quad (4.12)$$

An estimation of the Hölder exponent  $\hat{\alpha}_q$  for a numerical series, based on Eq. (3.15), is

$$\hat{\alpha}_q = \frac{\sum_{j=1}^s C_j^q \left[ \log C_j - \log \left( \sum_{j=1}^s C_j \right) \right]}{\left( \log \frac{1}{s} \right) \sum_{j=1}^s C_j^q} \quad (4.13)$$

From Eq. (3.16), an approximation of the Mandelbrot dimension of a numerical series,  $f(\hat{\alpha})$ , is

$$f(\hat{\alpha}) = \frac{\sum_{j=1}^s C_j^q (\log C_j) - \sum_{j=1}^s C_j^q \log \left( \sum_{j=1}^s C_j^q \right)}{\left( \log \frac{1}{s} \right) \sum_{j=1}^s C_j^q} \quad (4.14)$$

### 4.3.2 Multifractal Measures on the Single Fractal Numerical Series

To exam the model described above, let us calculate the single fractal Cantor set with our model. As explained in Chapter 3, the Cantor set is a single fractal and the theoretical value of it is  $\log(2)/\log(3)$ . First, we construct a numerical series which has the properties of the Cantor set. An example of the Cantor numerical series looks like this

888000888000000000888000888.....

The series is composed of two types of points with values of 0 and 8, respectively.

For a Cantor numerical series of above example with a length of  $M$ , as described in Fig. 3.1, there are  $2^k$  segments and each segment has a size of  $m$  after  $k$  iterations. Hence, the total "mass" of the series is  $m \cdot 2^k \cdot 8$  and the length of the series is  $M = m \cdot 3^k$ . So the estimated Rényi dimension is

$$D_q = \frac{\log \sum_{j=1}^s \left( \frac{8}{8 \cdot m \cdot 2^k} \right)^q}{(q-1) \log \left( \frac{1}{M} \right)} = \frac{\log \left[ m 2^k \left( \frac{1}{m 2^k} \right)^q \right]}{(q-1) \log \left( \frac{1}{m 3^k} \right)} = \frac{\log(m 2^k)}{\log(m 3^k)} \quad (4.15)$$

Because we can not iterate the process infinitely for a finite numerical series, the segments have to contain at least one point or  $m \geq 1$ , which is the maximum possible number of iterations. Therefore, for  $m = 1$ , the final estimated Rényi dimension is

$$D_q = \frac{\log(2)}{\log(3)} \quad (4.16)$$

which equals to the theoretical value of the Cantor set.

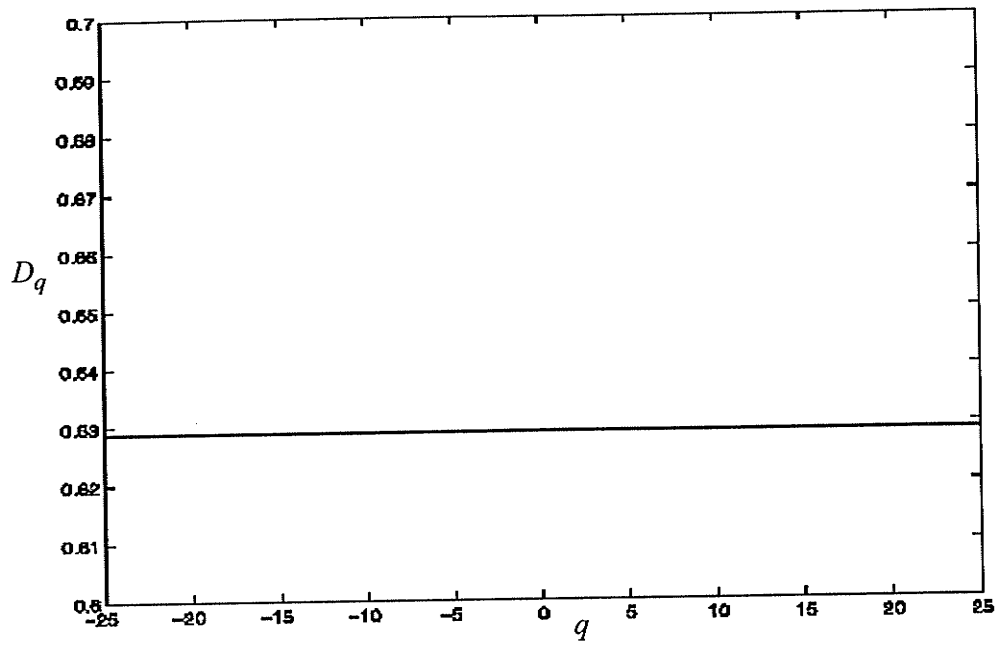


Fig. 4.2. The Rényi dimension spectrum of the Cantor numerical series.

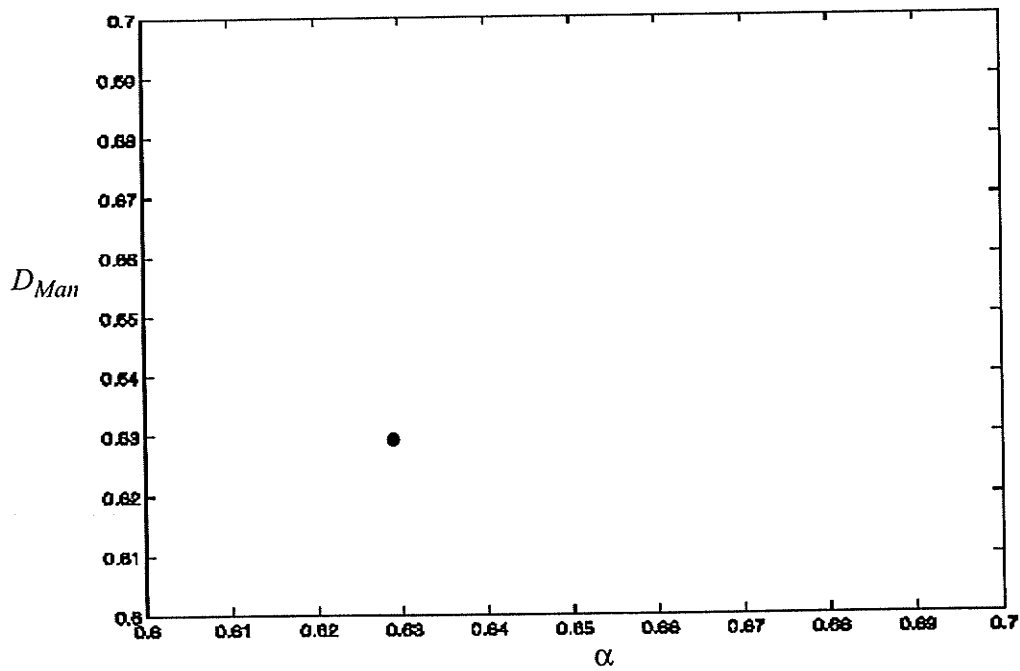


Fig. 4.3. The Mandelbrot spectrum of the Cantor numerical series.



The estimations of the Rényi dimension spectrum and the Mandelbrot spectrum are shown in Fig. 4.2 and 4.3, respectively. The Cantor numerical series used for estimating the spectra contains 2000 points. The experimental estimations of the multifractal dimensions are slightly smaller than the theoretical value (about 0.6309) because the population of the experimental series is not long enough. Since both the Mandelbrot dimension and the Hölder exponent are constant, the Mandelbrot spectrum of the Cantor numerical series is reduced to a single point.

### 4.3.3 Multifractal Measures on the Multifractal Numerical Series

Now we apply our model to the Lorenz system. As discussed in Chapter 3, the Lorenz system is a deterministic chaotic system and a multifractal. Figures 4.4 and 4.5 show the Rényi dimension spectra and the Mandelbrot spectra for the  $x$ -variable trajectory of the Lorenz system, respectively. It is noted that the multifractal measures shown in Fig. 4.4 and 4.5 are not the multifractal measure of the strange attractor of the Lorenz system but rather the measures of the trajectory for the  $x$ -variable time series shown Fig. 3.3. To measure the multifractality of the whole Lorenz system, we have to calculate its strange attractor in the three-dimensional space. The results of multifractal measure of the whole Lorenz system are shown in Figs. 4.6 and 4.7.

As can be seen in Figs. 4.4, 4.5, 4.6, and 4.7, although they are completely different objects (ref. Figs. 3.3 and 3.4), the multifractal spectra of the  $x$ -variable trajectory of the Lorenz system and the Lorenz strange attractor have a common multifractality structure. The differences of the multifractal dimension values of them reflect the space structure difference since the former is a one-dimensional fractal object and the latter is a three-

dimensional object. This result further supports the discussion in Section 3.3.3 that a time series of a single variable carries the information about the dynamics of the entire multi-variable system and therefore, the attractor of the system can be reconstructed based only on the information provided by a single variable measure.

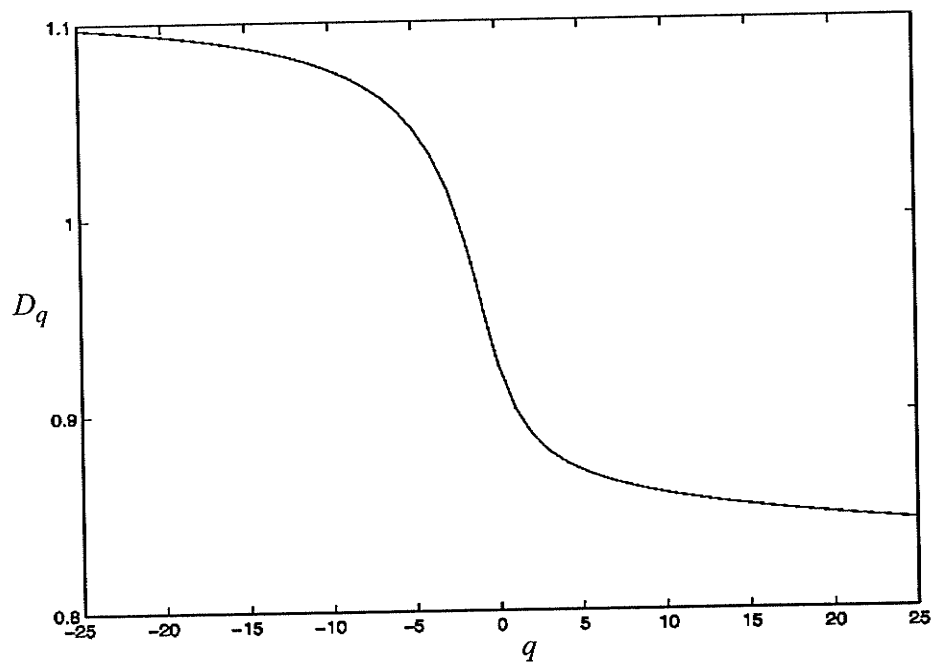


Fig. 4.4. The Rényi dimension spectrum of the x-variable trajectory of the Lorenz system.

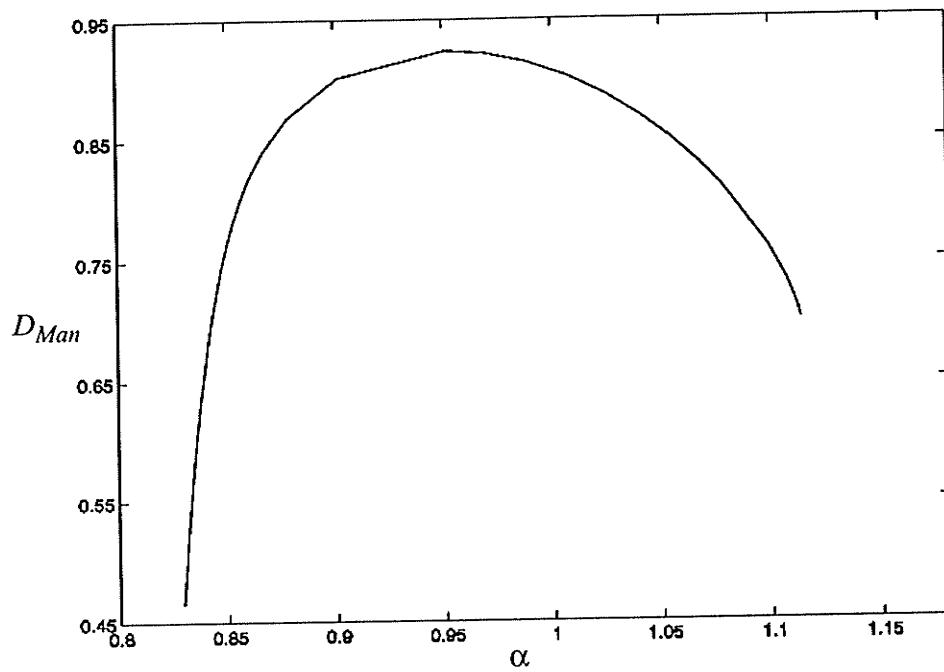


Fig. 4.5. The Mandelbrot spectrum of the x-variable trajectory of the Lorenz system.

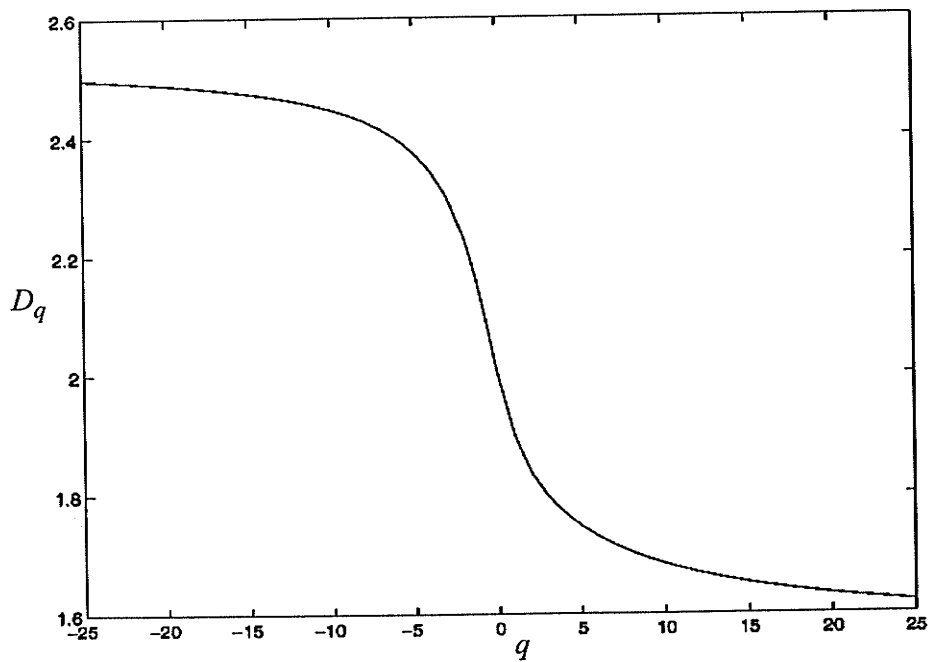


Fig. 4.6. The Rényi dimension spectrum of the Lorenz system. The calculation is based on 25,000 points of the system.

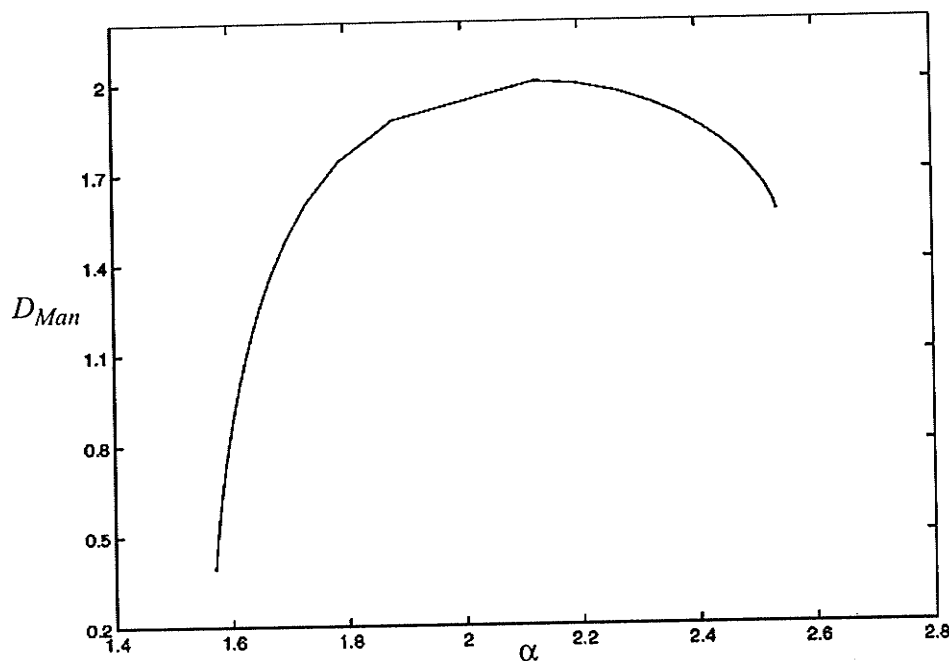


Fig. 4.7. The Mandelbrot spectrum of the Lorenz system. The calculation is based on 25,000 points of the system.

#### 4.4 Local Rényi Dimensions of the DNA Signals

In practice, a signal or a time series can be composed of several highly self-similar and nonself-similar regions. A multifractal measure of the series as a whole may not reveal the location information of these regions. For example, a DNA signal may contain both coding and non-coding regions. In many cases, we are interested in finding the locations of the coding regions, which have a higher self-similarity structure than the non-coding regions due to the codon usage. Therefore, local fractal dimensions may be a more realistic way in practice. Local dimension refers to the fact that the values of dimension are calculated based on the limited local data. A value of the fractal dimension is assigned to a point in the signal.

Our approach of local fractal dimension analysis first specifies a point of interest in a DNA signal or a DNA numerical sequence. A *sliding window* then masks the segment of the data sequence with the point located at a specific position of the segment. A dimension calculation is carried out on the masked data and the resulting value is assigned to the point. The computation is repeated for the other points by moving the window throughout the sequence.

#### 4.4.1 Window Size

The selection of the sliding window size is a critical issue. If it is too large, the estimated local dimensions may be interfered with other fractals in the signal. Another problem associated with a large window size is that a coding region which has a size less than the window size may not be distinguished from the non-coding background. However a smaller window size may result in a miscalculation due to the insufficient population of data for measuring.

As discussed in Chapter 2, the genomic DNA sequences of higher organisms are very complicated. The genes are scattered by large intergenic sequences and the coding regions of a gene are further separated by the large non-coding regions. The length of a gene or a coding region varies dramatically. To deal with that, different resolutions associated with different sliding window sizes are applied to the genomic DNA sequences. In practice, a window size of 122 bp is used for locating coding regions within a gene and a size between 200 and 300 bp is used for locating the genes within a genomic DNA sequence.

#### 4.4.2 Sliding Step

Another key factor of local dimension analysis is to determine the sliding step. The sliding step is the length of the sliding window moved each time and should be in multiples of three since a codon has a size of three bp. A large sliding step may result in missing a small exon. In practice, we choose a length of nine bp as the size of the sliding step.

#### 4.5 Summary

This chapter further describes the multifractal measures of the DNA sequences. A translation method which maps the DNA symbolic sequences into the DNA signals is discussed. The measure of a point in a  $m$ -dimensional fractal object is defined and the size of a point in a  $m$ -dimensional fractal object is determined for multifractal analysis. Based on this, a multifractal analysis model for numerical series is proposed. Some examples of single fractal and multifractal are calculated with this model. In the next chapter, the human DNA sequences are analyzed, using this model.

---

## CHAPTER V

### EXPERIMENT DESIGN

This chapter provides the experimental design for the characterization of the DNA sequences through multifractal analysis. As modeled in Chapter 4, the analysis of the DNA sequences includes four parts, e.g. numerical mapping, chaotic property, feature extraction, and “on-line” analysis. The specific design and detailed parameter setting are presented in the following sections.

#### 5.1 The Experimental DNA Sequences

Two types of DNA sequences are used in this thesis. Generated DNA sequences including the random DNA sequences and the Cantor DNA sequences serve as standard sequences for system testing. The human DNA sequences including genomic, cDNA, exon and intron sequences are selected from GenBank.

##### 5.1.1 The Generated Random and Cantor DNA Sequences

The random DNA sequences are generated from a uniform white noise source. Statistically, the four letters, A, T, C, and G have the same occurrence of 25% in the resulting random sequence. In practice, a random DNA sequence with forty thousand bp long is used for testing.

The generated Cantor DNA sequence has a property of the Cantor set (ref. Fig. 3.1). The symbolic Cantor DNA sequence contains only two characters. The points located at the line segments are filled with one character. The points located where no line segment appears are filled with the other character. A generated Cantor DNA sequence

---

looks like: AAACCCAAACCCCCCCCCAAACCCAAA. It is noted that the size of a generated Cantor DNA sequence should be a multiple of three since the size of a "point", which represents a codon, is three bp long. An example of the generated Cantor DNA sequence is shown in Appendix A. For the thesis experiments, a generated Cantor DNA sequence with a length of 20 kb was used.

### 5.1.2 The Human DNA Sequences

For real human sequences, genomic DNA, cDNA, exon, intron, flank region sequences have been obtained from GenBank, and are used in this thesis. Briefly, ph-20, a human genomic DNA sequence, is located on human chromosome 3 and contains the HYAL2 gene and the exon 1 of the HYAL1 gene. HUMHBB, a human genomic DNA sequence, is located on human chromosome 11 and contains a cluster of human  $\beta$ -globin genes including  $\epsilon$ , G $\gamma$ , A $\gamma$ ,  $\delta$ , and  $\beta$  globin genes. A pseudogene  $\beta$ -1 is located between the A $\gamma$  and  $\delta$  genes. The five  $\beta$ -like globin genes encode the  $\beta$  chain of the hemoglobin. The human NEB gene, located at chromosome 2, encodes a muscle protein involved in maintaining the structural integrity of sarcomeres and the membrane system associated with the myofibrils. The human HD gene, located at chromosome 4, plays a role in microtubule-mediated transport or vesicle function. The HYAL1 and HYAL2 genes, both related with the hyaluronoglucosaminidase activity, are located at Chromosome 3. The cDNA sequences of the HYAL1, HYAL2, H19, NEB, and HD genes are used for the experiments. The 5' end and the 3' end of the non-coding regions of the cDNA were removed prior to testing. The exon and intron sequences of the HYAL1 and HYAL2 genes are isolated and used as test samples. Approximately 1-3 kb flank regions of the HYAL1 and HYAL2 genes, including 5' flank and 3'



flank regions, are selected for this thesis. The detail information of these sequences is shown in Table 5.1.

Table 5.1. The information of the experimental human DNA sequence

Sequence Name	GenBank Access Number	Length (bp)	Description
ph-20	AC002455	29314	Genomic DNA, containing the HYAL2 gene and the exon 1 of the HYAL1 gene
HUMHBB	U01317	60000	Genomic DNA, containing the cluster of the five $\beta$ -globin gene and a pseudogene
HYAL1	U96078	1308	cDNA, in practice only the coding regions of the cDNA are used
HYAL2	U09577	1346	cDNA, in practice only the coding regions of the cDNA are used
NEB	NM_004543	20839	cDNA, only the coding regions of the cDNA are used
HD	NM_002111	13672	cDNA, only the coding regions of the cDNA are used
H19	BC004532	1001	cDNA, non-protein coding gene
HYAL1-5flank		3960	A part sequence of ph-20, the sequence is located at the 5' flank region of the HYAL1.
HYAL1-exon1		900	The first exon of the HYAL1 and a partial sequence of ph-20
HYAL1-intron1		979	The first intron of the HYAL1
HYAL2-5flank		1297	A partial sequence of ph-20 and located at the 5' flank region of the HYAL2
HYAL2-3flank		2100	Located at the 3' flank region of the HYAL2
HYAL2-exon1		921	The first exon of the HYAL2 gene
HYAL2-exon3		411	The third exon of the HYAL2 gene
HYAL2-intron1		520	The first intron of the HYAL2 gene
HYAL2-intron2		416	The second intron of the HYAL2 gene

## 5.2 Numerical Mapping of the DNA Symbolic Sequences

Prior to applying multifractal analysis, the DNA symbolic sequences have to be translated to numerical series or DNA signals. As stated in Section 4.1, the numerical mapping is based on the statistical codon usage of each codon in the DNA sequences. In this thesis, the human codon usage is used to assign the values to the 64 individual codons since we will analyze only human DNA sequences. A window with the size of three bp covers a segment of the DNA symbolic sequence. A value, according to the human codon usage, is assigned to the three-base sequence masked by the window. The window shifts one base at a time along the entire DNA symbolic sequence, thus producing a trajectory. This trajectory is called the DNA signal. As discussed in Chapter 2 and 4, a DNA sequence can yield three frames due to the fact that three bases represent a codon. Only one out of the three frames represents the actual amino acid sequence. Therefore, a DNA signal results in three translated frame signals. The numerical mapping machinery is illustrated as Fig. 5.1.

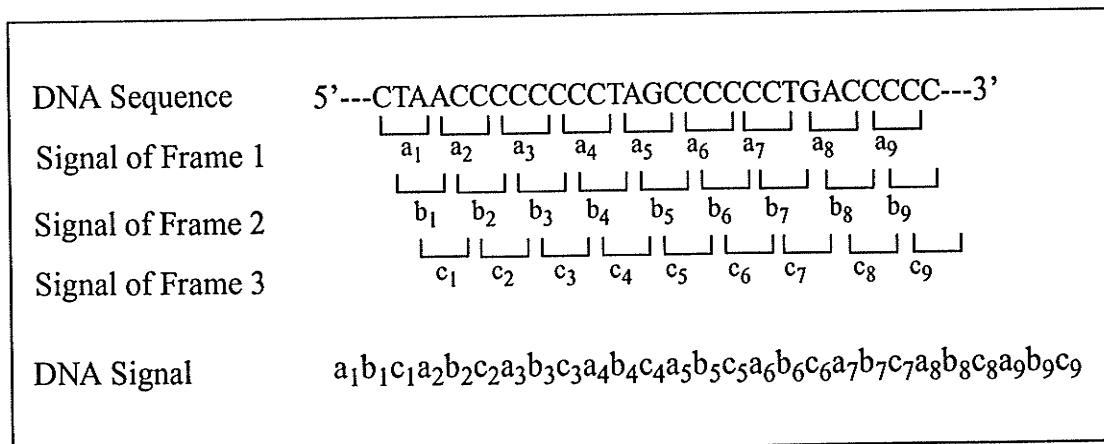


Fig. 5.1. An illustration of numerical mapping of the DNA sequence.  $a_1...a_9$ ,  $b_1...b_9$ , and  $c_1...c_9$  are the values for the specific codons.

---

An amplification and quantization of the raw translated numerical series is subsequently carried out since the values of the raw translated numerical series, which represent the 64 codons, are much less than 1. The amplification of the signals does not change the results because the fractal dimensions measure the complexity of the object rather than the values of the points in the object. In addition, fractal modelling achieves input data normalization automatically. As a matter of fact, there are theoretical lower and upper bounds for the values of the Rényi dimension regardless of the actual amplitude spread of the original signal. In fact, the Rényi dimension of a one-dimensional signal has a lower bound of 1.0 and an upper bound of 2.0. However, the experimental results may exceed the bound slightly because of the numerical artifacts introduced. The resulting DNA signals are used for multifractal measures analysis and chaotic characterization.

### 5.3 Chaotic Characterization of the DNA Sequences

The methods of the mutual information and the false nearest neighborhood are used for characterizing the chaotic properties of the DNA sequences. The white noise random DNA sequence, as a higher dimensional noise, and the deterministic chaotic system, Lorenz system, are served as standards for characterization. The DNA sequences are first mapped into the corresponding DNA signals. The DNA signals are subsequently separated into their frame signals. Both the DNA signals and the frame signals are used for analysis.

The mutual information is used to measure the general dependency of the signals and to choose the best lag value for further characterization.

The method of the false nearest neighborhood is used to distinguish a low-dimensional dynamical chaos from the high-dimensional noise systems and subsequently to

---

indicate the best embedding dimension value for a deterministic chaotic system. In practice, the  $A_{tol}$  value is set up to 2 for all experiments.

#### **5.4 Feature Extraction of the DNA Signals**

The feature extraction of the DNA signals is achieved by using multifractal measures, Rényi and Mandelbrot dimension spectra, for distinguishing the coding regions from the non-coding regions of a DNA signal. Prior to application to multifractal analysis, the DNA symbolic sequences are translated into the corresponding numerical sequences, or DNA signals. The frame signals are then isolated from the DNA signals. There are three frame signals related to a DNA signal since a codon is composed of three bases of a DNA sequence. Only one frame signal of a coding DNA sequence associates with the actual protein sequence.

#### **5.5 Local Fractal Dimension Analysis**

The local fractal dimension analysis is performed by calculating the local Rényi dimension of the trajectory of the DNA signals. To reduce computational cost, we calculate the local Rényi dimension with a moment order of  $q = -10$  since it gives the maximal differences between coding and non-coding signals.

Various sliding windows are used for different resolutions. A higher resolution of a DNA sequence is given with a smaller window size and therefore, provides more detailed information about the DNA sequence. However, more error may be introduced due to the smaller population used within a smaller window size. In practice, a sliding window size of 400 bp of the DNA signal, *e.g.* 133 points long of the frame signal, is used

for a gene along a large genomic DNA sequence. To distinguish the coding regions from the non-coding regions within a gene, a window size of 122 bp, or 60 points long in the frame signal, is applied.

## 5.6 Summary

The experiments performed for this thesis have been described in this chapter. The experimental samples including the human DNA sequences and the generated DNA sequences, such as random sequence and Cantor sequence, are discussed. The main configuration parameters are provided. In the next chapter, the experiments are conducted and results are presented along with the discussion.

---

## CHAPTER VI

### EXPERIMENTAL RESULTS AND DISCUSSION

Based on the theories and the experimental design discussed in Chapters 3, 4, and 5, the experimental work of the characterization of the DNA sequences is accordingly conducted in this chapter. First the chaotic properties of the DNA sequences are addressed. Section 6.2 focuses on the multifractality of the DNA sequences, using Rényi dimension and Mandelbrot dimension spectra. In Section 6.3, local dimension analysis is subsequently discussed according to the feature extraction of the multifractality of a DNA sequence to reveal local information along the DNA sequence.

#### 6.1 Chaotic Property of the DNA Sequences

##### 6.1.1 Mutual Information Analysis of the DNA Sequences

As described in Chapter 5, a random DNA sequence, which follows a uniform white noise distribution, and the DNA sequences of human NEB and HD genes are first translated into the corresponding DNA signals and frame signals. Mutual information analysis is subsequently applied to the resulting signals. The results are plotted as Figs. 6.1, 6.2, and 6.3.

The results, shown in Fig. 6.1, show that there is no dependence between the points along the DNA signal of the random sequence or the derived frame signals, indicating that there is no correlation between the parts of the random DNA sequence. However, as shown in Fig. 6.2, the high dependency within a range of three bps in the DNA signals of the NEB and HD genes indicates that there is a strong correlation between the bases which

have the distances within three bps in the DNA sequences. Three is the size of a codon. This suggests that there are strong correlations between the bases within a codon.

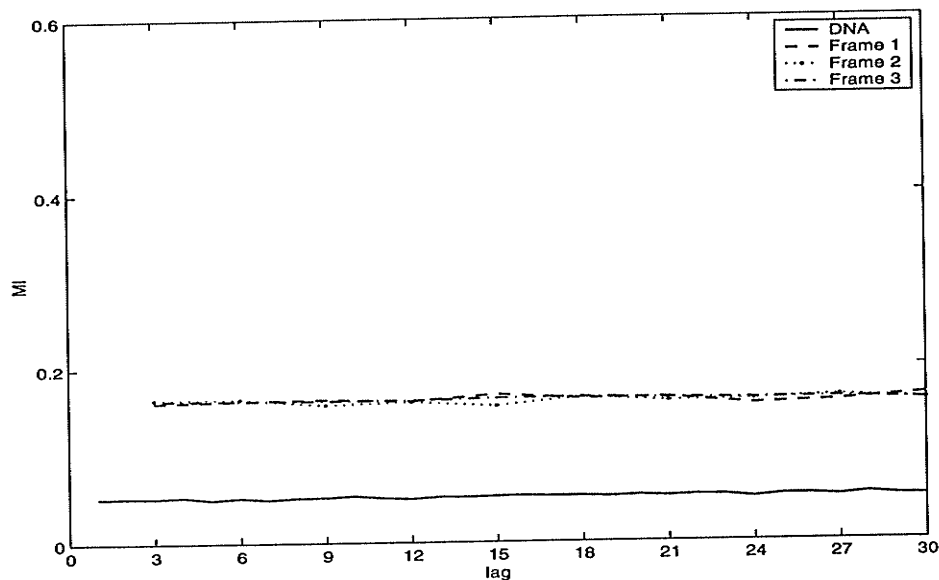


Fig. 6.1. The mutual information analysis of the DNA signal and frame signals associated to a random generated white noise DNA sequence with a size of 20 kb.

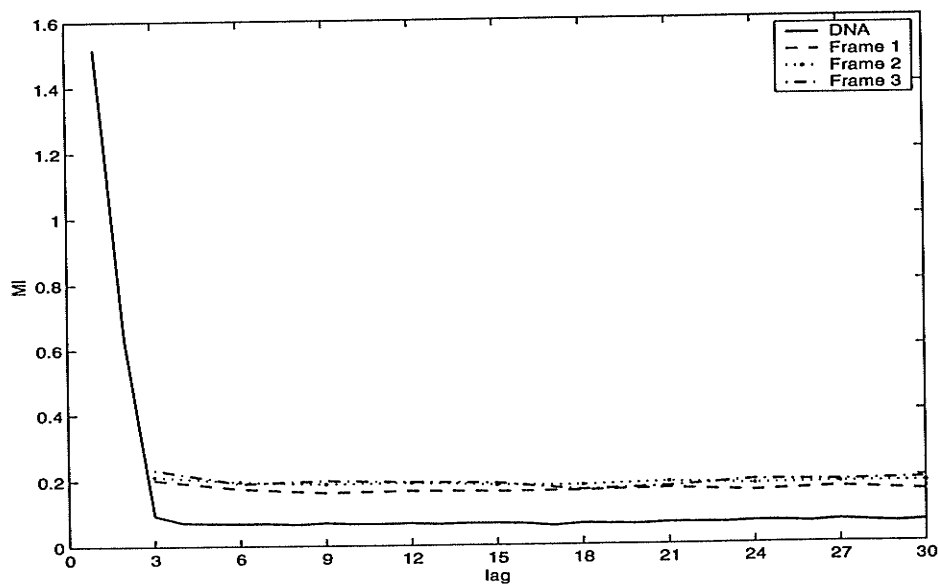


Fig. 6.2. The mutual information analysis of the human HD gene cDNA sequence.

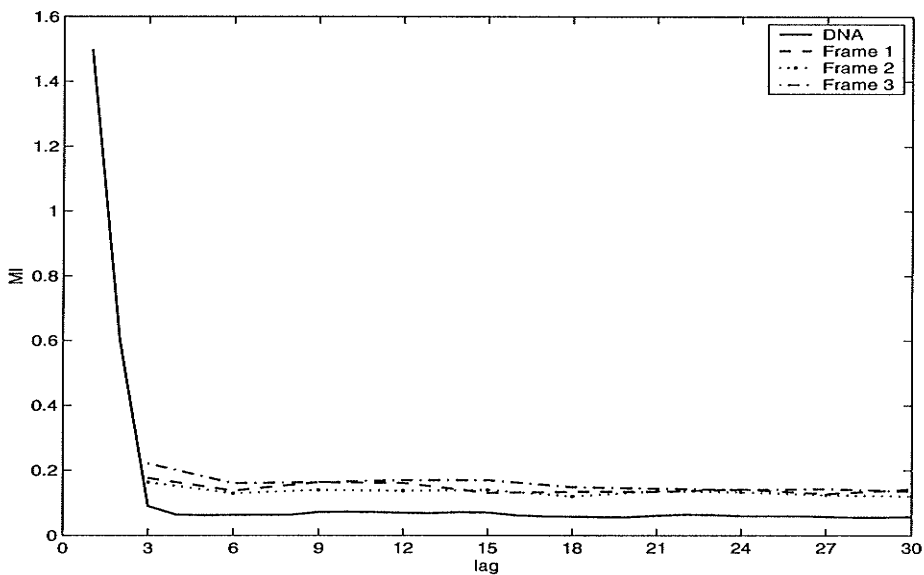


Fig. 6.3. The mutual information analysis of the human NEB gene cDNA sequence.

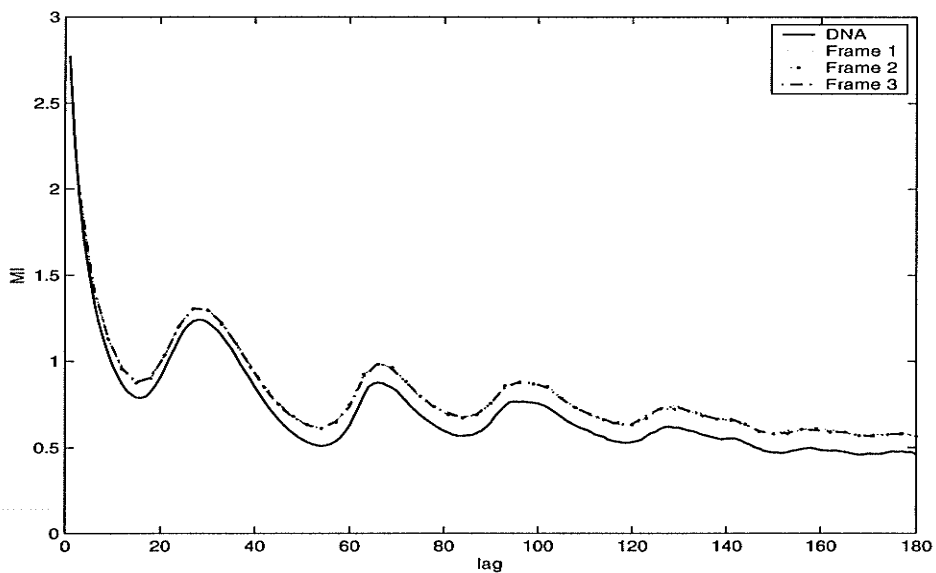


Fig. 6.4. The mutual information analysis of x-variable trajectory of the Lorenz system. The points of frame series is the collection of points with a interval of two points in the x-variable time series.



The results of the mutual information analysis also show that there is little dependency among the amino acids, suggesting that the DNA sequence as a whole has a statistical property similar as that of the white noise.

On the contrary, the result of the mutual information analysis of the x-component time series, shown in Fig. 6.4, exhibits that the points along the time series are highly correlated. The points of the frame signal are the collection of the points which have an interval of two bps in the associated x-variable time series of the Lorenz system. As can be seen, the time series signal and the frame signals have a similar dependency as the lag value increased.

#### 6.1.2 False Nearest Neighbourhood Analysis of the DNA sequences

The percentage of the false nearest neighbours of the human HD and NEB gene cDNA sequences were calculated. The experimental results are exhibited in Fig. 6.5 and 6.6. As can be seen, the percentage of the false nearest neighbours do not drop below 30% with an increasing of the embedding dimension for both cDNA sequences.

In Section 3.3.3.4, we described that a deterministic chaotic system, such as Lorenz system, has a low-dimensional structure. Hence, the percentage of the false nearest neighbours for the Lorenz system will drop to zero as the embedding dimension increases (Fig 3.7). However, for a white noise, the percentage of the false nearest neighbours does not drop to zero because it has a high-dimensional structure (Fig. 3.8). The results, shown in Figs. 6.5 and 6.6, strongly suggest that like white noise, the DNA sequences have a high-dimensional structure.

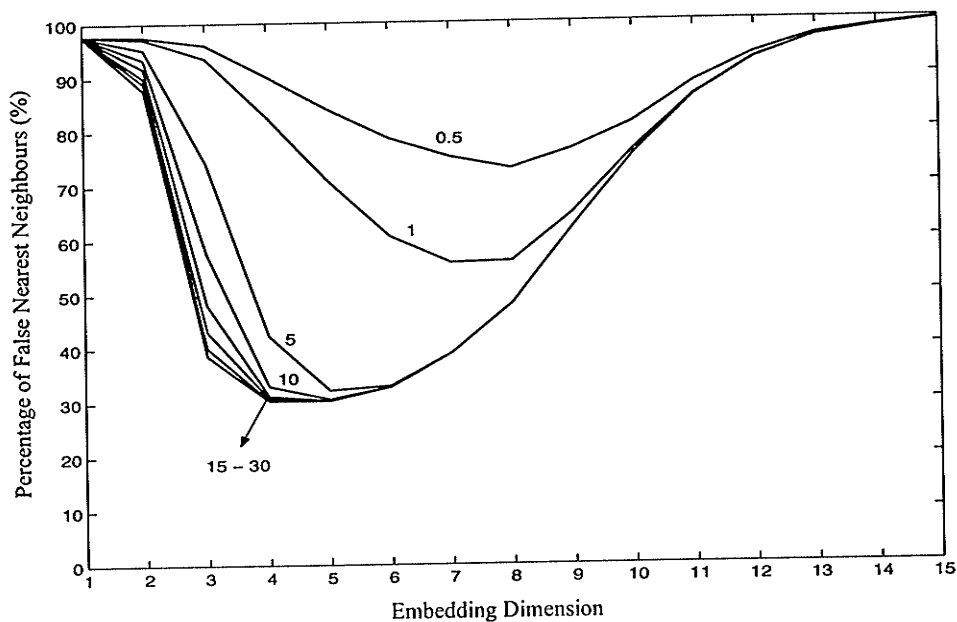


Fig. 6.5. The false nearest neighborhood analysis of the human HD gene cDNA sequence.

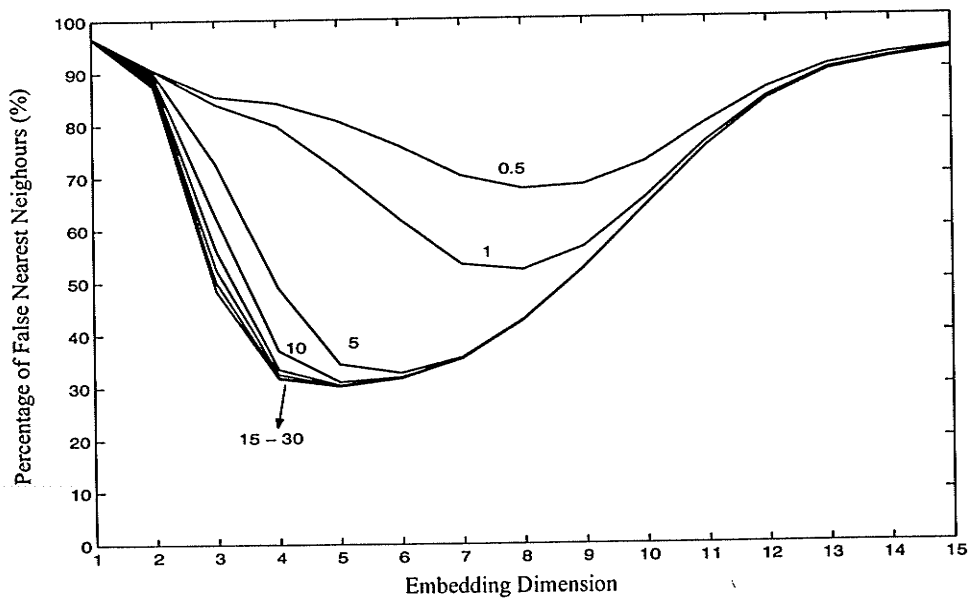


Fig. 6.6. The false nearest neighbourhood analysis of the human NEB gene cDNA sequence.

The results presented in the last two sections indicate that the DNA sequences exhibit chaos with high-dimensional properties. It should be noted, however, that no attempt has been made in this thesis to separate the chaotic from the noisy nature of DNA or its noisy representation. The advantage of this approach is, however, that the structure of DNA is studied without any assumptions about either its chaotic or noisy nature.

## 6.2 Multifractal Dimension Analysis of the DNA Sequences

### 6.2.1 Rényi Dimension Spectrum of the DNA Sequences

As stated in the previous chapters, a coding DNA sequence stands for three frames and only one frame, ORF, represents the actual amino acid sequence. To examine the multifractalities of the DNA sequences with our model, three frame signals were generated from the DNA numerical sequences, the random DNA sequence and the Cantor DNA sequence are first tested. The results are plotted in Figs. 6.7 and 6.8. Shown in Fig. 6.7, the three frame signals of the random DNA sequence have a similar structures of the Rényi dimension spectra, indicating that the three frames of the random DNA sequence have the similar multifractality. On the other hand, the Rényi dimension spectra in Fig. 6.8 indicate that Frame 1 of the Cantor dna sequence is almost a strict single fractal whose value is very close to the theoretical value of  $\log_2/\log_3$ . On the other hand, Frames 2 and 3 exhibit a slight multifractality (spread) due to the shift of one and two bases, respectively, that renders the sequence non-Cantorian.

Figures 6.9, 6.10, and 6.11 show the strong differences of Rényi dimension spectra between the ORFs and the other reading frames. The differences are mainly due to the bias codon usage in the ORFs but not in the other frames. The conclusion is further supported

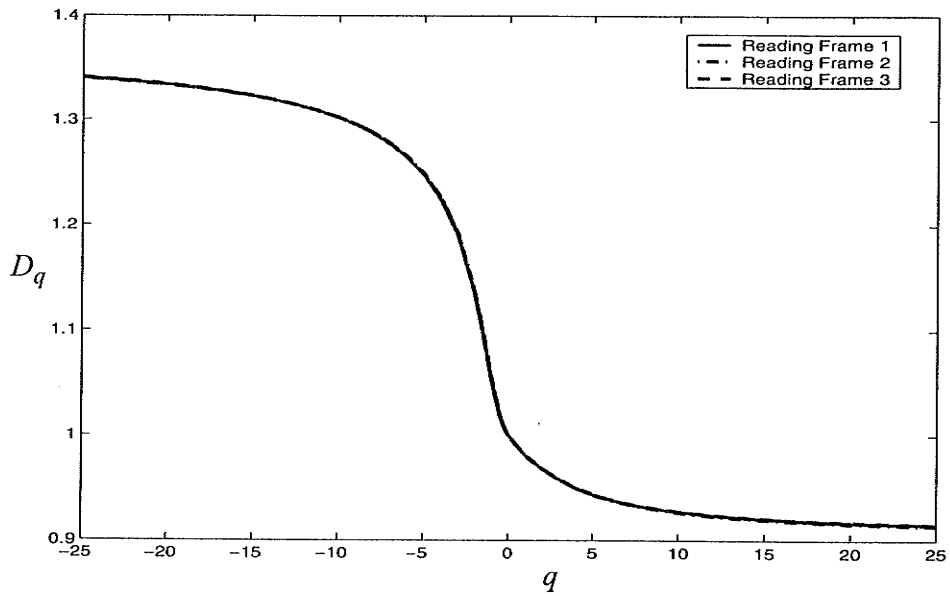


Fig. 6.7. The Rényi dimension spectra of the random DNA sequence. The solid, dashed, and dot-dash lines represent the three frames, respectively.

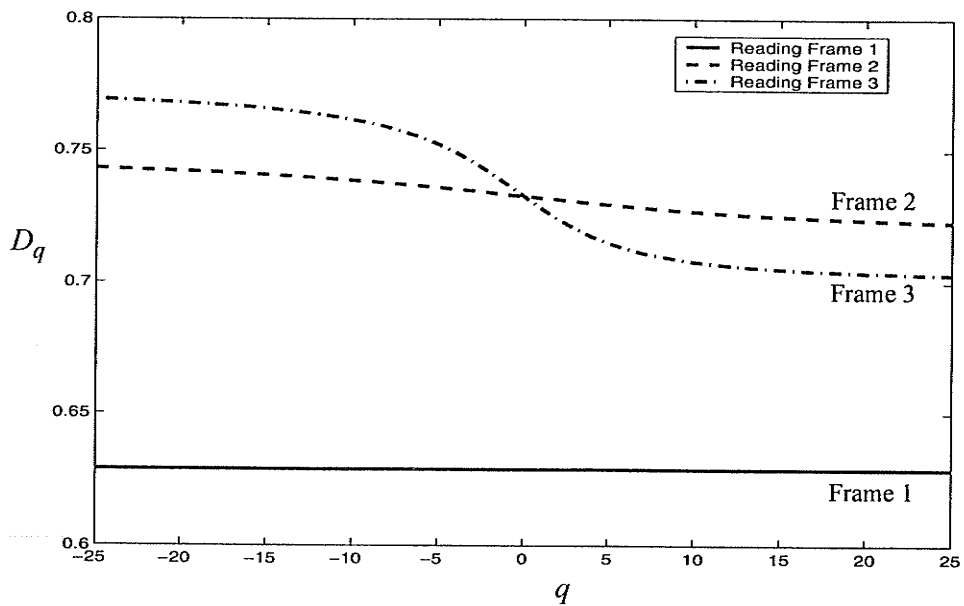


Fig. 6.8. The Rényi dimension spectra of the Cantor DNA sequence. The solid, dashed, and dot-dash lines represent the three frames, respectively.

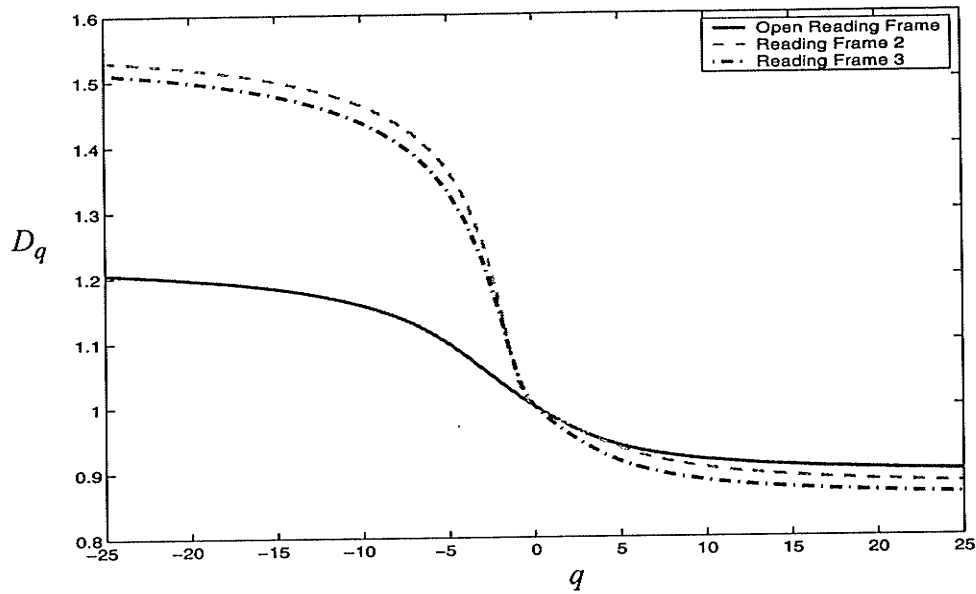


Fig. 6.9. The Rényi dimension spectra of the HYAL2 cDNA coding sequence. The solid curve represents the ORF, the dashed and the dot-dash curves denote the other two reading frames.

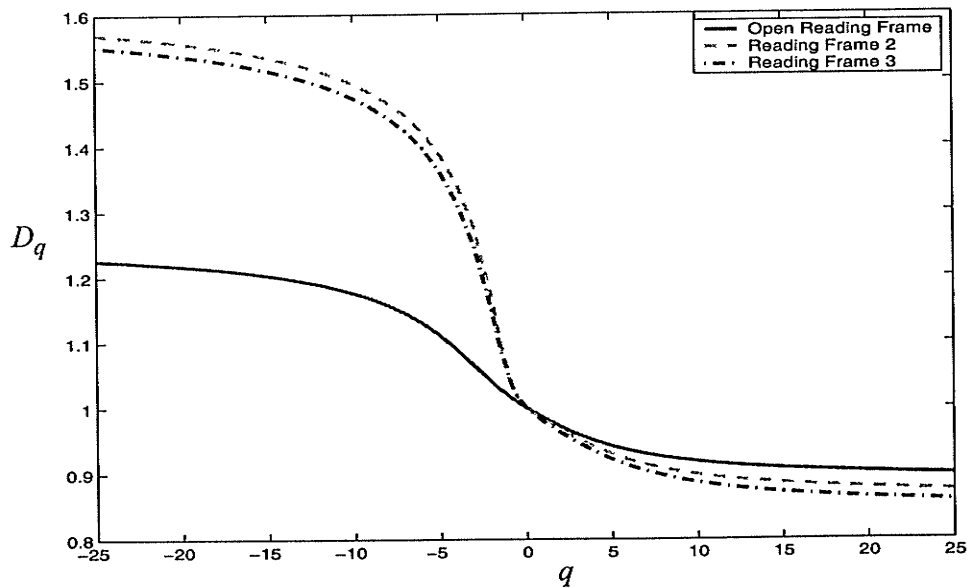


Fig. 6.10. The Rényi dimension spectra of the first exon of the HYAL1 gene. The solid curve represents the ORF, the dashed and the dot-dash curves denote the other frames.

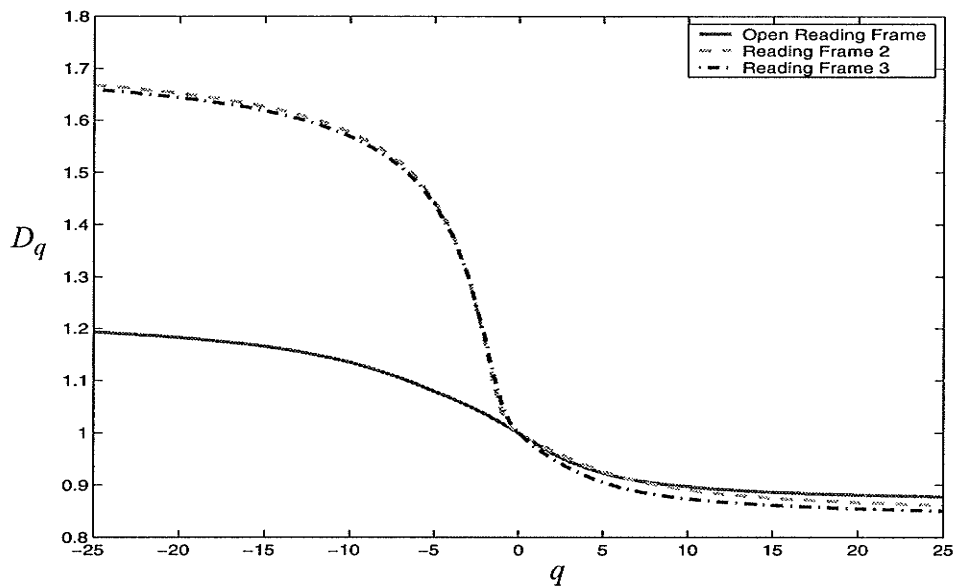


Fig. 6.11. The Rényi dimension spectra of the third exon of the HYAL2 gene. The solid curve represents the ORF, the dashed and the dot-dash curves denote the other frames.

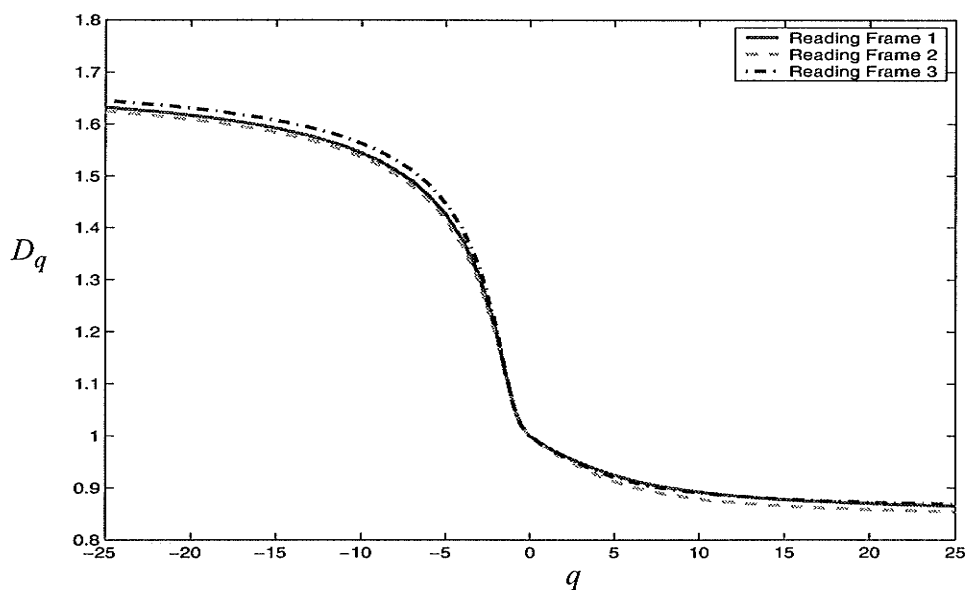


Fig. 6.12. The Rényi dimension spectra of the first intron of the HYAL2 gene. The solid, dashed, and dot-dash lines represent the three frames, respectively.

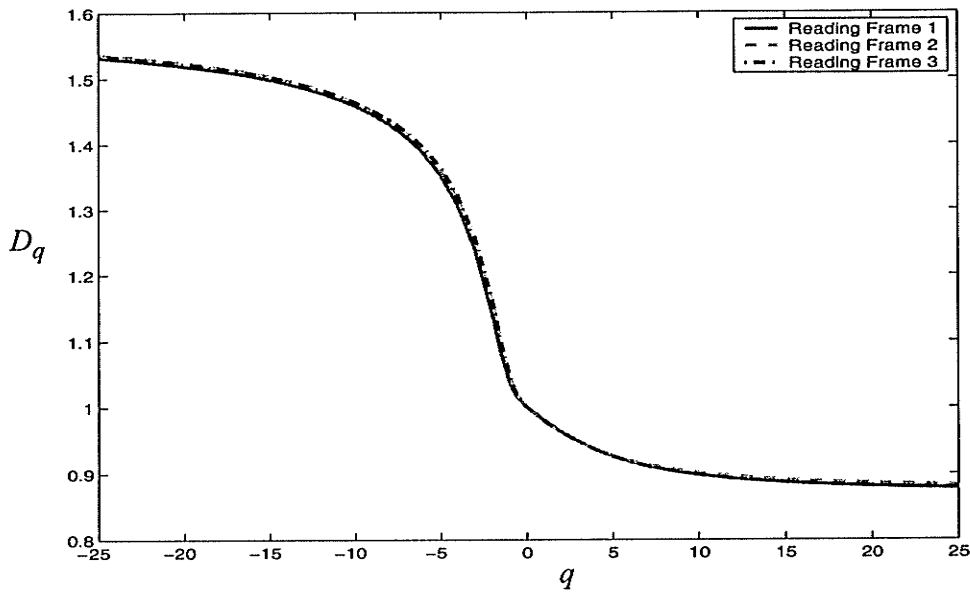


Fig. 6.13. The Rényi dimension spectra of the 5' flank region of the HYAL2 gene. The solid, dashed, and dot-dash lines represent the three frames, respectively.

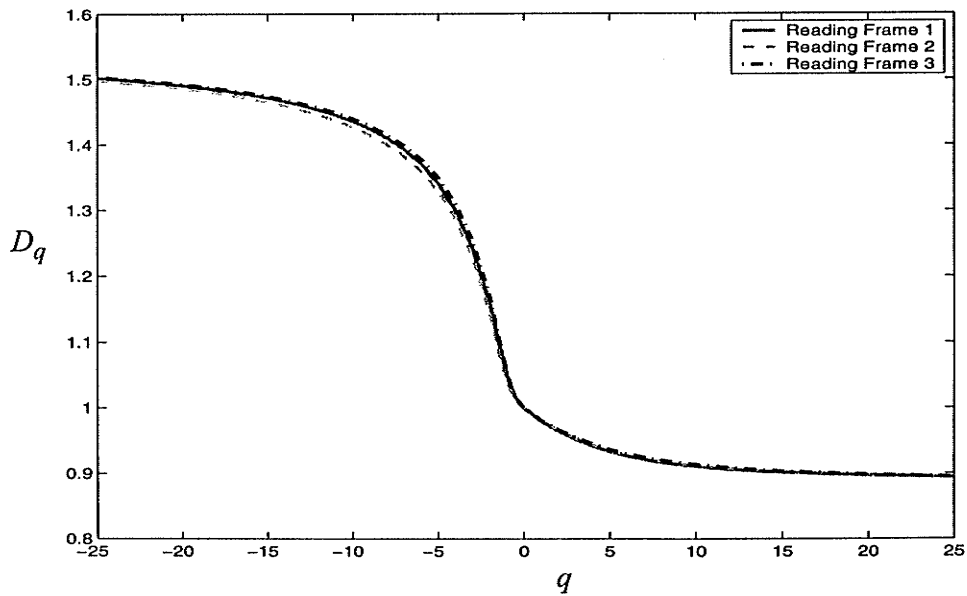


Fig. 6.14. The Rényi dimension spectra of the 3' flank region of the HYAL2 gene. The solid, dashed, and dot-dash lines represent the three frames, respectively.

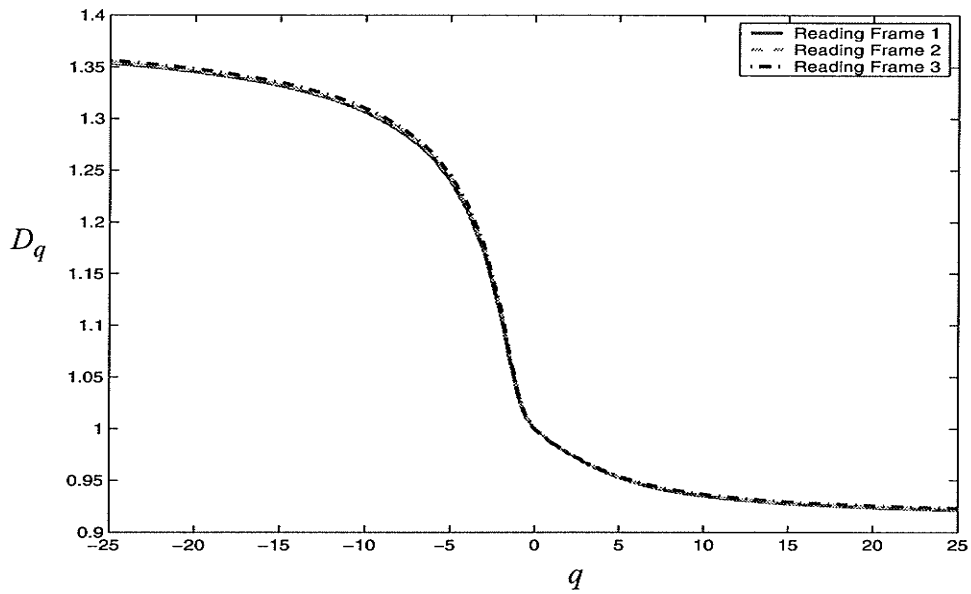


Fig. 6.15. The Rényi dimension spectra of the ph-20, a human genomic DNA sequence which contains the entire HYAL2 gene and the 5' flank region, the first exon and intron of the HYA11 gene. The solid, dashed, and dot-dash lines represent the three frames, respectively.

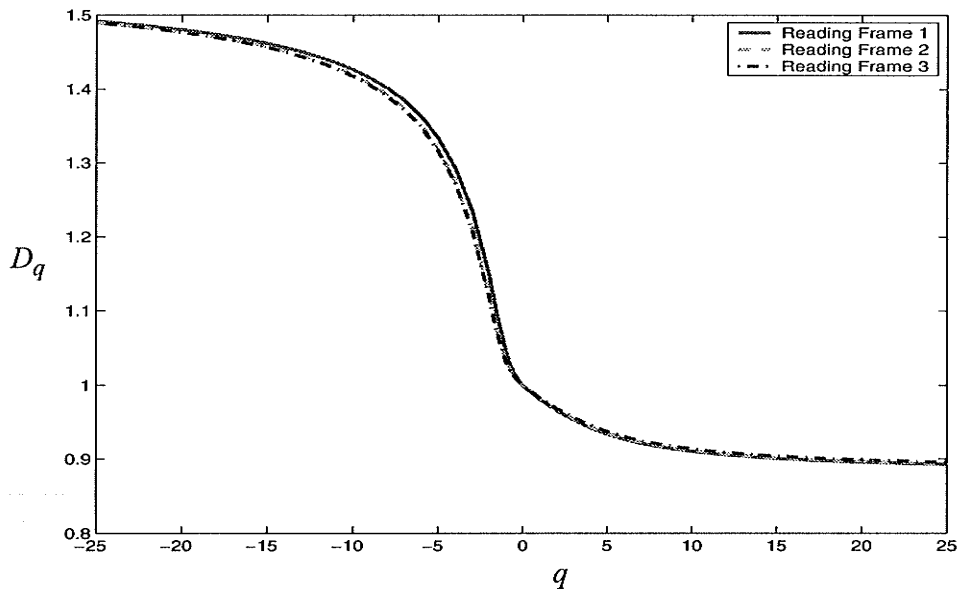


Fig. 6.16. The Rényi dimension spectra of the H19 gene. The solid, dashed and dot-dash lines represent the three frames, respectively.



by the fact, shown in Figs. 6.12, 6.13, and 6.14, that there are no significant difference of the Rényi dimension spectra among the three frames of non-coding sequences.

Although the ph-20, a human genomic DNA sequence, contains the whole HYAL2 gene and the 5' flank region, the first exon, and the first intron sequences of the HYAL1 gene, the result shows that it shares a common multifractal feature with the non-coding sequences and the random DNA sequence (Fig. 6.15). The reason for this is due to the fact that the total coding regions are less than 2000 bp or approximately 5% of the entire ph-20 sequence. Therefore, the sequence as a whole exhibits the statistical properties of a non-coding sequence.

The non-protein coding genes are also examined with our model. The expressed product of H19 gene, for example, is only untranslated mRNA and function as an RNA rather than a protein. Figures 6.16 shows the Rényi dimension spectra of the H19 cDNA sequence. As can be seen, the three frames of the gene have similar multifractality, suggesting none of them is a coding frame. Therefore, our established model is not able to distinguish the non-protein coding gene from non-coding background since the model is based on the bias usage of codons in a specie.

### 6.2.2 Mandelbrot Dimension Spectrum of the DNA Sequences

Figure 6.17 shows the Mandelbrot spectra of the generated random DNA sequences. The three frames of the random DNA sequence have a similar spectrum structure indicating that they have the same multifractality. In Fig. 6.18, the result shows that out of the three frames of the single fractal Cantor DNA sequence, the frame with a strict Cantor set property demonstrates a single fractal structure. The Mandelbrot dimension and

the Hölder exponent of the frame are constant and therefore, its Mandelbrot spectrum is degraded to a single point. The other frames show a slight multifractality since they are strictly not the Cantor set.

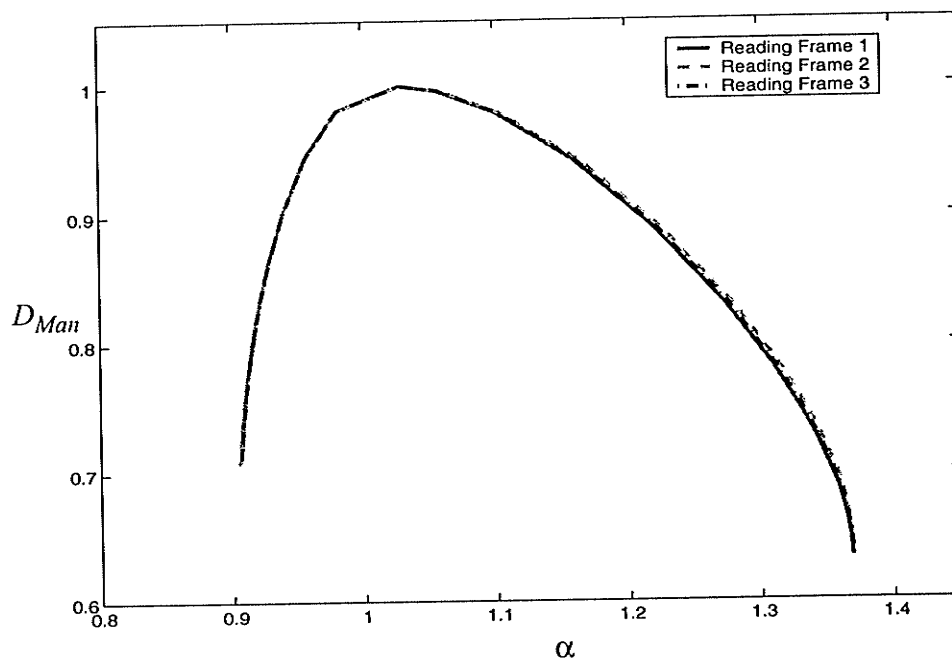


Fig. 6.17. The Mandelbrot spectra of the random DNA sequence. The solid, dashed, and dot-dash curves represent the three frames, respectively.

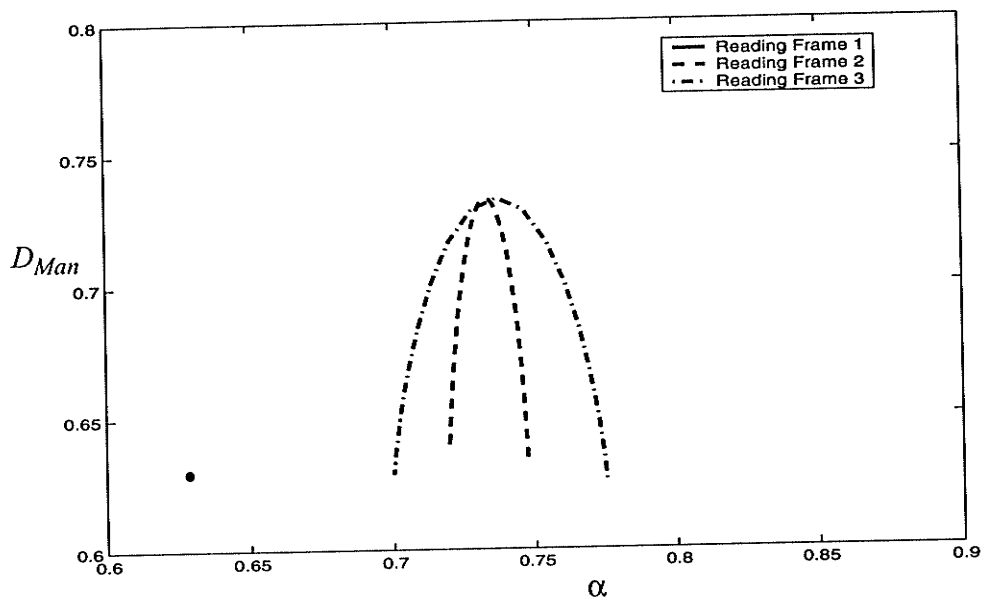


Fig. 6.18. The Mandelbrot spectra of the Cantor DNA sequence. The solid, dashed, and dot-dash curves represent the three frames, respectively. The solid curve, which represents the frame following the strict Cantor set property, deduced to a point since both the  $D_{Man}$  and  $\alpha$  are constant.

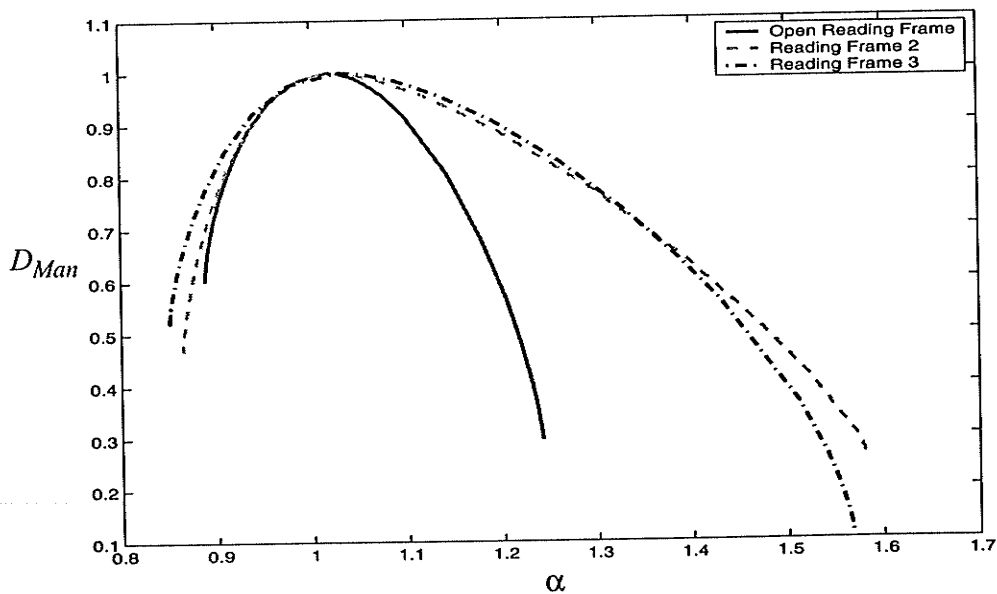


Fig. 6.19. The Mandelbrot Spectra of the HYAL2 cDNA sequence. The solid represents the ORF. The dashed and dot-dashed curves represent the other frames.

Figures 6.19 to 6.21 show the Mandelbrot spectra of the coding sequences. As shown in the pictures, the ORFs exhibit the significant different multifractality in comparison to the non-coding frames. However, for the non-coding sequences (shown in Figs. 6.22 to 6.24), there is no significant difference of the multifractalities among the three frames further proving an even codon usage in the non-coding frames.

It is interesting to note that the results shown in Figs. 6.13 and 6.22, suggest an even codon usage in the 5' flank region of the HYAL2 too. Similar results are also shown in the 5' flank regions of other genes. As mentioned in Chapter 2, the 5' flank regions of genes are GC-rich regions due to the presence of the multiple clusters of regulatory elements and promoter region. These results suggest that the GC-content is not the primary factor resulting in a bias codon usage in a sequence.

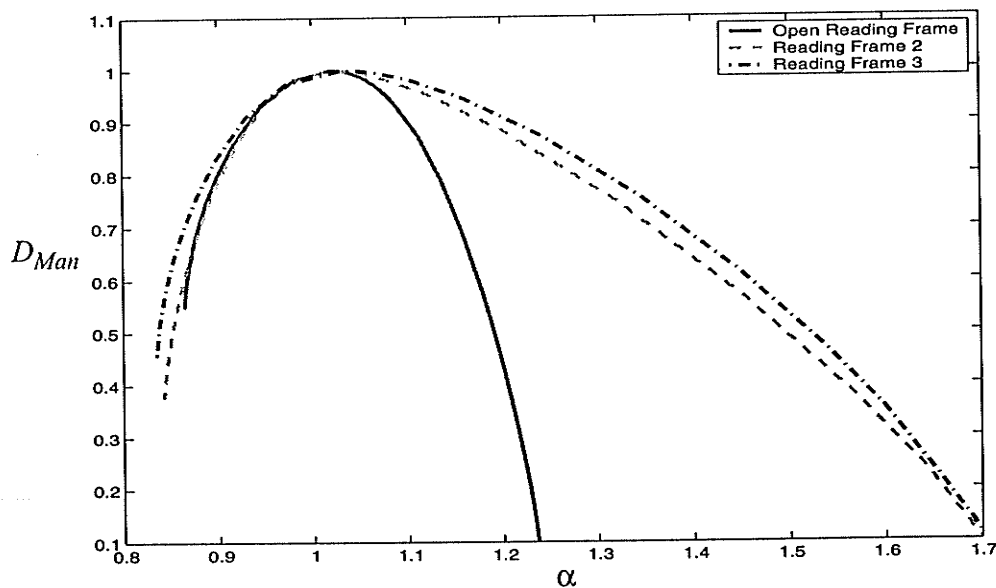


Fig. 6.20. The Mandelbrot Spectra of the third exon of the HYAL2 gene. The solid curve represents the ORF. The dashed and dot-dash curves represent the other frames.

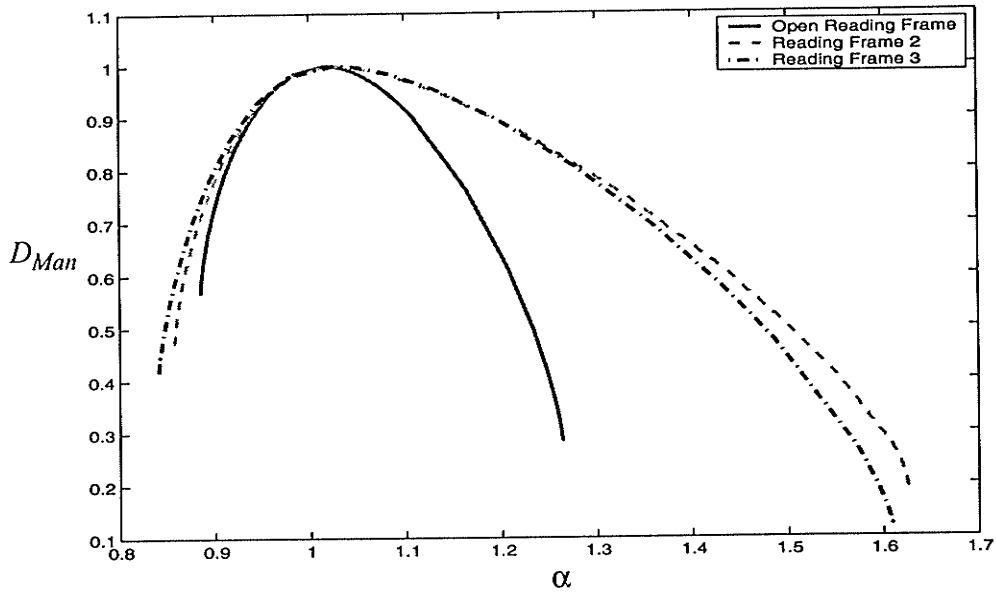


Fig. 6.21. The Mandelbrot Spectra of the first exon of the HYAL1 gene. The solid curve represents the ORF. The dashed and dot-dash curves represent the other frames.

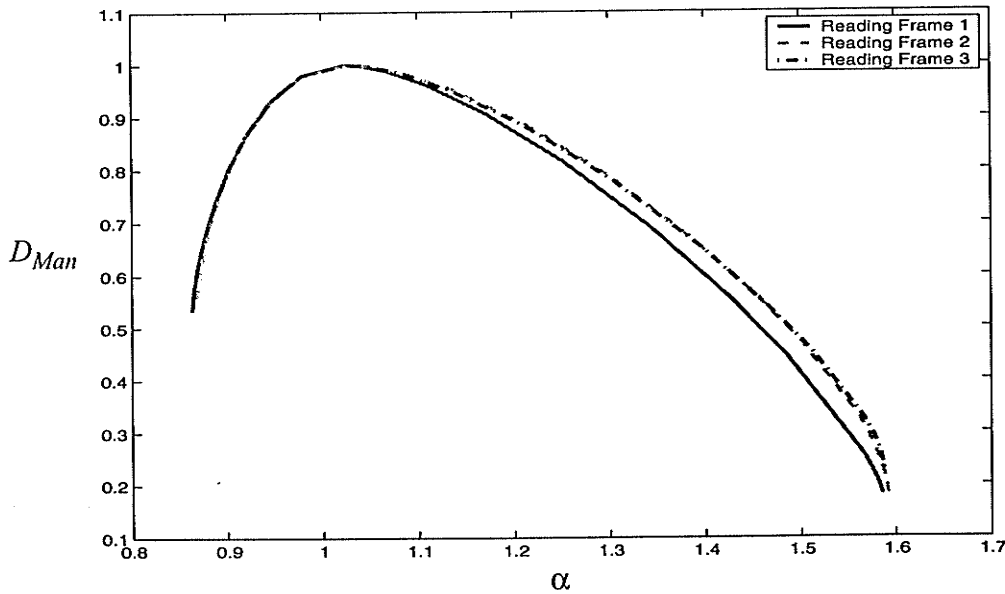


Fig. 6.22. The Mandelbrot Spectra of the 5' flank sequence of the HYAL2 gene. The solid curve represents the ORF. The dashed and dot-dash curves represent the other frames.

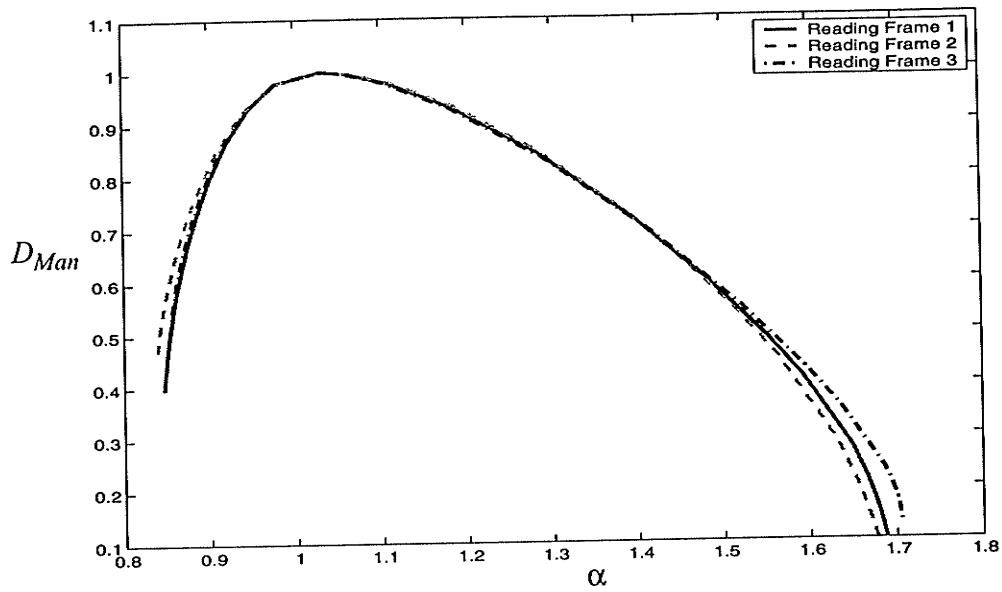


Fig. 6.23. The Mandelbrot Spectra of the first intron sequence of the HYAL2 gene. The solid curve represents the ORF. The dashed and dot-dash curves represent the other frames.

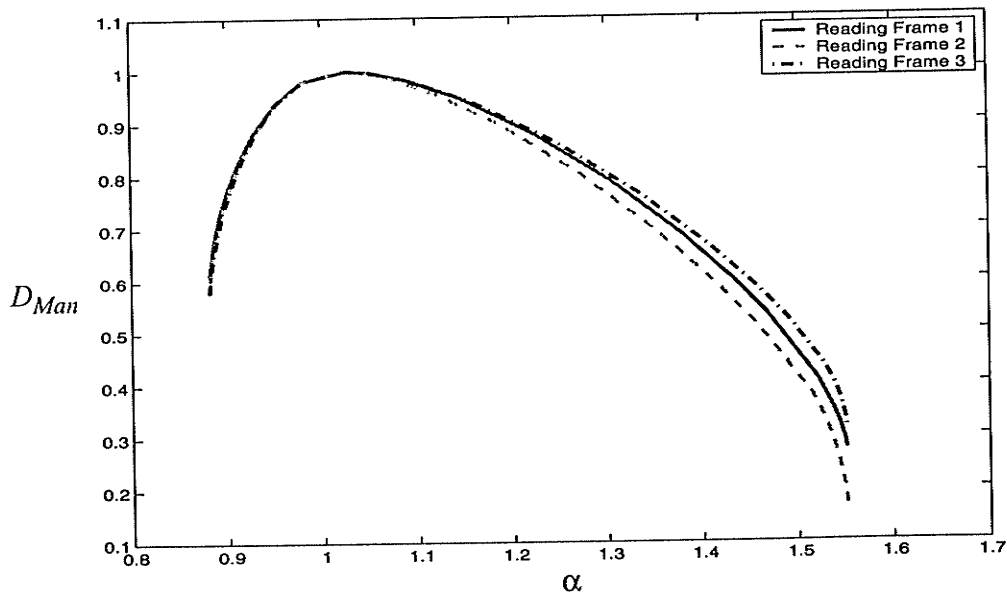


Fig. 6.24. The Mandelbrot Spectra of the 3' flank sequence of the HYAL2 gene. The solid curve represents the ORF. The dashed and dot-dash curves represent the other frames.

For the same reason as explained in previous section, the genomic DNA sequence, ph-20, exhibits a similar multifractal property of non-coding sequences although it contains the entire HYAL2 gene and the first exon of the HYAL1 gene (Fig. 6.25).

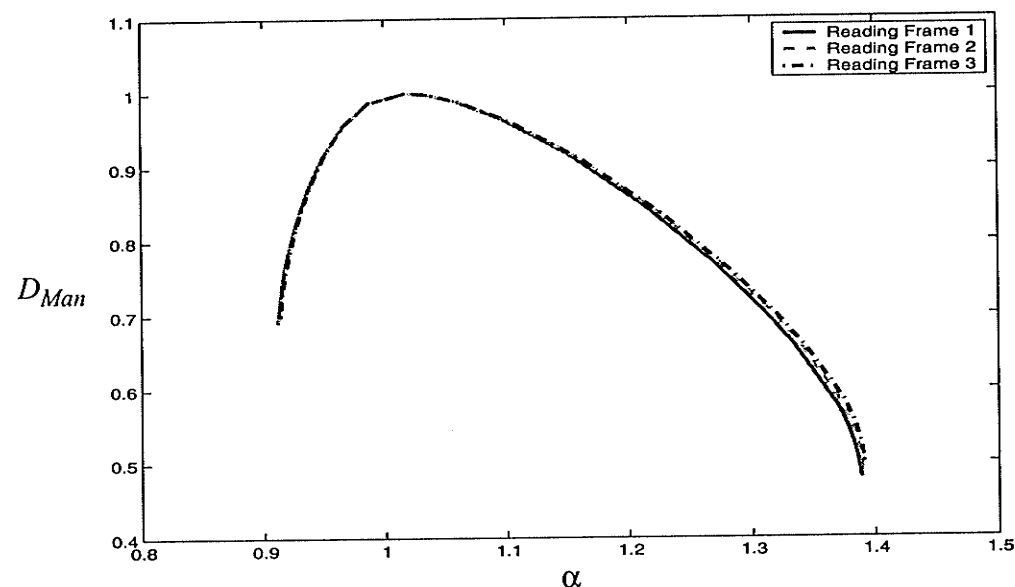


Fig. 6.25. The Mandelbrot Spectra of the genomic DNA sequence, ph-20. The solid, dashed, and dot-dash curves represent the three frames.

### 6.3 Local Rényi dimension Analysis of the DNA sequences

#### 6.3.1 Analysis With High Resolution Reveals Gene Structures

For the higher resolution local Rényi dimension analysis, a sliding window size of 122 bp is applied, as described in Sec. 4.4. The challenge of local Rényi dimension analysis of DNA sequences is that the coding regions of a gene does not always stay in the same reading frame. As described in Sec. 6.2, the experimental results show that Rényi dimen-

sion values of non-coding reading frames are significantly higher than that of the coding reading frame and there is no significant difference of the Rényi dimension values between two non-coding reading frames. Therefore, According to the experimental results shown in Sec. 6.2, a composite dimension,  $D^*$ , is computed based on

$$D^* = (D_{h1} + D_{h2})/2 - D_{low} \quad (6.1)$$

where  $D_{h1}$  and  $D_{h2}$  correspond to the highest local Rényi dimension values corresponding to two non-coding frames. The  $D_{low}$  corresponds to the smallest value of the third reading frame which may be related to a coding frame. In practice, the human  $\beta$ -globin gene sequence is used as the test sample. The actual location of the exons in the gene and the corresponding composite dimension,  $D^*$ , based on the local fractal dimension analysis,

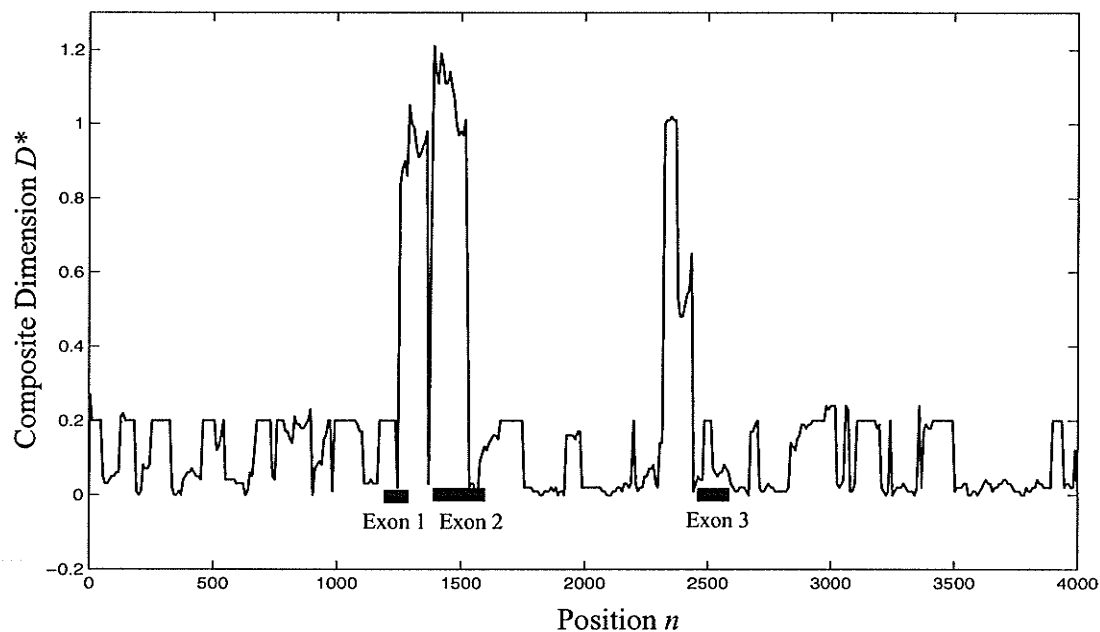


Fig. 6.26. Higher resolution local fractal dimension analysis of the  $\beta$ -globin gene. A sliding window size of 122 bp is used.



---

are plotted in Fig. 6.26. The position,  $n$ , represents the position of the first point of the sliding window in the DNA sequence.

As can be seen, the high value regions of  $D^*$  closely match the actual exon positions of the  $\beta$ -globin gene. Since the value of the  $n$ th point represents the fractality of a region with a window size of 122 bp located around the position  $n$ , the actual predicted coding regions should outspread 122 bp from the edge of the high computed value regions. If to do so, the predicted coding regions will cover the entire three exon regions of the  $\beta$ -globin gene.

### 6.3.2 Analysis With Low Resolution Reveals Genomic DNA Structures

To minimize the effect of noise and the interference of non-local fractals, a lower resolution with a window size of 400 bp is applied to the larger genomic DNA sequences. Figure 6.27 shows a local fractal dimension analysis of a 73 kb long human genomic DNA sequence, which contains the five globin genes and one pseudogene. There are several anomalies which are clearly visible from this diagram. First of all, the regions of all the genes are associated with the highly positive regions of the local fractal analysis. Secondly, there are several positive regions of the local fractal analysis which are not associated with the coding regions. Most of these positive regions appear in periods. For example, indicated as the dashed rectangle regions in Fig. 6.27, the non-coding positive regions are distributed with an interval of 2-2.5 kb in the genomic DNA sequence. The nature of these regions are not known. It may relate with the higher-order packing structure of chromosomes.

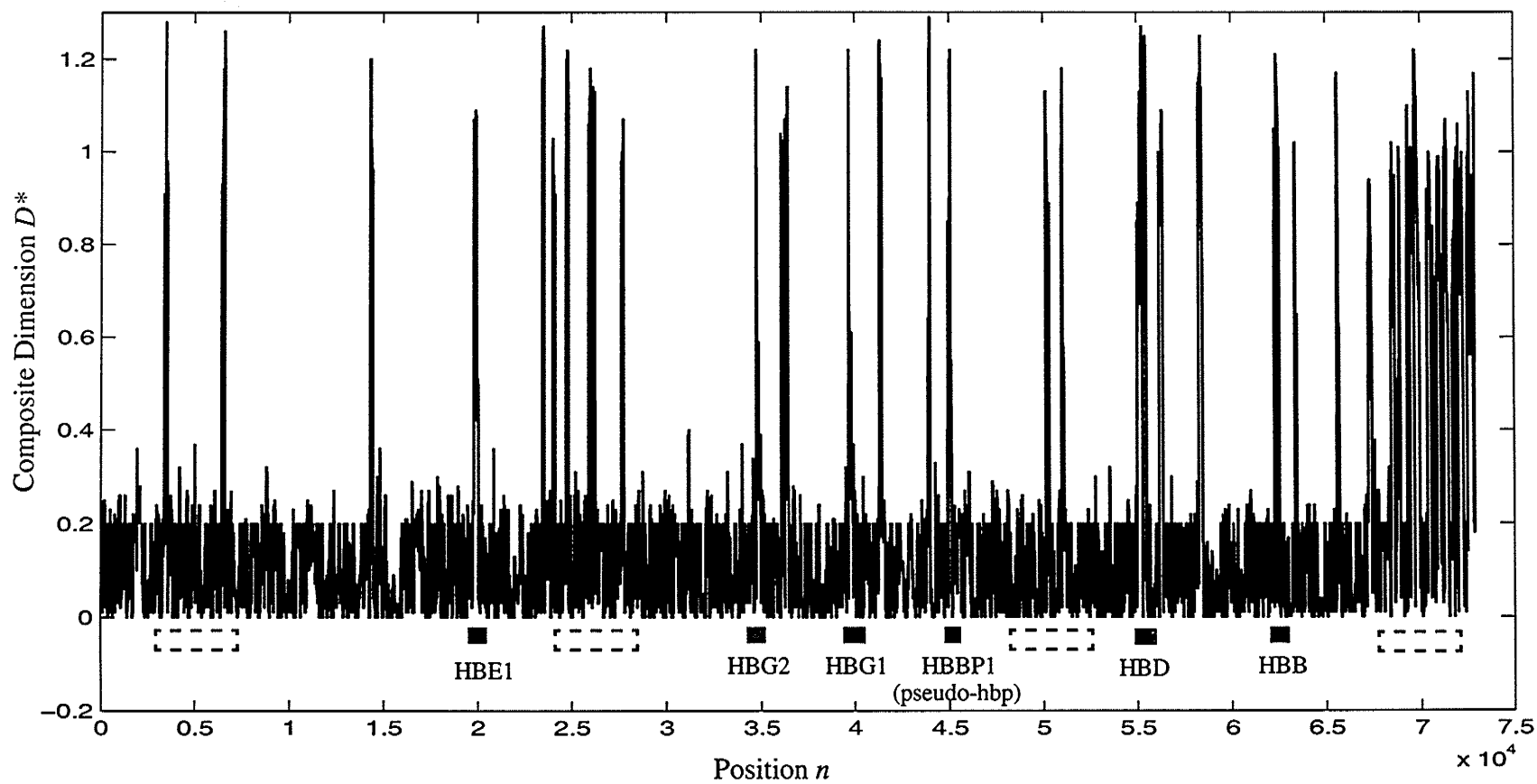


Fig. 6.27. Lower-resolusional local fractal dimension analysis of the human genomic DNA sequence. A sliding window size of 400 bp is used.

## 6.4 Summary

This chapter presented the results of the DNA sequence analysis. The mutual information and false nearest neighbourhood analysis provided a powerful tool for the chaotic characterization of DNA sequences. Like white noise, DNA sequences have a high-dimensional structure. There is a strong dependence within three bases in the DNA sequences, indicating a high correlation of among the three bases in a codon.

It is shown that the ORFs have a significant feature different from that of the non-coding reading frames, using Rényi dimension and Mandelbrot dimension spectra analysis. Local Rényi dimension analysis with different resolution is possible for the chromosome structure analysis and coding regions prediction.

---

## CHAPTER VII CONCLUSIONS

### 7.1 Conclusions

This thesis has presented (i) a review of the multifractal measures on the  $m$ -dimensional chaotic dynamical systems, and (ii) a development of the DNA sequences characterization based on multifractal techniques. A novel method of DNA symbolic sequence numerical mapping is established. The chaotic property of the DNA sequences are studied. A multifractal measure of the time series is formulated to extract the features of the DNA sequences. Finally, based on the above theory, an approach of the coding DNA sequence prediction was modelled and developed.

The experimental results indicate that there is a strong correlation within the three bases of the codons. The high-dimensional chaotic properties of the DNA sequences exhibited with the experiments suggests that DNA sequences have a high-dimensional structure.

It is shown in Chapter 6 that the multifractal spectrum of the DNA sequences can be used to distinguish the ORFs from the other non-coding reading frames. The experimental results demonstrate that for the DNA sequences, only the ORFs have the significant different multifractalities in comparison to the non-coding reading frames. All the non-coding frames have a similar multifractal property.

Using the local fractal dimension analysis, the multifractal features of the ORFs and the non-coding frames can be applied for characterization of the chromosomes and

---

coding prediction. Our experimental results of local fractal dimension analysis further exhibit that the interior structure of a gene can be revealed with a high resolution while a higher order structure of the genomic DNA sequences may be revealed in a low resolution.

## 7.2 Contributions

This thesis has provided the following contributions:

- The development of a new method for DNA symbolic sequences numerical mapping.
- A study of the chaotic property of the DNA sequences.
- A new and novel approach to characterization of the DNA sequences through frame sequence analysis, using multifractal techniques.
- A framework for modelling non-stationary  $m$ -dimensional signals as strange attractors.
- A study of multifractal measures for  $m$ -dimensional signals, especially for one-dimensional time series.
- The development of an approach for characterization of genomic DNA sequences and coding prediction.
- An application of the Rényi dimension spectrum for DNA sequence feature extraction and local structure analysis.
- A study of the relationship between the different resolutions and the structures of DNA sequences.

### 7.3 Recommendations

Based on the work of this thesis, the following recommendations are suggested for further research on this topic:

- Development of a systematic theory of the approximation of multifractal measures.
- Development of a systematic multifractal theory for measuring an  $m$ -dimensional signal.
- Further research in local fractal dimension analysis to reveal the correlation between the non-coding positive regions and the higher order packing structure of the chromosome.
- Further development of the approach for DNA coding prediction based on neural-network techniques.
- Development of an application for local DNA analysis, using the extracted features from the Mandelbrot spectrum.

---

**REFERENCES**

- [ArMo93] G. Arents and E. N. Moudrianakis, "Topography of the histone octamer surface: Repeating structural motifs utilized in the docking of nucleosomal DNA," *Proc. Natl. Acad. Sci. USA*, vol. 90, pp. 10489-10493, 1993.
- [ATVA01] B. Audit, C. Thermes, C. Vaillant, Y. d'Aubenton-Carafa, J. F. Muzy, and A. Arneodo, "Long-range correlations in genomic DNA: a signature of the nucleosomal structure," *Phys. Rev. Lett.*, vol. 86, pp. 2471-2474, 2001.
- [BoMc93] M. Borodovsky and J. McIninch, "GeneMark: Parallel Gene Recognition for both DNA Strands," *Comp. Chem.*, vol. 17, pp. 123-130, 1993.
- [BrKi86] D. S. Broomhead and G. P. King, "Extracting qualitative dynamics from experimental data," *Physica*, vol. 20D, pp. 217-236, 1986.
- [BuKa97] C. Burge and S. Karlin, "Prediction of complete gene structure in human genomic DNA," *J. Mol. Biol.*, vol. 268, pp. 78-94, 1997.
- [Bulm87] M. Bulmer, "Coevolution of codon usage and transfer RNA abundance," *Nature*, vol. 325, pp. 728-730, 1987.
- [Cant83] G. Cantor, "Über unendliche, lineare Punktmannigfaltigkeiten V," *Mathematische Annalen*, vol. 21, pp. 545-591, 1983.
- [CaSm99] C. R. Cantor and C. L. Smith, "Genomics," *John Wiley & Sons*, 596 pp., 1999.
- [Chen97] H. Chen, "Accuracy of fractal and multifractal measures for signal analysis," *M.Sc. Thesis, University of Manitoba, Winnipeg, manitoba, Canada*, 193 pp., 1997.
- [ChLa93] C. A. Chatzidimitriou-Dreismann and D. Larhammar, "Long-range correlations in DNA," *Nature*, vol. 361, pp. 212-213, 1993.
- [Clav97] J. M. Claverie, "Computational methods for the identification of genes in vertebrate genomic sequences," *Hum. Mol. Genet.*, vol. 6, pp. 1735-1744, 1997.
- [Clav00] J. M. Claverie, "From bioinformatics to computational biology," *Genome Res.*, vol. 10, pp. 1277-1279, 2000.
- [ClSB90] J. M. Claverie, I. Sauvaget, and L. Bougueleret, "k-tuple frequency analysis: from intron/exon discrimination to T-cell epitope mapping," *Meth. Enzymol.*, vol. 183, pp. 237-252, 1990.
-

- 
- [CUTG01] Codon Usage Tabulated from GenBank is maintained by the First Laboratory for Plant Gene Research, Kazusa DNA Research Institute. Available from (as of July 2001) [www.kazusa.or.jp/codon](http://www.kazusa.or.jp/codon).
- [Dans01] R. M. Dansereau, "Progressive image transmission using fractal and wavelet techniques with image complexity measures," *M.Sc. Thesis, University of Manitoba*, Winnipeg, manitoba, Canada, 278 pp., 2001.
- [DMAB91] G. D'Onofrio, D. Mouchiroud, B. Aissani, C. Gautier, and G. Bernardi, "Correlations between the compositional properties of human genes, codon usage, and amino acids composition of proteins," *J. Mol. Evol.*, vol. 32, pp. 504-510, 1996.
- [DOE01] Genomes to Life, Department of Energy, 2001.  
<http://www.DOEGenomesToLife.org/>
- [DoSe94] S. Dong and D. B. Searls, "Gene Structure Prediction by Linguistic Methods," *Genomics*, vol. 23, pp. 540-551, 1994.
- [DSRC99] I. Dunham, N. Shimizu, B. A. Rpe, S. Chissoe, *et al.*, "The DNA sequence of human chromosome 22," *Nature*, vol. 402, pp. 489-495, 1999.
- [Ehti99] T. Ehtiati, "Multifractal characterization of electromyogram signals," *M.Sc. Thesis, University of Manitoba*, Winnipeg, manitoba, Canada, 120 pp., 1999.
- [EMKS00] W. Ebeling, L. Molgedey, J. Kurths, and U. Schwarz, "Entropy, complexity, predictability and data analysis of time series and letter sequences," unpublished, 1997.  
available at: <http://summa.physik.hu-berlin.de/tsd>
- [Eyre96] A. Eyre-Walker, "Synonymous codon bias is related to gene length in *Escherichia coli*: selection for translational accuracy?" *Mol. Biol. Evol.*, vol. 13, pp. 864-887, 1996.
- [Fick96] J. W. Fickett, "The gene identification problem: an overview for developers," *Comput. Chem.*, vol. 20, pp. 103-118, 1996.
- [FiTu92] J. W. Fickett and C. S. Tung, "Assessment of protein coding measures," *Nucl. Acids Res.*, vol. 20, pp. 6441-6450, 1992.
- [FrSw86] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," *Phys. Rev.*, vol. 33A, no. 2, pp. 1134-1139, 1986.
- [GGGP80] R. Grantham, C. Gautier, M. Gouy, R. Mercier, and A. Pave, "Codon cata-
-



- log usage and the genome hypothesis," *Nucl. Acids Res.*, vol. 8, pp. r49-62, 1980.
- [GHBS00] I. Grosse, H. Herzel, S. V. Buldyrev, and H. E. Stanley, "Species independence of mutual information in coding and noncoding DNA," *Phys. Rev. E*, vol. 61, pp. 5624-5629, 2000.
- [GKDS92] R. Guigo, S. Knudsen, N. Drake, and T. Smith, "Prediction of gene structure," *J. Molecular Biology*, vol. 226, p141-157, 1992.
- [GMBG00] S. K. Gupta, S. Majumdar, T. K. Bhattacharya, and T. C. Ghosh, "Studies on the relationships between the synonymous codon usage and protein secondary structural units," *Biochem. Biophys. Res. Commun.*, vol. 269, pp. 692-696, 2000.
- [GrDB84] M. Gribskov, J. Devereux, and R. B. Burgess, "The codon preference plot: graphic analysis of protein coding sequences and prediction of gene expression," *Nucl. Acids Res.*, vol. 12, pp. 539-549, 1984.
- [GrHe99] T. R. Gregory and P. D. Hebert, "The modulation of DNA content: proximate causes and ultimate consequences," *Genome Res.*, vol. 9, pp. 317-324, 1999.
- [Grie96] W. S. Grieder, "Variance fractal dimension for signal feature enhancement and segmentation from noise," *M.Sc. Thesis, University of Manitoba, Winnipeg, manitoba, Canada*, 369 pp., 1996.
- [Guig97] R. Guigo, "Computational gene identification: an open problem," *Compu. Chem.*, vol. 21, pp. 215-222, 1997.
- [Hart00] D. L. Hartl, "Molecular melodies in high and low C," *Nature Rev. Genet.*, vol. 1, pp145-149, 2000.
- [HeGr95] H. Herzel and I. Grosse, "Measuring correlations in symbolic sequences," *Physica A*, vol. 216, pp. 518-542, 1995.
- [HeSF97] J. Henderson, S. Salzberg, and K. H. Fasman, "Finding genes in DNA with a hidden Markov model," *J. Computational Biology*, vol. 4, p127-141, 1997.
- [HGP01] International Human Genome Sequencing Consortium, "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, pp. 860-921, 2001.
- [HJKS86] T. C. Halsey, M. H. Jensen, L. P. Kadanoff, I. Procaccia, and B. Shraiman, "Fractal measures and their singularities: The characterization of strange sets," *Phys. Rev. A*, vol. 33, pp. 1141-1151, 1986.

- 
- [Holm86] L. Holm, "Codon usage and gene expression," *Nucleic Acids Res.*, vol. pp. 3075-3087, 1986.
- [HuHa92] G. Hutchinson and M. Hayden, "The prediction of exons through an analysis of spliceable open reading frames," *Nucleic Acids Res.*, vol. 20, p3453-3462, 1992.
- [HuKH92] M. A. Huynen, D. A. Konings, and P. Hogeweg, "Equal G and C contents in histone genes indicate selection pressures on mRNA secondary structure," *J. Mol. Evol.*, vol. 34, pp. 280-291, 1992.
- [HuYo93] B. R. Hunt and J. A. Yorke, "Maxwell on chaos," *Nonlinear Sci. Today*, vol. 3, pp. 1-4, 1993.
- [Ikem81] T. Ikemura, "Codon usage and tRNA content in unicellular and multicellular organisms," *Mol. Biol. Evol.*, vol. 2, pp. 13-34, 1985.
- [Jang97] E. Jang, "Compression of fingerprints based on wavelet packet decomposition and fractal singularity measures," *M.Sc. Thesis, University of Manitoba*, Winnipeg, manitoba, Canada, 206 pp., 1997.
- [JoSm87] D. W. Jordine and P. Smith, "Nonlinear ordinary differential equations," *NY: Oxford Applied Mathematics & Computing Science Series*, New York, 287 pp., 1987.
- [KaMr96] S. Karlin and J. Mrazek, "What drives codon choices in human genes?" *J. Mol. Biol.*, vol. 262, pp. 459-472, 1996.
- [KeBA92] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," *Phys. Rev. A*, vol. 45, pp. 3403-3411, 1992.
- [KHRE96] D. Kulp, D. Haussler, M. G. Reese, and F. H. Eeckman, "A generalized hidden Markov model for the recognition of human genes in DNA," In *proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology* (eds. D. States et al.), AAAI Press, Menlo Park, CA, USA, pp. 134-142, 1997.
- [Kins91] W. Kinsner, "Review of data compression methods, including Shannon-Fano, Huffman, arithmetic, Storer, Lempel-Ziv-Welch, Fractal, neural network, and wavelet algorithms," *Technical Report DEL91-1*, Dept. Electrical & Computer Engineering, University of Manitoba, 157pp., 1991.
- [Kins94] W. Kinsner, "Fractal dimensions: Morphological, entropy, spectrum, and variance classes," *Technical Report, DEL94-5*, University of Manitoba,
-

- 
- May 1994, 146 pp.
- [Kins95] W. Kinsner, "Fractal and chaos engineering," Course note, Dept. of Electrical and Computer Engineering, University of Manitoba, 1995.
- [KnFL01] R. D. Knight, S. J. Freeland, and L. F. Landweber, "A simple model based on mutation and selection explains trends in codon and amino-acid usage and GC composition within and across genomes," *Genome Biol.*, vol. 2, 2001.
- [Krog97] A. Krogh, "Two methods for improving performance of an HMM and their application for gene-finding," In *Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology* (eds. T. Gaasterland *et al.*), AAAI Press, Menlo Park, CA, USA, pp. 179-186, 1997.
- [Lang96] A. Langi, "Wavelet and fractal processing and compression of nonstationary signals," *Ph. D. Thesis, University of Manitoba, Winnipeg, Manitoba, Canada*, 227 pp., 1996.
- [LiYo75] T. -Y. Li and J. A. Yorke, "Period three implies chaos," *Am. Math. Mon.*, vol. 82, pp. 985-992, 1975.
- [Lobr97] J. R. Lobry, "Influence of genomic G+C content on average amino-acid composition of proteins from 59 bacterial species," *Gene*, vol. 205, pp. 309-316, 1997.
- [Lore63] E. N. Lorenz, "Deterministic nonperiodic flow," *J. Atmo. Sci.*, vol. 20, pp. 130-141, 1963.
- [LuBo98] A. V. Lukashin and M. Borodovsky, "GeneMark.hmm: New solutions for gene-finding," *Nucl. Acids Res.*, vol. 26, pp. 1107-1115, 1998.
- [Mand83] B. B. Mandelbrot, "The fractal geometry of nature," New York, NY: W. H. Freeman, 468 pp., 1983.
- [MBGS95] R. N. Mantegna, S. V. Buldyrev, A. L. Goldberger, S. Havlin, C. -K. Peng, M. Simons, H. E. Stanley, "Systematic analysis of coding and noncoding DNA sequences using methods of statistical linguistics," *Phys. Rev. E*, vol. 52, pp. 2939-2950, 1995.
- [NCBI01] GenBank statistics result, available at:  
<http://www.ncbi.nlm.nih.gov/GenBank/genbankstats.html>
- [NHGI01] National Human Genome Research Institute, web site available at:  
<http://www.nhgri.nih.gov/DIR/VIP/Glossary/Illustration>.
-

- 
- [OOYU88] S. Osawa, T. Ohama, F. Yamao, A. Muto, T. H. Jukes, H. Ozeki, and K. Umesono, "Directional mutation pressure and transfer RNA in choice of the third nucleotide of synonymous two-codon sets," *Proc. Natl. Acad. Sci. USA*, vol. 85, pp. 1124-1128, 1988.
- [PBGS92] C. K. Peng, S. Buldyrev, A. L. Goldberg, S. Havlin, F. Sciortino, M. Simons, and H. E. Stanley, "Long-range correlations in nucleotide sequences," *Nature*, vol. 356, pp. 168-171, 1992.
- [PCFS80] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw, "Geometry from a time series," *Phys. Rev. Lett.*, vol. 45, pp. 712-716, 1980.
- [PeJS92] H. Peitgen, H. Jürgens, and D. Saupe, "Chaos and fractals: new frontiers of science," *NY: Springer-Verlag*, New York, 894 pp., 1992.
- [PGHS92] C. K. Peng, S. Buldyrev, A. L. Goldberg, S. Havlin, F. Sciortino, M. Simons, and H. E. Stanley, "Long-range correlations in nucleotide sequences," *Nature*, vol. 356, pp. 168-171, 1992.
- [PrA197] A. Provata and Y. Almirantis, "Scaling Properties of Coding and Non-coding DNA Sequences," *Physica A*, vol. 247, pp. 482, 1997.
- [PrCl92] V. V. Prabhu and J. M. Claverie, "Correlations in intronless DNA," *Nature*, vol. 359, pp. 782-, 1992.
- [Rifa98] R. Rifaat, "Multifractal analysis of DNA," M. Sc. Thesis, University of Manitoba, Winnipeg, MB, Canada, 62 pp., 1998.
- [Shaw97] D. B. Shaw, "Classification of transmitter transients using fractal measures and probabilistic neural networks," *M.Sc. Thesis, University of Manitoba*, Winnipeg, manitoba, Canada, 375 pp., 1997.
- [Sind94] R. R. Sinden, "DNA structure and function," *Academic Press*, San Diego, California, 398 pp., 1994.
- [SnSt93] E. E. Snyder and G. D. Stormo, "Identification of coding regions in genomic DNA sequences: An application of dynamic programming and neural networks," *Nucleic Acids Res.*, vol. 21, p607-613, 1993.
- [SoSL94] V. Solovyev, A. Salamov, and C. Lawrence, "Predicting internal exons by oligonucleotide composition and discriminant analysis of spliceable open reading frames," *Nucleic Acids Res.*, vol. 22, p5156-5163, 1994.
- [StMc82] R. Staden and A. McLachlan, "Codon preference and its use in identifying protein coding regions in long DNA sequences," *Nucl. Acids Res.*, vol. 10, pp. 141-156, 1982.
-

- 
- [Take81] F. Takens, "Detecting strange attractors in turbulence," *Dynamical Systems and Turbulence*, D.A. Rand and L. S. Young, Eds., Springer-Verlag, pp. 366-381, 1981.
- [TRBR97] S. Tiwari, S. Ramachandran, A. Bhattacharya, S. Bhattacharya, and R. Ramaswamy, "Prediction of probable genes by Fourier analysis of genomic sequences," *Comput. Appl. Biosci.*, vol. 13, pp. 263-270, 1997.
- [Uber01] E. Uberbacher, "Bioinformatics-introduction," available at: <http://www.ornl.gov/ORNLReview/v30n3-4/genome.htm>
- [UbMu91] E. Uberbacher and R. Mural, "Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach," *Proc. Natl. Acad. Sci USA*, vol. 88, p11261-11265, 1991.
- [VAML01] J. C. Venter, M. D. Adams, E. W. Myers, and P. W. Li *et al.*, "The sequence of the human genome," *Science*, vol. 291, pp. 1304-1351, 2001.
- [Voss92] R. F. Voss, "Evolution of long-range fractal correlations and 1/f noise in DNA base sequences," *Phys. Rev. Lett.*, vol. 68, pp. 3805-3808, 1992.
- [WAAB94] R. Wilson, R. Ainscough, K. Anderson, C. Baynes, *et al.*, "2.2 Mb of contiguous nucleotide sequence from chromosome III of *C. elegans*," *Nature*, vol. 368, pp. 32-38, 1994.
- [WiVa99] V. Wilquet and M. Van de Castele, "The role of the codon first letter in the relationship between genomic GC content and protein amino acid composition," *Res. Microbiol.*, vol. 150, pp. 21-32, 1999.
- [Xia98] X. Xia, "How optimized is the translational machinery in *Escherichia coli*, *Salmonella typhimurium* and *Saccharomyces cerevisiae*?" *Genetics*, vol. 149, pp. 37-44, 1998.
- [YeLB01] R. Yeh, L. P. Lim, and C. B. Burge, "Computational inference of homologous gene structures in the human genome," *Genome Res.*, vol. 11, pp. 803-816, 2001.
- [YuAW01] Z. Yu, V. V. Anh, and B. Wang, "Correlation property of length sequences based on global structure of the complete genome," *Phys. Rev. E*, vol. 63, pp. 1-8, 2001.
- [YuAn01] Z. Yu and V. Anh, "Time series model based on global structure of complete genome," *Chaos, Solitons & Fractals*, vol. 12, pp. 1827-1834, 2001.
-

- 
- [ZhKi98] H. Zhang and W. Kinsner, "Feature extraction and fractal analysis of DNA sequences," *The 12<sup>th</sup> International Conference on mathematical & Computer Modeling and Scientific Computing*, 1999.

---

## APPENDICES

### A. Generated Sample Sequences

#### A.1 An example of the random DNA sequence

TAGGAATTAGGGGAAAAGAGTGGAAAGGGGGTGGGGAGAAAAGATAGGAAAGGAGAAA  
 GTAATAGTATTGTGGGAGGTGAGGGTGGTAAGTTGAGTTGATTGAAAATTAGGATGTT  
 TTGATAGAGGGGTAGAAGAGAAGGGGTAGGGTGTATATGAATGGTGAATGTGG  
 GTAAGTGTGTGTGAGGGAATTATGATTGTGGTTGAGTATTGGAGTGGAGGAAAGGTGTA  
 GGAAGGTGGAAGTTTTAGAGGAGTGTATTTTTATAGGTGTA AAAAGGGAAGAGGGTA  
 GTAGGGGGGATTGGGGTGGGGTGGAGGAGGTAGGTAGTGTAGAAGGTAAGTGGGGGAG  
 GGGGGTGGTTGGTGTAGAGGATGGTGAGTAAGGGGGGTAGTGGGAGATTAGAGGGGG  
 GGGGATTGGTGGGAAGGAGGGGTTTGTGGGGGGGTGGGTAGGTATGGGTATTATAGATA  
 AGGTGGGATGGGAGGTTGAATGGTGGAGGAGTTGTGGTGGAGGGTTGTTGTAAGGTA  
 GTAGGTTGGGGTGTGAGGGGAGGGGGGGGGGAAGGAATGGGATGAGGAGGTTAG  
 AGGTGGGTAGTAAGATAGGTAGGGGGATTAGGATGGATTGGATGGGGGTATGGGGGGTG  
 GGAGTGTGATTGTAGGTGGGAAGGGTGGGTTGGAGGTGGGTGAGTAGTTGTGTAGGGA  
 ATGTAAGGTTGTTGTAGAGGAGGGTAGGGGATGAAAGTGGTATGTGGGGATGTGAGGG  
 AGGGGGAGATGTAGGGTTAGGGAAGGATTGGGAAGAGGTGGGGATAAAGGGG  
 GGGTGTAGGGAAGGGGGTTGGGGGAGTGGTATTGAGTGAGTAAATTAAGTGGTGGT  
 GAATTAGTGGGGAGTGTGAGAATTGGGTAGAAGATAGTATGTAGGGAGATTGGGGGG  
 TGAGGGAGTTGGGTTGAGTGGGAGTGATTTTGGTGTGTGGGTTTATTGAGGTGTGAGT  
 TTATTGGTAAATATTAAGGATGGAAATGGATGAGATGATGAGTGGGGTTGGAGAGTA  
 GGTGAGAGGGTGTGGTGGTAGAGGGGTGAAAAGTTGTTAGTTGGTTGGTTTGAAGGG  
 AGGGTGGATGATGGGGGTTGGTGGGTAAGTGTGATAAATTGTAGGTGGGGGGGTGGAG  
 GTGGGAGGAAGTTTGGGATGAGAATTGAGGAGGAGAGGGTAGGATATGTGTGGTGTG  
 AGATGGAGTTGTTGGGATGGTTGTGTGGGAGGGTGTAAATAGAATTAGAGGGATTGAGGG  
 GGAATGGTGGTAAGAAAGGGGTGAGTGGAGGGATAAGAATAAATGTGAGGGGGGGGG

#### A.2 The Cantor DNA sequence

AAACCCAAACCCCCCCCCAAACCCAAACCCCCCCCCCCCCCCCCCCCCCCCCCCCCAAAC  
 CCAAACCCCCCCCCAAACCCAAACCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC  
 CCAACCCAAACCC  
 CCAAACCCAAACCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCAACCCAAACCC  
 CCC  
 CCC  
 CCC  
 CCCCCCCCCCCCCCCCCCAACCCAAACCCCCCCCAACCCAAACCCCCCCCCCCCCCC  
 CCCCCCCCCCCCCCCCCCAACCCAAACCCCCCCCAACCCAAACCCCCCCCCCCCCCC  
 CCC  
 CCCCCAAACCCAAACCCCCCCCAACCCAAACCCCCCCCCCCCCCCCCCCCCCCCCCC  
 CCAAACCCAAACCCCCCCCAACCCAAACCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC  
 CCC





---

**B. Source Code**

```
/*#####  
                                LorenzGen.c  
  
Discription:  This program generates Lorenz-II attractor of x, ro y, or z variable  
              time series. The length of the time series is inputs.  
  
Author:   Hong Zhang  
  
Version:   1.0  
Last Update:  May 2, 2001  
  
#####*/  
  
#include <math.h>  
#include <stdio.h>  
#include <stdlib.h>  
  
#define VARIABLE1           //1-x, 2-y, 3-z  
#define B                   4  
#define A                   0.25  
#define F                   8  
#define G                   1  
#define X_INITIAL          12  
#define Y_INITIAL          12  
#define Z_INITIAL          34  
  
const int iB = B;  
const int iF = F;  
const int iG = G;  
const double fA = A;  
const double fTimeStep = 0.05;  
  
void Lorenz(double, double, double);  
  
long iLen;  
double *fNewData;  
  
int main()  
{  
    int i;  
    double x, y, z;  
    FILE *fw;  
  
    printf("Enter the sequence length your want to be generated: ");
```

---

```

scanf("%d", &iLen);

x = X_INITIAL;
y = Y_INITIAL;
z = Z_INITIAL;

if( (fw = fopen("Lorenz.txt", "w")) == NULL)
{
    printf("Error of open the file.\n");
    exit(0);
}

fNewData = (double *)malloc(4*sizeof(double));
if(fNewData==NULL)
{
    printf("Error for space malloc of fNewData.\n");
    exit(0);
}
fprintf(fw, "%d%d\n", iLen, 0);
    //write out the 1st line which describes the length of the time series

for(i=0; i<iLen; i++)
{
    Lorenz(x,y,z);
    x = fNewData[0];
    y = fNewData[1];
    z = fNewData[2];
#if VARIABLE==1
    fprintf(fw, "%d%f\n", i+1, x);
#elif VARIABLE==2
    fprintf(fw, "%d%f\n", i+1, y);
#else VARIABLE==3
    fprintf(fw, "%d%f\n", i+1, z);
#endif
}
    fclose(fw);
return 0;
}

void Lorenz(double x, double y, double z)
{
    double dt, xt, yt, zt;
    dt = fTimeStep;
    xt = x + iSigma*(y-x)*dt;
    yt = y + (iR*x - x*z - y)*dt;
    zt = z + (x*y - fEita*z)*dt;

```

---

```

    fNewData[0] = xt;
    fNewData[1] = yt;
    fNewData[2] = zt;
}

/*#####
    contourGenerate.c

```

Discription: Contour set generator. Generate Contour set in DNA charactor sequence the smallest unit is depended on the user input.

Author: Hong Zhang

Version: 1.0

Last Update: Jan. 23, 2001

```

#####*/

```

```

#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <string.h>
#include <math.h>

```

```

#define LOWVALUE          0
#define HIGHVALUE         154
#define LINELENGTH        70
#define UPCHAR             'A'
#define LOWCHAR            'C'
#define OUTFILENAME        "contour"

```

```

char *charSeq_generator(int,int);
int *intSeq_generator(int, int);

```

```

int main()
{

```

```

    int i, k, iUnit, iSeqLength, iLength, iTemp=0, iLine, iOrder, *iContourSeq;

```

```

    char imfileC[25], imfileI[25], *FileName, *cContourSeq;
    FILE *outChar, *outInt;

```

```

    iUnit = 3;
    iSeqLength = 20000;

```

```

    /*    printf("Enter the number of bp for basic unit : ");
        scanf("%d", &iUnit);

```

---

```

printf("Enter the roughly length of the sequence : ");
scanf("%d", &iSeqLength);
*/
i = 0;
while( iTemp<(int)(iSeqLength/iUnit) )
iTemp = (int)pow(3, i++);
iOrder = i - 1;

/* generate characte sequence of Contour set*/
cContourSeq = charSeq_generator(iUnit, iOrder);

FileName = OUTFILENAME;
strcpy(imfileC, FileName);
strcat(imfileC, "_char");
if ((outChar = fopen(imfileC, "w" ) == NULL)
{
    printf("write error for charactor sequence.");
    exit(0);
}
iLength = (int)pow(3, iOrder)*iUnit;
iLine = LINELENGTH;
iTemp = (int)floor((float)iLength/iLine);
for(i=0; i<iTemp; i++)
{
    for(k=0; k<iLine; k++)
        fprintf(outChar, "%c", cContourSeq[i*iLine+k]);
    fprintf(outChar, "\n");
}
for(k=0; k<(int)(iLength%iLine); k++)
    fprintf(outChar, "%c", cContourSeq[iTemp*iLine+k]);

free(cContourSeq);
fclose(outChar);

/* generate time series sequence of Contour set*/
iContourSeq = intSeq_generator(iUnit, iOrder);

strcpy(imfileI, FileName);
strcat(imfileI, "_int");
if ((outInt = fopen(imfileI, "w" ) == NULL)
{
    printf("write error for the time series sequence.");
    exit(0);
}
for (i=0; i<iLength; i++)
    fprintf(outInt, "%d %d\n", i, iContourSeq[i]);

```

---

```
    free(iContourSeq);
    fclose(outInt);

    printf("The sequence length = %d\n", (int)pow(3, iOrder)*iUnit);
    return 0;
}

char *charSeq_generator(int iNumUnitBase, int iGrade)
{
    int i, k, j, iLen, iBuffSize;
    char *cTemp, *cSeq, cUpper, cLower, ch;

    cUpper = UPCHAR;
    cLower = LOWCHAR;
    iLen = (int)pow(3, iGrade);
    for(i=2; i<=iGrade; i++)
    {
        iBuffSize = (int)pow(3, i);
        cTemp = (char *)calloc(iBuffSize, sizeof(char));
        if(cTemp==NULL)
        {
            printf("Failed to malloc space for cTemp buffer.\n");
            exit(0);
        }

        iBuffSize = (int)pow(3, i-1);
        if(i==2)
        {
            cSeq = (char *)calloc(3, sizeof(char));
            cSeq[0] = cUpper;
            cSeq[1] = cLower;
            cSeq[2] = cUpper;
        }

        for(k=0; k<iBuffSize; k++)
        {
            ch = cSeq[k];
            switch(ch)
            {
                case UPCHAR:
                    cTemp[k*3] = cUpper;
                    cTemp[k*3+1] = cLower;
                    cTemp[3*k+2] = cUpper;
                    break;
                case LOWCHAR:
                    for(j=0; j<3; j++)
```

```

                                cTemp[3*k+j] = cLower;
                                break;
                            }
                        }
                    free(cSeq);

                    iBuffSize = (int)pow(3, i);
                    cSeq = (char*)calloc(iBuffSize, sizeof(char));
                    if(cSeq==NULL)
                    {
                        printf("Failed to malloc space for cSeq buffer.\n");
                        exit(0);
                    }
                    for(k=0; k<iBuffSize; k++)
                        cSeq[k] = cTemp[k];
                    free(cTemp);
                }

                iBuffSize *= iNumUnitBase;
                cTemp = (char*)calloc(iBuffSize, sizeof(char));
                if(cTemp==NULL)
                {
                    printf("Failed to malloc space for cTemp buffer.\n");
                    exit(0);
                }

                for(i=0; i<(int)pow(3, iGrade); i++)
                {
                    ch = cSeq[i];
                    switch(ch)
                    {
                        case UPCHAR:
                            for(k=0; k<iNumUnitBase; k++)
                                cTemp[i*iNumUnitBase+k] = cUpper;
                            break;
                        case LOWCHAR:
                            for(k=0; k<iNumUnitBase; k++)
                                cTemp[i*iNumUnitBase+k] = cLower;
                            break;
                    }
                }
                free(cSeq);
                return cTemp;
            }

int *intSeq_generator(int iNumUnitBase, int iGrade)

```

```
{
    int i, k, j, iLen, iBuffSize;
    int *iTemp, *iSeq, iUpper, iLower, ch;

    iUpper = HIGHVALUE;
    iLower = LOWVALUE;
    iLen = (int)pow(3, iGrade);
    for(i=2; i<=iGrade; i++)
    {
        iBuffSize = (int)pow(3, i);
        iTemp = (int *)calloc(iBuffSize, sizeof(int));
        if(iTemp==NULL)
        {
            printf("Failed to malloc space for iTemp buffer.\n");
            exit(0);
        }

        iBuffSize = (int)pow(3, i-1);
        if(i==2)
        {
            iSeq = (int *)calloc(3, sizeof(int));
            iSeq[0] = iUpper;
            iSeq[1] = iLower;
            iSeq[2] = iUpper;
        }

        for(k=0; k<iBuffSize; k++)
        {
            ch = iSeq[k];
            switch(ch)
            {
                case HIGHVALUE:
                    iTemp[k*3] = iUpper;
                    iTemp[k*3+1] = iLower;
                    iTemp[3*k+2] = iUpper;
                    break;
                case LOWVALUE:
                    for(j=0; j<3; j++)
                        iTemp[3*k+j] = iLower;
                    break;
            }
        }
        free(iSeq);

        iBuffSize = (int)pow(3, i);
        iSeq = (int *)calloc(iBuffSize, sizeof(int));
    }
}
```

---

```
        if(iSeq==NULL)
        {
            printf("Failed to malloc space for iSeq buffer.\n");
            exit(0);
        }
        for(k=0; k<iBuffSize; k++)
            iSeq[k] = iTemp[k];
        free(iTemp);
    }

    iBuffSize *= iNumUnitBase;
    iTemp = (int *)calloc(iBuffSize, sizeof(int));
    if(iTemp==NULL)
    {
        printf("Failed to malloc space for iTemp buffer.\n");
        exit(0);
    }

    for(i=0; i<(int)pow(3, iGrade); i++)
    {
        ch = iSeq[i];
        switch(ch)
        {
            case HIGHVALUE:
                for(k=0; k<iNumUnitBase; k++)
                    iTemp[i*iNumUnitBase+k] = iUpper;
                break;
            case LOWVALUE:
                for(k=0; k<iNumUnitBase; k++)
                    iTemp[i*iNumUnitBase+k] = iLower;
                break;
        }
    }
    free(iSeq);
    return iTemp;
}
```

---



---

```
/*#####
whiteNoiseGen.c
```

Discription: This program generates uniform white noise time series and random DNA sequence.

Author: Hong Zhang

Version: 1.0

Last Update: May 2001

```
#####*/
```

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
```

```
#define IA          16807
#define IM          2147483647
#define AM          (1.0/IM)
#define IQ          127773
#define IR          2836
#define NTAB        32
#define NDIV        (1+(IM-1)/NTAB)
#define EPS         1.2e-7
#define RNMX        (1.0-EPS)
#define CHAR_ARRAY  0
#define RANGE       402//402.19
```

```
float ran1(long*);
long iy=0, iv[NTAB], seqLen, *idum;
const int lineLenC = 80, lineLenI = 8;
```

```
int main()
{
```

```
    int i, k, j;
    float fTemp;
    char ch;
    FILE *fw;
```

```
    printf("Enter the sequence length your want to be generated (has to be times of %d): ", lineLenC);
    scanf("%d", &seqLen);
```

```
    idum = (long *)malloc(1*sizeof(long));
    *idum = -4;
    k = seqLen/lineLenC;
    if( (fw = fopen("randomSeq.txt", "w")) != NULL)
    {
```

---

```

#if CHAR_ARRAY == 1
    for(j=0; j<k; j++)
    {
        for(i=0; i<lineLenC; i++)
        {
            fTemp = ran1(idum);
            if(fTemp<0.25)
                ch = 'A';
            else if((fTemp>=0.25)&&(fTemp<0.50))
                ch = 'T';
            else if((fTemp>=0.50)&&(fTemp<0.75))
                ch = 'C';
            else
                ch = 'G';
            fprintf(fw, "%c", ch);
        }
        fprintf(fw, "\n");
    }
#else
    fprintf(fw, "%d%d\n", seqLen, 0); //1st line
    for(j=0; j<seqLen; j++)
    {
        fTemp = ran1(idum);
        fprintf(fw, "%f%f\n", fTemp, RANGE*fTemp);
    }
#endif
}
fclose(fw);
return 0;
}

float ran1(long *idum)
{
    int j;
    long k;
    float temp;

    if(*idum <= 0 || !iy)
    {
        if(-(*idum)<1)
            *idum = 1;
        else
            *idum = -(*idum);
        for(j=NTAB+7; j>=0; j--)
        {
            k = *idum/IQ;

```

---

```

        *idum = IA*( *idum - k*IQ) - IR*k;
        if(*idum<0)
            *idum += IM;
        if(j<NTAB)
            iv[j] = *idum;
    }
    iy = iv[0];
}
k = (*idum)/IQ;
*idum = IA*( *idum - k*IQ) - IR*k;
if(*idum<0)
    *idum += IM;
j = iy/NDIV;
iy = iv[j];
iv[j] = *idum;
temp=AM*iy;
if(temp > RNMX)
    return RNMX;
else
    return temp;
}

```

```

/*#####
mi.c

```

Discription: This program calculate the average mutaul information of a time series.  
This program uses only one grid size value and calculates all three frames.

Author: Hong Zhang

Version: 1.0

Last Update: Feb. 2001

```

#####*/

```

```

#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <string.h>
#include <math.h>

```

```

#define GRID_SIZE_FACTOR64

```

```

#define MAX_LAG200

```

```

float *FileIO(char*, int);
static int iSeqLen;
static float fMax, fMin;
float *fMI, fGridSize;

```

---

```

const float fGridSizeFactor = 1.0/(float)GRID_SIZE_FACTOR;

int main()
{
    int i, k, j, u, v, s, t, iGridNum, iLag, iLen, iLen1, iLen2, iLen3;
    int *iProb, *iProb1, *iColProb, *iColProb1, *iRowProb, *iRowProb1, *iProb2, *iProb3,
        *iColProb2, *iColProb3, *iRowProb2, *iRowProb3;
    char *cInFileName, *cOutFileName;
    float fSum, *fTimeSeries, fTemp;

    cInFileName = (char *)calloc(50, sizeof(char));
    cOutFileName = (char *)calloc(50, sizeof(char));
    if((cInFileName==NULL)||(cOutFileName==NULL))
    {
        printf("Error for malloc space to cInFileName or cOutFileName.\n");
        exit(0);
    }
    printf("\n Enter name of the data file: ");//get input file name
    scanf("%s",cInFileName);
    for(i=0; i<50; i++)//get output file name
    {
        if(cInFileName[i]!='.')
            cOutFileName[j] = cInFileName[i];
        else
            break;
    }
    strcat(cOutFileName, "_2.mi");
    fTimeSeries = FileIO(cInFileName, 0);//get time series

    //set up grid size and number
    fGridSize = (fMax - fMin)*fGridSizeFactor;
    fMI = (float *)calloc(MAX_LAG*4, sizeof(float));
    if(fMI==NULL)
    {
        printf("Error of malloc space for fMI.\n");
        exit(0);
    }
    iGridNum = (int)((fMax-fMin)/fGridSize) + 1;
    iColProb = (int *)malloc((iGridNum+1)*sizeof(int));
    iRowProb = (int *)malloc((iGridNum+1)*sizeof(int));
    iProb = (int *)malloc((iGridNum*iGridNum+1)*sizeof(int));
    iColProb1 = (int *)malloc((iGridNum+1)*sizeof(int));
    iRowProb1 = (int *)malloc((iGridNum+1)*sizeof(int));
    iProb1 = (int *)malloc((iGridNum*iGridNum+1)*sizeof(int));
    iColProb2 = (int *)malloc((iGridNum+1)*sizeof(int));
    iRowProb2 = (int *)malloc((iGridNum+1)*sizeof(int));

```

---

---

```

iProb2 = (int *)malloc((iGridNum*iGridNum+1)*sizeof(int));
iColProb3 = (int *)malloc((iGridNum+1)*sizeof(int));
iRowProb3 = (int *)malloc((iGridNum+1)*sizeof(int));
iProb3 = (int *)malloc((iGridNum*iGridNum+1)*sizeof(int));
if(((iProb==NULL)||(iProb1==NULL))|(((iRowProb==NULL)|(iRowProb1==NULL))|
  ((iColProb==NULL)|(iColProb1==NULL))))
{
    printf("Error of mallic space for iProb, iProb1, iRowProb, iRowProb1, iColProb1, or
      iColProb.\n");
    exit(0);
}
if(((iProb2==NULL)||(iProb3==NULL))|(((iRowProb2==NULL)|
  (iRowProb3==NULL))|(((iColProb2==NULL)|(iColProb3==NULL))))
{
    printf("Error of mallic space for iProb2, iProb3, iRowProb2, iRowProb3, iColProb2,
      or iColProb3.\n");
    exit(0);
}
for(iLag=1; iLag<=MAX_LAG; iLag++)
{
    iLen = iSeqLen - iLag;//do not consider frame
    for(k=0; k<iGridNum; k++)//clear up
    {
        iColProb[k] = 0;
        iRowProb[k] = 0;
        iColProb1[k] = 0;
        iRowProb1[k] = 0;
        iColProb2[k] = 0;
        iRowProb2[k] = 0;
        iColProb3[k] = 0;
        iRowProb3[k] = 0;
        for(u=0; u<iGridNum; u++)
        {
            iProb[u*iGridNum+k] = 0;
            iProb1[u*iGridNum+k] = 0;
            iProb2[u*iGridNum+k] = 0;
            iProb3[u*iGridNum+k] = 0;
        }
    }
    for(k=iLag; k<iSeqLen; k++)//compute P(k,j) without concerning the frames' effects
    {
        for(u=0; u<iGridNum; u++)
            if((fTimeSeries[k]>=(fMin+u*fGridSize))
              &&(fTimeSeries[k]<(fMin+(u+1)*fGridSize)))
                break;
        for(v=0; v<iGridNum; v++)

```

---

---

```

        if((fTimeSeries[k-iLag]>=(v*fGridSize+fMin))&&
           (fTimeSeries[k-iLag]<((v+1)*fGridSize+fMin)))
            break;
        iProb[u*iGridNum+v]++;
    }
//compute P(k,j) with concerning the codon frame's effect
for(k=3*iLag, iLen1=0, iLen2=0, iLen3=0; k<iSeqLen-2; k+=3)
{
    for(u=0; u<iGridNum; u++)
        if((fTimeSeries[k]>=(u*fGridSize+fMin))&&
           (fTimeSeries[k]<((u+1)*fGridSize+fMin)))
            break;
    for(v=0; v<iGridNum; v++)
        if((fTimeSeries[k-3*iLag]>=(v*fGridSize+fMin))&&
           (fTimeSeries[k-3*iLag]<((v+1)*fGridSize+fMin)))
            break;
    iLen1++;
    iProb1[u*iGridNum+v]++;

    for(u=0; u<iGridNum; u++)
        if((fTimeSeries[k+1]>=(u*fGridSize+fMin))&&
           (fTimeSeries[k+1]<((u+1)*fGridSize+fMin)))
            break;
    for(v=0; v<iGridNum; v++)
        if((fTimeSeries[k+1-3*iLag]>=(v*fGridSize+fMin))&&
           (fTimeSeries[k+1-3*iLag]<((v+1)*fGridSize+fMin)))
            break;
    iLen2++;
    iProb2[u*iGridNum+v]++;
    for(u=0; u<iGridNum; u++)
        if((fTimeSeries[k+2]>=(u*fGridSize+fMin))&&
           (fTimeSeries[k+2]<((u+1)*fGridSize+fMin)))
            break;
    for(v=0; v<iGridNum; v++)
        if((fTimeSeries[k+2-3*iLag]>=(v*fGridSize+fMin))&&
           (fTimeSeries[k+2-3*iLag]<((v+1)*fGridSize+fMin)))
            break;
    iLen3++;
    iProb3[u*iGridNum+v]++;
}
for(u=0; u<iGridNum; u++)
{
    for(v=0; v<iGridNum; v++)
    {
        iColProb[u] += iProb[u*iGridNum+v]; //compute P(k)
        iRowProb[v] += iProb[u*iGridNum+v]; //compute P(j)
    }
}

```

---

---

```

        iColProb1[u] += iProb1[u*iGridNum+v];
                                //compute P(k) in the coding frame
        iRowProb1[v] += iProb1[u*iGridNum+v];
                                //compute P(j) in the coding frame
        iColProb2[u] += iProb2[u*iGridNum+v];
        iRowProb2[v] += iProb2[u*iGridNum+v];
        iColProb3[u] += iProb3[u*iGridNum+v];
        iRowProb3[v] += iProb3[u*iGridNum+v];
    }
}

for(u=0; u<iGridNum; u++)
{
    if(iColProb[u]>0)
    {
        for(v=0; v<iGridNum; v++)
        {
            if((iRowProb[v]>0)&&(iProb[u*iGridNum+v]>0))
            {
                fSum =
                (float)(iProb[u*iGridNum+v]*iLen)/(iColProb[u]*iRowProb[v]);
                fMI[4*(iLag-1)]+=
                (float)(iProb[u*iGridNum+v]*log(fSum)/iLen);
            }
        }
    }
    if(iColProb1[u]>0)
    {
        for(v=0; v<iGridNum; v++)
        {
            if((iRowProb1[v]>0)&&(iProb1[u*iGridNum+v]>0))
            {
                fSum=
                (float)iProb1[u*iGridNum+v]*iLen1/(iColProb1[u]*iRowProb1[v]);
                fMI[4*(iLag-1)+1]+=
                (float)(iProb1[u*iGridNum+v]*log(fSum)/iLen1);
            }
        }
    }
    if(iColProb2[u]>0)
    {
        for(v=0; v<iGridNum; v++)
        {
            if((iRowProb2[v]>0)&&(iProb2[u*iGridNum+v]>0))
            {
                fSum=

```

---

```

        (float)(iProb2[u*iGridNum+v]*iLen2)/(iColProb2[u]*iRowProb2[v]);
        fMI[4*(iLag-1)+2]+=
            (float)(iProb2[u*iGridNum+v]*log(fSum)/iLen2);
    }
}
}
if(iColProb3[u]>0)
{
    for(v=0; v<iGridNum; v++)
    {
        if((iRowProb3[v]>0)&&(iProb3[u*iGridNum+v]>0))
        {
            fSum=
                (float)iProb3[u*iGridNum+v]*iLen3/(iColProb3[u]*iRowProb3[v]);
            fMI[4*(iLag-1)+3]+=
                (float)(iProb3[u*iGridNum+v]*log(fSum)/iLen3);
        }
    }
}
}
free(iProb);
free(iColProb);
free(iRowProb);
free(iProb1);
free(iColProb1);
free(iRowProb1);
free(iProb2);
free(iColProb2);
free(iRowProb2);
free(iProb3);
free(iColProb3);
free(iRowProb3);
FileIO(cOutFileName, 1);
return 1;
}

float *FileIO(char *cFileName, int iNum)
{
    int i, k, iLen, iTemp;
    float *fSeq;
    float fTemp, fArray[8], ff;
    char ch;
    FILE *inFile, *outFile;

    if(iNum==0)//open, read the input filecFileName

```



```

{
    if((inFile=fopen(cFileName, "r")) == NULL)
    {
        printf("\n Failed to open the input file\n");
        exit(0);
    }
    fseek(inFile, 0, SEEK_END);
    iLen = ftell(inFile);
    rewind(inFile);
    fSeq = (float*)calloc(iLen+1, sizeof(float));
    if(fSeq==NULL)
    {
        printf("Error of malloc space for fSeq.\n");
        exit(0);
    }
    rewind(inFile);
    while(ch!='\n')//discard the first line
        ch = getc(inFile);
    i = 0;
    fMin = 99999999.0;
    fMax = 0.0;
    while(fscanf(inFile, "%d%f", &iTemp, &fTemp)==2)
    {
        fSeq[i++] = fTemp;
        if(fMax<fTemp)
            fMax = fTemp;
        if(fMin>fTemp)
            fMin = fTemp;
    }
    fclose(inFile);
    iSeqLen = i;
    return fSeq;
}
else //open, write to the output file
{
    if((outFile=fopen(cFileName, "w")) == NULL)
    {
        printf("\n Failed to open the output file\n");
        exit(0);
    }
    fprintf(outFile, "%d%f%d%d%d\n", MAX_LAG, fGridSize, 1, 2, 3);
    for(i=1; i<=MAX_LAG; i++)
        fprintf(outFile, "%d%f%f%f%f\n", i, fMI[4*(i-1)], fMI[4*(i-1)+1],
                fMI[4*(i-1)+2], fMI[4*(i-1)+3]);

    fSeq = (float *)malloc(1*sizeof(float));
    fSeq[0] = -1.0;
}

```

```

        fclose(outFile);
        return fSeq;
    }
}

/*#####
                                FNN.c

```

Discription: This program calculate the percentage of the fase nearest neighbours of a time series, using the Kennel method.

Author: Hong Zhang

Version: 1.0

Last Update: May 2001

```
#####*/
```

```

#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <string.h>
#include <math.h>
#define NUM_FRAME          0//0-not frame, 1-frame 1, 2-frame 2, 3-frame3
#define ABS(a)              ((a)>=0) ? (a) : (-(a))

```

```

float *FileIO(char*, int, float*, int);
const int iMaxA = 3; //MAX_A_TOTAL;
const int iNumM = 10; //MAX_EMBEDDING_DIMENSION;
const int iMaxR = 30; //MAX_R_TOTAL;
const float fFactorR = 0.5000; //R_TOTAL_FACTOR;
const float fFactorA = (float)0.2000; //A_TOTAL_FACTOR;

```

```

static int iSeqLen, iNumRtol, iNumAtol, iLag;
float fR[50], fA[50], *fFNN;

```

```
int main()
```

```

{
    int i, k, m, u, r, j, iStatus=0, iFalse;
    float *fTimeSeries, *fMinD, *fNextD, *fND;
    char *cInFileName, *cOutFileName;
    float fTemp, fRa, fMean, fR_m, fStdDev, fMinDistance, fNextDist;

    cInFileName = (char *)malloc(50*sizeof(char));
    cOutFileName = (char *)malloc(50*sizeof(char));
    if((cInFileName==NULL)||(cOutFileName==NULL))
    {
        printf("Error for malloc space to cInFileName or cOutFileName.\n");
    }
}

```

---

```

        exit(0);
    }
    printf("\n Enter name of the data file: "); //get input file name
    scanf("%s", cInFileName);
    printf("\n Enter lag value for the data file: ");
    scanf("%d", &iLag);

    for(i=0; i<50; i++) //get output file name
    {
        if(cInFileName[i]!='.')
            cOutFileName[i] = cInFileName[i];
        else
            break;
    }
    strcat(cOutFileName, ".fnnK");
    for(i=0, iNumRtol=0; ; i++)
    {
        if(i<2)
            fR[i] = fFactorR*(1+i);
        else
            fR[i] = 10*fFactorR*(i-1);
        if(fR[i]>iMaxR)
            break;
        iNumRtol++;
    }
    for(i=1, iNumAtol=0; ; i++)
    {
        if(i<6)
            fA[i-1] = fFactorA*i;
        else
            fA[i-1] = (float)0.5*(i-3);
        if(fA[i-1]>iMaxA)
            break;
        iNumAtol++;
    }
    fTimeSeries = FileIO(cInFileName, 0, fFNN, iStatus); //get time series
    fFNN = (float *)calloc(iNumRtol*iNumM*iNumAtol+1, sizeof(float));
    fMinD = (float *)malloc((iSeqLen-iLag)*iNumM*sizeof(float));
    fNextD = (float *)malloc((iSeqLen-iLag)*iNumM*sizeof(float));
    if(((fFNN==NULL)||((fMinD==NULL)||((fNextD==NULL))))
    {
        printf("Error of malloc space for fFNN, or fMinD, or fNextD.\n");
        exit(0);
    }
    for(i=0, fRa=0.0; i<iSeqLen; i++)
        fRa += fTimeSeries[i];

```

---

---

```

fMean = fRa/iSeqLen;
for(i=0, fRa=0.0; i<iSeqLen; i++)
    fRa += (fTimeSeries[i]-fMean)*(fTimeSeries[i]-fMean);
fStdDev = (float)sqrt(fRa/iSeqLen); //compute the standard deviation of the data
for(m=1; m<=iNumM; m++)
{
    for(i=iLag*m, j=0; i<iSeqLen; i++)
    {
        fMinDistance = (float)99999999.0;
        for(u=iLag*m; u<iSeqLen; u++) //find the nearest neighbour
        {
            if(i!=u)
            {
                fR_m = 0;
                for(k=0; k<m; k++)
                {
                    fTemp=
                        fTimeSeries[i-iLag*k]-fTimeSeries[u-iLag*k];
                    fR_m += fTemp*fTemp;
                }
                if(fMinDistance>fR_m)
                {
                    fMinDistance = fR_m;
                    fNextDist =
                        fTimeSeries[i-iLag*m]-fTimeSeries[u-iLag*m];
                }
            }
        }
        fMinD[iNumM*j+m-1] = (float)sqrt(fMinDistance);
        if(fNextDist<0)
            fNextD[iNumM*j+m-1] = -fNextDist;
        else
            fNextD[iNumM*j+m-1] = fNextDist;
        j++;
    }
}
for(r=0; r<iNumRtol; r++)
{
    for(j=0; j<iNumAtol; j++)
    {
        for(m=1; m<=iNumM; m++)
        {
            iFalse=0;
            for(i=iLag*m, k=0; i<iSeqLen; i++)
            {

```

---

```

        if(fMinD[iNumM*k+m-1]!=0)//criterion I
        {
            fTemp =
                fNextD[iNumM*k+m-1]/fMinD[iNumM*k+m-1];
            if(fTemp>fR[r])
                iFalse++;
            else {
                fTemp =
                    (float)sqrt(fNextD[iNumM*k+m-1]*fNextD[iNumM*k+m-1]+fMinD
                        [iNumM*k+m-1]*fMinD[iNumM*k+m-1]);
                if((fTemp/fStdDev)>fA[j])//criterion II
                    iFalse++;
            }
        }
        else
            if(fNextD[iNumM*k+m-1]!=0)
                iFalse++;
        k++;
    }
    fFNN[iNumM*(iNumAtol*r+j)+m-1] = (float)iFalse/(float)k;
}
}
}
FileIO(cOutFileName, 1, fFNN, 0);//output the result
free(fFNN);
return 1;
}

float *FileIO(char *cFileName, int iNum, float *fOutputData, int status)
{
    int i, k, j, iLen, iTemp;
    float *fSeq, fTemp;
    FILE *inFile, *outFile;

    if(iNum==0)//open, read the input file cFileName
    {
        if((inFile=fopen(cFileName, "r")) == NULL)
        {
            printf("\n Failed to open the input file\n");
            exit(0);
        }
        fseek(inFile, 0, SEEK_END);
        iLen = ftell(inFile);
        rewind(inFile);
        fSeq = (float*)calloc(iLen+1, sizeof(float));
        if(fSeq==NULL)

```

```

    {
        printf("Error of malloc space for fSeq.\n");
        exit(0);
    }
    if(fscanf(inFile, "%d%d", &i, &iTemp)==2);
        i = 0;
    while(fscanf(inFile, "%d%f", &iTemp, &fTemp)==2)
        fSeq[i++] = fTemp;
    if(status>0)
    {
        for(k=0, j=0; k<i; k+=3)
            fSeq[j++] = fSeq[k+status-1];
        iSeqLen = j;
    }
    else
        iSeqLen = i;
    fclose(inFile);
    return fSeq;
}
else//open, write to the output file
{
    if((outFile=fopen(cFileName, "w")) == NULL)
    {
        printf("\n Failed to open the output file\n");
        exit(0);
    }
    fprintf(outFile, "%d%d%d", iNumRtol, iNumM, iNumAtol);
    for(i=1; i<=iNumRtol-2; i++)
        fprintf(outFile, "%d", 0);//first line
    fprintf(outFile, "\n");
    for(i=0; i<iNumAtol; i++)
    {
        fprintf(outFile, "%.1f", fA[i]);
        for(k=0; k<iNumRtol; k++)
            fprintf(outFile, "%.1f", fR[k]);
        fprintf(outFile, "\n");
        for(j=0; j<iNumM; j++)
        {
            fprintf(outFile, "%d", j+1);
            for(k=0; k<iNumRtol; k++)
                fprintf(outFile, "%.2f", 100*fFNN
                    [iNumM*(iNumAtol*k+i)+j]);
            fprintf(outFile, "\n");
        }
    }
}
//fOutputData0.005+

```

```

        fSeq = (float *)calloc(1, sizeof(float));
        fSeq[0] = -1.0;
        fclose(outFile);
        return fSeq;
    }
}

#####
                                codUgTimeSeriesReniMan.c

Discription:  Loads the data file, calculates the Renyi Dimension, singularity spectrum,
              and Mandbort Dimension.

Author:       Hong Zhang
Version:      1.0
Last Update:  Jun. 2001
#####*/

#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <string.h>
#include <math.h>

#ifdef CONTOUR_INT
    #define CONTOUR
#endif
#define DELTA          0.01
#define RANGE          25

float* loadTimeSeries(char*);
void renyCalc(float*, char*);
long fileLength;

int main()
{
    int i;
    char imfile[80];
    float *data;

    printf("\n Enter name of the data file: ");
    scanf("%s",imfile);
    //*****
#ifdef CONTOUR_INT
    data = loadTimeSeries(imfile);
    printf("Sequence length = %d\n", fileLength);

```

```

        renyCalc(data, imfile);
        return 0;
#endif
//*****

        data = loadTimeSeries(imfile);
        renyCalc(data, imfile);
        return 0;
}

float* loadTimeSeries(char* fileName)
{
    FILE *inFile;
    int i=0, k, iTemp[2];
    float *fSeq, fTemp[3];

    if((inFile = fopen(fileName, "r")) == NULL)
    {
        printf("Failed to open the time series data file\n");
        exit(0);
    }
    fseek (inFile, 0, SEEK_END);
    fileLength = ftell(inFile);
    rewind(inFile);
    fSeq = (float *)calloc(fileLength+1, sizeof(float));
    if(fSeq==NULL)
    {
        printf("Failed to malloc for the iSeq buffer.\n");
        exit(0);
    }

    while(fscanf(inFile, "%d%f%f%f", &iTemp[0],&fTemp[0],&fTemp[1],&fTemp[2])==4)
        fSeq[i++] = (float)fTemp[0];
    fclose(inFile);
    fileLength = i;
    printf("fileLength = %d\n", fileLength);
    return fSeq;
}

void renyCalc(float* signalSeq, char* imfile)
{
    float *codeBook, *renyDim, *mandbort, *singuSpc;
    float fQ[2], delta;
    double dTmp, tem[2], logTemp[2], totalN[2], temp1, temp2;
    int i, j, k, q, len, temp, subQ, factor, x;
    char outfile[50];

```



```

FILE *out;

delta = DELTA;
fQ[0] = 1 - delta;
fQ[1] = 1 + delta;
singuSpc = (float *)calloc((RANGE*2+7), sizeof(float));
mandbort = (float *)calloc((RANGE*2+7), sizeof(float));
renyDim = (float *)calloc((RANGE*2+7), sizeof(float));
if((renyDim == NULL)&&((singuSpc == NULL)&&(mandbort == NULL)))
{
    printf("Error of not enough space for renyDim, singuSpc, or/and mandbort.\n");
    exit(0);
}
else
{
    len = fileLength;
    totalN[0] = 0.0;
    totalN[1] = 0.0;
    for(i=0; i<len; i++)
        if((temp=(int)floor(signalSeq[i]+0.5))>0)
            totalN[0] += (double)temp;
    totalN[1] = log10(totalN[0]);
    for(q=-RANGE; q<RANGE+1; q++)
    {
        if((q==2)||(q==1))
            factor = 1;
        else
            factor = 0;
        for(subQ=0; subQ<3*factor+1; subQ++)
        {
            tem[0] = 0.0;
            tem[1] = 0.0;
            temp1 = 0.0;
            temp2 = 0.0;
            for(i=0; i<len; i++)
            {
                temp = (int)floor(signalSeq[i]+0.5);
                if(temp>0)
                {
                    if(q==1)
                    {
                        tem[0] += pow(temp, fQ[0]);
                        tem[1] += pow(temp, fQ[1]);
                    }
                    else
                        tem[0] += pow(temp, (float)q+subQ*0.25);
                }
            }
        }
    }
}

```

```

        temp1+=pow((float)temp/100, (float)q+subQ*0.25);
        temp2+=
        (log10(temp))*pow((float)temp/100, (float)q+subQ*0.25);
    }
}
if(q==1)
{
    tem[0] = log10(tem[0]) - fQ[0]*totalN[1];
    tem[1] = log10(tem[1]) - fQ[1]*totalN[1];
}
else
    tem[0] = log10(tem[0]) - ((float)q+subQ*0.25)*totalN[1];
if(q==1)
{
    logTemp[0] = (1-fQ[0])*log10(len);
    logTemp[1] = (1-fQ[1])*log10(len);
}
else
    logTemp[0] = (1-(float)q-subQ*0.25)*log10(len);
if(q<-2)
{
    renyDim[RANGE+q] = (float)(tem[0]/logTemp[0]);
    dTmp = temp2 - temp1*totalN[1];
    singuSpc[RANGE+q]=
        (float)(dTmp/(log10(1/(float)len)*temp1));
    dTmp = q*dTmp - temp1*(log10(temp1) + 2*q - q*totalN[1]);
    mandbort[RANGE+q]=
        (float)(dTmp/(log10(1/(float)len)*temp1));
}
else if((q>=-2)&&(q<0))
{
    renyDim[RANGE+q+(2+q)*3+subQ] =
        (float)(tem[0]/logTemp[0]);
    dTmp = temp2 - temp1*totalN[1];
    singuSpc[RANGE+q+(2+q)*3+subQ] =
        (float)(dTmp/(log10(1/(float)len)*temp1));
    dTmp = (q+subQ*0.25)*dTmp - temp1*(log10(temp1)
+ 2*((float)q+subQ*0.25) - ((float)q+subQ*0.25)*totalN[1]);
    mandbort[RANGE+q+(2+q)*3+subQ] =
        (float)(dTmp/(log10(1/(float)len)*temp1));
}
else
{
    if(q==1)
        renyDim[RANGE+q+6] =
        (float)(tem[0]/logTemp[0] + tem[1]/logTemp[1])/2;

```

```
renyDim[RANGE+q+6] = (float)(tem[0]/logTemp[0]);
dTmp = temp2 - temp1*totalN[1];
singuSpc[RANGE+q+6] =
    (float)(dTmp/(log10(1/(float)len)*temp1));
dTmp = q*dTmp - temp1*(log10(temp1) + 2*q - q*totalN[1]);
mandbort[RANGE+q+6] =
    (float)(dTmp/(log10(1/(float)len)*temp1));
    }
    }
}
strcpy(outfile, imfile);
strcat(outfile, ".reni");

if ((out = fopen(outfile, "w") ) == NULL)
{
    printf("write error");
    exit(0);
}
for (k=0; k<56; k++)
{
    fprintf(out, "%.4f%.4f%.4f\n", renyDim[k], singuSpc[k], mandbort[k]);
}
fprintf(out, "%.4f%.4f%.4f", renyDim[k], singuSpc[k], mandbort[k]);
fclose(out);
}
```