TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Nachrichtentechnik

Selected Communications Theoretic Aspects in Genetics

Pavol Hanus

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor–Ingenieurs

genehmigten Dissertation.

Vorsitzender:		Univ.–Prof. Dr.–Ing. Jörg Eberspächer
Prüfer:	1.	Univ.–Prof. Dr.–Ing. Dr.–Ing. E.h. Joachim Hagenauer
	2.	Univ.–Prof. Dr.–Ing. Martin Bossert, Universität Ulm

Die Dissertation wurde am 21.04.2010 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 13.07.2010 angenommen.

Preface

This thesis was written during my time as a research assistant at the Institute for Communications Engineering (LNT) at the Technische Universität München (TUM). Many people, whom I am very grateful to, have contributed in different ways to the success of this work.

In the first place, I would like to thank my supervisor Prof. Dr.-Ing. Dr.-Ing. E.h. Joachim Hagenauer for his guidance and support. He gave me the opportunity to conduct research in the novel interdisciplinary field of information and communication theory in molecular biology. The work at his institute was a very fruitful and enriching experience. The interdisciplinary and open nature of my research gave me the opportunity to be creative and learn a lot. Looking at molecular biology from the perspective of engineering by comparing the mechanisms employed by nature to the solutions an engineer would design has been very challenging and rewarding at the same time. In addition, I would like to express my gratitude to Prof. Dr.-Ing. Martin Bossert for his interest in these topics and for co-supervising my work.

I am also very grateful to all my colleagues at the institute for an enjoyable and inspiring atmosphere. It is you that make the institute such an agreeable environment. My special thanks goes to those colleagues and students whom I have been doing research with in our interdisciplinary ComInGen group, especially Janis.

At the faculty I have been appointed with the management of the international graduate program Master of Science in Communications Engineering. I would like to extend my thanks to those whom I have been working with in this context, Ms. Rossmann in particular.

Finally, I would like to express my gratitude to my dear and loved ones for their unique support and encouragement they provided me throughout the years!

Munich, April 2010 Pavol Hanus

- To my parents -

Contents

Zusammenfassung

Es werden neue Anwendungen der Kommunikations- und Informationstheorie auf Probleme in der molekularen Biologie behandelt. Im ersten Teil wird Quellencodierung genutzt um die Speicheranforderungen von genomweiten Sequenzalignment Datensätzen zu verringern. Ein hocheffizienter Kompressionsalgorithmus basierend auf statistischen Modellen der Evolution und Techniken aus der binären Bildcodierung wird vorgeschlagen. Im zweiten Teil werden Parallelen zwischen der Marker Synchronisation über verrauschte Kanäle und der Protein-DNA Bindungsstellensuche studiert. Statistische Bindungsstellen Modelle und Inferenztechniken werden aus informationstheoretischer Sicht analysiert und erweitert. Synchronisationseigenschaften von ausgewählten molekularen Markern werden evaluiert und Evidenz für Selektionsdruck zugunsten effizienter Marker gefunden.

Abstract

This thesis covers novel applications of concepts from communications engineering to problems in molecular biology. In the first part the focus is placed on applying source coding techniques to reduce the storage requirement of multiple genome alignment datasets used in comparative genomics. A highly efficient lossless compression algorithm using well established models of genome evolution and binary image compression techniques is introduced. The second part studies parallels between sequence specific protein binding on the molecular level and marker synchronization. The engineering concept of threshold based marker synchronization over noisy channels is revised and extended. Binding site models and in silico inference techniques are reviewed using information theory. Synchronization properties of selected molecular markers are analysed and evidence for natural selection pressure towards good markers is found.

1

Introduction

In 1948, C. E. Shannon has established the theoretical fundamentals of modern digital communication systems [Sha48]. He quantitatively defined information in an abstract way, independent of semantics, relying solely on the statistical characteristics of the information source. Shannon has proved that messages generated by an information source can be coded at an average rate as low as the uncertainty of the source and still be losslessly reconstructed (source coding theorem). He has also proved that it is possible to code the messages in a way that one can transmit them error-free over a noisy channel at an average rate as high as the channel capacity (channel coding theorem). Ever since, communications engineers have been devising algorithms to achieve the limits of these two theorems. An extensive information and communication theoretic framework has been developed to help analyse and design efficient ways to store, transmit and process information.

In 1953, J. D. Watson and F. H. C. Crick [WC53] have deciphered the double helix structure of the deoxyribonucleic acid (DNA). It has become apparent that the genetic information stored in the DNA that is passed from generation to generation is stored as a digital message from a quaternary alphabet. Thus, information and communication theory can be used to study and analyse the storage, processing and transmission of genetic information on the molecular level. The best understood type of genetic information stored in the DNA are protein coding genes. The original involvement of information theorists with molecular genetics goes back to the discovery of the genetic code, the mapping rule from the 4 letter alphabet of the DNA to the 20 letter alphabet of the proteins. Prior to the experimental decipherment of the actual genetic code in the 1960s, several different mapping schemes have been proposed by information theorists [Hay98], some having high information density, others with error correcting capabilities. Thereafter, the interaction between the two communities has ceased until recently. Some of the reasons

are that for a long time it was believed that protein coding genes are more or less the only functional regions in the genome and the fact that only a very limited amount of sequence data was available. Recent advances in high-throughput sequencing technology have made it possible to sequence whole genomes of complex species. The completion of the first draft of the human genome in 2001 [LLB+01] has revealed that only several percent of it is protein coding and that there are less protein coding genes than expected. The sequencing of other vertebrate genomes could be completed soon after and comparative genomics studies have uncovered that there exist many highly conserved putatively functional regions under strong evolutionary pressure not coding for proteins [BPM+04]. Their function is still largely unknown, but it has been suggested that they might play a role in gene regulation [Mat07], which governs when and in what amounts particular genes get expressed into proteins.

The recent findings have raised many new open questions about the genetic information stored in the genomes, its functionality, storage and processing. Together with the exponential growth of sequence data available in public databases, this has reignited the interest of information theorists and communications engineers for molecular biology. To name a few contributions: G. Battail has postulated the hypothesis that error correcting codes are used on the DNA sequence level [Bat97]. J. Hagenauer et al. applied information theory to gene mapping of complex diseases [HDG+04, SGD+07] and to classification of genetic sequence data [DHHM05] as well as to comparative genomics [DHL+08]. O. Milenkovic et al. introduced methods from channel coding to gene regulation [MV04]. W. Szpankowski et al. used source coding techniques to identify protein coding regions [SRS05]. The initial research results confirm that modelling and analysing the way nature deals with genetic information from a communications engineering perspective can lead to its better understanding. The new insights generated by this interdisciplinary research might even have the potential to help advance future communications systems. Furthermore, methods from communications engineering can be used to reduce the storage requirement of the exponentially growing sequence datasets.

This thesis covers novel applications of concepts from communications engineering to problems in molecular biology. It has two parts. In Part I the focus is placed on applying source coding techniques to reduce the storage requirement of multiple genome alignment datasets used in comparative genomics. Such alignments represent one of the largest sequence datasets used in molecular biology. A highly efficient lossless compression algorithm for multiple genome alignments is introduced. It uses well established models of genome evolution and techniques from binary image compression. Part II of this thesis studies the parallels between sequence specific binding on the molecular level and threshold based marker synchronization. Nature uses specific sequence patterns to mark the target sites for different regulatory proteins and to distinguish information carrying regions e.g. genes. The engineering concept of threshold based marker synchronization over noisy channels is revised and extended. Binding site models and in silico inference techniques are studied and reviewed using an information theoretic framework. Synchronization properties of molecular markers are analysed and evidence for selection pressure towards good markers is found. The structure of this thesis is as follows.

Fundamentals

Chapter 2 introduces the fundamentals from statistics and information theory required in later chapters in the notation used throughout this work.

Chapter 3 provides the necessary background on molecular biology and genetics for non-biologists. Topics from structural biology, molecular biology and evolutionary genetics are covered. The different aspects of genetic information storage in the DNA, its processing during gene expression and its transmission from generation to generation are presented from the perspective of a communications engineer. Error correction implications of DNA damage and mismatch repair are treated. Established statistical models of evolution are explained in detail and used to model the evolutionary mutation channel in later chapters.

Part I

Chapter 4 deals with source coding in engineering. Entropy coding is explained with focus on arithmetic coding. The concept of universal statistical prediction based coding is presented next. The Context Tree Weighting algorithm serves as a representative. It is very efficient at adaptively compressing Markov type dependencies. Universal dictionary based coding is described as well. Concepts and techniques required by the multiple alignment compressor proposed in Chapter 5 of this thesis are explained in more detail.

Chapter 5 treats compression of DNA sequence datasets. First, a brief overview of DNA sequence compressors is provided. Subsequently, a novel compression scheme for multiple genome alignment datasets is introduced. The construction and statistical regularities of such alignment datasets are explained. The proposed algorithm is a highly efficient lossless prediction based compressor combining predictions from statistical models of evolution and binary image compression. The algorithm is shown to be nearly optimal under the used model. With respect to the usage scenario, assuming a central provider of the dataset, the algorithm is designed to achieve high compression at the cost of increased encoder complexity. In addition, it is accounted for the possibility to decompress individual blocks. Therefore, the algorithm could also be used to reduce the storage requirement of respective database servers.

Part II

Chapter 6 deals with marker synchronization in engineering. The focus is placed on threshold based marker synchronization over noisy channels. At the transmitter specific sequence patterns are inserted into the datastream to mark certain positions. A sliding window detector evaluates a likelihood function at the receiver in order to determine the original marker insertion points. Optimal likelihood functions are derived for various channels including evolutionary channel models. The marker choice has a strong influence on the synchronizer's performance. Markers of the same length perform differently. The exact computation of the synchronization success probability of a particular marker is

difficult due to dependencies in the likelihood function values of neighbouring positions. Herein a recursive formula for the exact computation of the synchronization probability is derived for threshold based marker synchronization over discrete memoryless channels and compared to the commonly used approximation assuming positionwise independence.

Chapter 7 analyses the use of markers on the molecular level with focus on sequence specific DNA protein binding. The amount of experimentally verified binding sites is currently very limited. Existing binding site models and in silico inference methods of novel putative sites are studied using an information theoretic framework and extensions are proposed. Subsequently, the synchronization properties of molecular markers are investigated. Actual binding sites are modelled as received noisy realizations of the original marker. The marker evaluation method is used to study selected markers of important DNA binding proteins in bacteria. Evidence for natural selection pressure towards the use of markers with good synchronization properties is found.

Chapter 8 concludes the thesis providing an overview of the main contributions. It also outlines possible directions of future research.

Partial results have been published in journal and conference papers listed in Appendix A. The innovative multiple genome alignment compression algorithm has received the 2009 Capocelli Prize, which is the Best Paper Award of the Data Compression Conference (DCC) [HDCH09]. The work on the synchronization properties of molecular markers has been awarded the 2006 Best Student Presentation Award of the International Society for Computational Biology (ISCB) Student Council [HW06].

2

Fundamentals in Engineering

The purpose of this chapter is to introduce several fundamental concepts from communications engineering. Particularly important to this field is statistics detailed in Section 2.1 and information theory described in Section 2.2. With the establishment of information theory in 1948 C. E. Shannon [Sha48] has revolutionized communications engineering and laid the ground for modern digital communications systems. The following overview is restricted to the concepts required in later chapters. Its primary purpose is to introduce the notation that is used consistently throughout this work.

2.1 Theory of Statistics

The theory of statistics provides a framework for the collection, analysis and interpretation of data. It allows to model, predict and draw inferences about the regularities characterizing the data. In the following, selected concepts from probability theory, estimation theory and statistical hypothesis testing will be presented. The reader is referred to [Hae01] for further reading.

2.1.1 Probability Theory

Probability theory builds the mathematical foundation of statistics providing abstract models for non-deterministic events and measured quantities.

Random Variables

Let $\mathcal{X}$ denote the countable non-empty sample space (alphabet) of the random variable (RV) X and $x \in \mathcal{X}$ be a realization of X. Let $\Pr(event)$ denote the probability of

a particular *event* of the random experiment. The cumulative density function (CDF) of X is defined as $F_X(x) = \Pr(X \leq x)$. If X is a discrete random variable:

$$F_X(x) = \sum_{x' \in \mathcal{X}: x' \leq x} P_X(x'), \tag{2.1}$$

where $P_X(x) = \Pr(X = x)$ is referred to as the probability mass function (PMF) and satisfies $\sum_{x \in \mathcal{X}} P_X(x) = 1$ whereas $P_X(x) \geq 0, \forall x \in \mathcal{X}$. The cardinality (size) of the alphabet is denoted with $|\mathcal{X}|$.

For a continuous random variable X the CDF $F_X(x) = \Pr(X \leq x)$ is assumed to be continuous in x and differentiable. The probability density function (PDF) is defined as $p_X(x) = \partial F_X(x)/\partial x$ and satisfies $\int_{\mathcal{X}} p_X(x)\mathrm{d}x = 1$. Thus,

$$F_X(x) = \int_{-\infty}^{x} p_X(x')\mathrm{d}x'. \tag{2.2}$$

Expectation

The expectation $\mathrm{E}\{\cdot\}$ of an arbitrary function $f(x)$ over the random variable X is defined as:

$$\mathrm{E}\{f(x)\} = \sum_{x \in \mathcal{X}} f(x) P_X(x) \qquad \text{for discrete } X, \tag{2.3}$$

$$\mathrm{E}\{f(x)\} = \int_{\mathcal{X}} f(x) p_X(x)\mathrm{d}x \qquad \text{for continuous } X. \tag{2.4}$$

The expectation $\mu_X = \mathrm{E}\{X\}$ is referred to as the mean. The i-th centralized moment is defined as $\mathrm{E}\{|X - \mu_X|^i\}$, whereas the special case $\sigma^2 = \mathrm{E}\{|X - \mu_X|^2\}$ is the so called variance and σ the standard deviation.

Joint Distribution

Let Y be another random variable. The joint PMF (for discrete X and Y) and PDF (for continuous X and Y) are respectively defined as:

$$P_{XY}(x, y) = \Pr(\{X = x\} \cap \{Y = y\}), \tag{2.5}$$

$$p_{XY}(x, y) = \frac{\partial^2}{\partial x \partial y} F_{XY}(x, y) = \frac{\partial^2}{\partial x \partial y} \Pr(\{X \leq x\} \cap \{Y \leq y\}). \tag{2.6}$$

Thus, $P_X(x) = \sum_{y \in \mathcal{Y}} P_{XY}(x, y)$ in the discrete and $p_X(x) = \int_{-\infty}^{\infty} p_{XY}(xy)\mathrm{d}y$ in the continuous case. In the context of joint distribution functions, the original distributions of X and Y are referred to as marginals.

Conditional Distribution

The conditional probability of an *event* given a *condition* is defined as $\Pr(event|condition)$. The conditional distribution function of two discrete random vari-

ables X and Y is defined according to the Bayes' rule as:

$$P_{Y|X}(y|x) = \frac{P_{X,Y}(x,y)}{P_X(x)} = \frac{P_{X|Y}(x|y)P_Y(y)}{P_X(x)}. \tag{2.7}$$

From this follows the chain rule:

$$P_{XY}(x,y) = P_{X|Y}(x|y)P_Y(y) = P_{Y|X}(y|x)P_X(x). \tag{2.8}$$

The random variables X and Y are independent if $P_{Y|X}(y|x) = P_Y(y)$, or by applying the chain rule if $P_{XY}(x,y) = P_X(x)P_Y(y)$. Generalization to more than two random variables is straightforward.

Example Distribution Functions

Let $\boldsymbol{X} = (X_1, X_2 \dots X_N)$ be a vector of N independent discrete random variables $X_i \in \mathcal{X}, \forall i$. They are referred to as independent and identically distributed (IID) iff $P_{X_i}(x) = P_X(x), \forall x \in \mathcal{X}, i = 1 \dots N$. A discrete distribution is uniform iff $P_X(x) = \frac{1}{|\mathcal{X}|}, \forall x \in \mathcal{X}$. Iff $P_{X_i}(x) = \frac{1}{|\mathcal{X}|}, \forall x \in \mathcal{X}, i = 1 \dots N$ the random variables are referred to as independent and uniformly distributed (IUD).

A binary random variable is called Bernoulli distributed with $P_X(x = 1) = \theta$ and $P_X(x = 0) = 1 - \theta$. For N independent Bernoulli distributed RVs, the random variable $K = \sum_{i=1}^{N} X_i$ measuring the number of independent success trials follows a binomial distribution:

$$P_K(k) = \binom{N}{k} \theta^k (1-\theta)^{N-k}, \qquad 0 \leq k \leq N. \tag{2.9}$$

Important continuous distributions used in this work are the Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ with variance σ and mean μ

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{2.10}$$

the Exponential distribution $\mathcal{E}(\lambda)$ with the rate parameter λ

$$p_X(x|\lambda) = \lambda e^{-\lambda x} \qquad x \geq 0, \tag{2.11}$$

and the Gamma distribution $\Gamma(n, \lambda)$

$$p_X(x|n, \lambda) = \frac{\lambda^n}{\Gamma(n)} x^{n-1} e^{-\lambda x}, \tag{2.12}$$

where x is the sum of n independent λ-exponentially (2.11) distributed random variables and $\Gamma(n) = (n-1)!, n \in \mathbb{N}$ is the so called Gamma function.

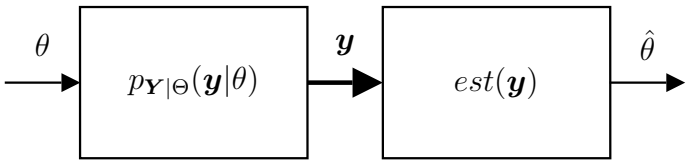


Figure 2.1: Estimation model.

2.1.2 Estimation Theory

Estimation theory deals with inferring parameters underlying a probabilistic model from measured observable data.

The estimation scenario used in this work is depicted in Figure 2.1. The vector $\boldsymbol{y} = (y_1, y_2 \dots y_N)$ corresponds to realizations of the random variables $\boldsymbol{Y} = (Y_1, Y_2 \dots Y_N)$ with $y_i \in \mathcal{Y}, \forall i$. The observation is conditioned on θ according to the PDF $p_{\boldsymbol{Y}|\Theta}(\boldsymbol{y}|\theta)$, where θ in itself is a realization of a discrete random variable Θ distributed according to $P_\Theta(\theta)$. An estimator tries to compute an estimate value $\hat{\theta}$ for θ from the observation $\boldsymbol{y}$ using a suitable estimation function $est(\boldsymbol{y})$.

Estimation Functions

An optimal Bayesian estimator function is the so called maximum a-posteriori probability estimator (MAP)

$$\hat{\theta} = \arg\max_{\theta\in\Theta}(P_{\Theta|\boldsymbol{Y}}(\theta|\boldsymbol{y})) = \arg\max_{\theta\in\Theta}\left(\frac{p_{\boldsymbol{Y}|\Theta}(\boldsymbol{y}|\theta)P_\Theta(\theta)}{p_{\boldsymbol{Y}}(\boldsymbol{y})}\right) = \arg\max_{\theta\in\Theta}(p_{\boldsymbol{Y}|\Theta}(\boldsymbol{y}|\theta)P_\Theta(\theta)), \tag{2.13}$$

where $P_{\Theta|\boldsymbol{Y}}(\theta|\boldsymbol{y})$ is the posterior PMF. If the prior distribution $P_\Theta(\theta)$ is uniformly distributed, the estimation function simplifies to the maximum likelihood estimator (ML)

$$\hat{\theta} = \arg\max_{\theta\in\Theta}(p_{\boldsymbol{Y}|\Theta}(\boldsymbol{y}|\theta)), \tag{2.14}$$

which can also be used for suboptimal estimation if the prior distribution is unknown.

2.1.3 Hypothesis Testing

Hypothesis testing is part of statistical inference dealing with statistical decision making based on observed data. In this work the log likelihood ratio (LLR) will be used.

Log Likelihood Ratio

The log likelihood ratio originates in statistical binary hypothesis testing. Let $\boldsymbol{X} = (X_1, X_2 \dots X_N)$ be a vector of N random variables with parametrized probability density function $p_{\boldsymbol{X}|\Theta}(\boldsymbol{x}|\theta)$ and let $\boldsymbol{x} = (x_1, x_2 \dots x_N)$ be an observed realization of $\boldsymbol{X}$. Given the parameter space Θ and two complementary subspaces Θ_0 and Θ_1, where $\Theta_0 \cup \Theta_1 = \Theta$ and $\Theta_0 \cap \Theta_1 = \emptyset$, the likelihood ratio $\Lambda(\boldsymbol{x})$ is defined as the quotient of the likelihood

of observing $\boldsymbol{x}$ under the two competing hypothesis $\mathcal{H}_0 : \theta \in \Theta_0$ being the so called null hypothesis and $\mathcal{H}_1 : \theta \in \Theta_1$ being the alternative hypothesis:

$$\Lambda(\boldsymbol{x}) = \frac{p_{\boldsymbol{X}|\mathcal{H}_0}(\boldsymbol{x}|\mathcal{H}_0)}{p_{\boldsymbol{X}|\mathcal{H}_1}(\boldsymbol{x}|\mathcal{H}_1)}. \tag{2.15}$$

The log likelihood ratio $L(\boldsymbol{x})$ is defined as

$$L(\boldsymbol{x}) = \log(\Lambda(\boldsymbol{x})) = \log\left(\frac{p_{\boldsymbol{X}|\mathcal{H}_0}(\boldsymbol{x}|\mathcal{H}_0)}{p_{\boldsymbol{X}|\mathcal{H}_1}(\boldsymbol{x}|\mathcal{H}_1)}\right). \tag{2.16}$$

If the random variables in $\boldsymbol{X}$ are independent $p_{\boldsymbol{X}|\Theta}(\boldsymbol{x}|\theta) = \prod_{i=1}^{N} p_{X_i|\Theta}(x_i|\theta)$. For independent X_i the likelihood ratio $\Lambda(\boldsymbol{x}) = \prod_{i=1}^{N} \Lambda(x_i)$ and $L(\boldsymbol{x}) = \sum_{i=1}^{N} L(x_i)$, where $L(x_i)$ is the symbolwise log likelihood ratio.

In hypothesis testing a threshold λ is used $L(\boldsymbol{x}) \leq \lambda$ to determine whether to reject the null hypothesis $\mathcal{H}_0$ in favour of $\mathcal{H}_1$. Type I error occurs if $\mathcal{H}_0$ is falsely rejected and type II error if it is falsely accepted. The threshold λ is usually chosen according to the so called Neyman-Pearson criterion by upper bounding the probability of the type I error $p(L(\boldsymbol{x}) \leq \lambda|\mathcal{H}_0)$. Thus,

$$\lambda = \arg\max_{\mu} \big(p(L(\boldsymbol{x}) \leq \mu|\mathcal{H}_0) \leq \alpha\big), \tag{2.17}$$

where α is the chosen upper bound.

2.2 Information Theory

Information theory was founded by C.E. Shannon in 1948 [Sha48]. It provides a mathematical framework for information quantification in the context of message transmission and storage. Shannon's definition of information is universal. It is independent from the meaning of the message and relies solely on statistical properties of the message. Shannon proved that a message generated by an information source can be losslessly compressed up to the entropy of the source (source coding theorem) and that it is possible to code the information in a way such that it can be transmitted error free over the channel (channel coding theorem). In the following, relevant information theoretic quantities will be introduced in Section 2.2.1. Subsequently, fundamentals of source coding are presented in Section 2.2.2. Section 2.2.3 deals with information transmission. The reader is referred to [CT06] for further reading.

2.2.1 Information Theoretic Quantities

In the following, important information theoretic quantities will be defined. The definitions make use of statistical terms and notation introduced in Section 2.1.1.

Entropy

Shannon has defined self-information of an event x as $H(x) = \log(1/P_X(x))$. The expectation $E\{H(x)\}$ is referred to as entropy and represents a measure of the uncertainty about the outcome of the random variable X. For a discrete random variable X, it can be written:

$$H(X) = E\{H(x)\} = E\left\{ \log \frac{1}{P_X(x)} \right\} = \sum_{x \in \mathcal{X}} P_X(x) \log \frac{1}{P_X(x)}. \tag{2.18}$$

It can be shown that $0 \leq H(X) \leq \log |\mathcal{X}|$, whereas the maximum entropy $H(X) = \log |\mathcal{X}|$ is reached in case of a uniform distribution and the minimum $H(X) = 0$ for a strictly deterministic process, carrying no uncertainty about the outcome. The unit of entropy depends on the chosen base of the logarithmic function in the self-information. For $\log = \log_2 = \mathrm{ld}$ the unit of the entropy is bit.

For two discrete random variables X and Y the joint entropy is defined as the expectation of the joint self-information $H(x, y)$:

$$H(X,Y) = E\{H(x,y)\} = E\left\{ \log \frac{1}{P_{XY}(x,y)} \right\} = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P_{XY}(x,y) \log \frac{1}{P_{XY}(x,y)}, \tag{2.19}$$

Joint entropy is the uncertainty of X combined with Y. It is symmetric $H(X,Y) = H(Y,X)$ and attains values in the range $\max(H(X), H(Y)) \leq H(X,Y) \leq H(X) + H(Y)$, whereas the lower bound is reached when the uncertainty of the random variable with lower entropy is contained in the uncertainty of the variable with higher entropy. The upper bound is achieved for independent random variables.

The conditional entropy describes the uncertainty remaining in X given the random variable Y and is defined as the expectation of the conditional self-information $H(x|y)$:

$$H(X|Y) = E\{H(x|y)\} = E\left\{ \log \frac{1}{P_{X|Y}(x|y)} \right\} = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P_{X,Y}(x,y) \log \frac{1}{P_{X|Y}(x|y)}. \tag{2.20}$$

Conditional entropy is in the range $0 \leq H(X|Y) \leq H(X)$. It converges towards zero if Y determines X and maximizes to $H(X)$ if X and Y are statistically independent. Thus, conditioning can only reduce entropy. Conditional entropy is in general not symmetric $H(Y|X) \neq H(X|Y)$. The following chain rule applies to entropies

$$H(X_1, \ldots, X_n) = \sum_{i=1}^{n} H(X_i | X_1, \ldots, X_{i-1}). \tag{2.21}$$

For two random variables X and Y it reduces to $H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y) = H(Y,X)$.

Relative Entropy

The divergence of two discrete probability distributions with the probability mass functions $P_X(X)$ and $Q_X(X)$ can be measured in terms of relative entropy, also called Kullback-Leibler divergence [Kul59], defined as:

$$D(P_X(X)||Q_X(X)) = E_P\left\{ \log \frac{P_X(x)}{Q_X(x)} \right\} = \sum_{x\in\mathcal{X}} P_X(x) \log \frac{P_X(x)}{Q_X(x)}. \tag{2.22}$$

However, relative entropy is not a metric distance measure as it only satisfies the identity axiom

$$D(P_X(X)||Q_X(X)) = 0 \text{ iff } P_X(X) = Q_X(X), \tag{2.23}$$

but neither the symmetry

$$D(P_X(X)||Q_X(X)) \neq D(Q_X(X)||P_X(X)), \tag{2.24}$$

nor the triangle inequality

$$D(P_X(X)||Q_X(X)) \not\leq D(P_X(X)||R_X(X)) + D(R_X(X)||Q_X(X)). \tag{2.25}$$

The relative entropy can also be interpreted as the increase in the entropy estimate if the random variable is falsely assumed to be distributed according to $Q_X(x)$ instead of the true distribution $P_X(X)$

$$D(P_X(X)||Q_X(X)) = E_P\left\{ \log \frac{1}{Q_X(x)} \right\} - H(P_X(X)). \tag{2.26}$$

In the context of hypothesis testing by setting $P_X(X) = P(X|\mathcal{H}_0)$ and $Q_X(X) = P(X|\mathcal{H}_1)$, the Kullback-Leibler divergence $D(P(X|\mathcal{H}_0)||P(X|\mathcal{H}_1))$ is the so called discrimination information for hypothesis $\mathcal{H}_0$ over $\mathcal{H}_1$, which is the mean information per sample for discriminating in favour of hypothesis $\mathcal{H}_0$ against the hypothesis $\mathcal{H}_1$, when hypothesis $\mathcal{H}_0$ is true. This corresponds to the weight of evidence for $\mathcal{H}_0$ over $\mathcal{H}_1$ expected from each sample.

Mutual Information

Shannon has defined information as the reduction of uncertainty. As shown above, without the knowledge of Y the uncertainty of X is $H(X)$. Once Y is given the uncertainty reduces to $H(X|Y)$ and vice versa. Thus mutual information is defined as

$$I(X;Y) = H(X) - H(X|Y) = H(X) + H(Y) - H(Y,X) = H(Y) - H(Y|X). \tag{2.27}$$

It attains values in the range of $0 \leq I(X;Y) \leq \min(H(X), H(Y))$ and vanishes for random variables that are independent. In terms of probabilities the definition becomes:

$$I(X;Y) = E\{I(x,y)\} = E\left\{ \log \frac{P_{XY}(x,y)}{P_X(x)P_Y(y)} \right\} = \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} P_{XY}(x,y) \log \frac{P_{XY}(x,y)}{P_X(x)P_Y(y)},$$

where $I(x, y)$ is referred to as the mutual self-information. Thus, mutual information equals the relative entropy of the joint probability mass function $P_{XY}(X, Y)$ and the product $P_X(X) \cdot P_Y(Y)$ of marginal probability mass functions of X and Y

$$I(X;Y) = D(P_{XY}(X,Y)||P_X(X) \cdot P_Y(Y)). \tag{2.29}$$

Therefore, it can be considered a measure of statistical dependence between the random variables X and Y.

2.2.2 Source Coding

Source coding refers to the effort of removing redundancy from an information source. The aim of source coding is to represent the messages generated by an information source with as few bits as possible. This process is called compression. A compression algorithm makes use of the knowledge of statistical dependencies present in the information source to achieve compression. In Chapter 4 of this thesis, typical source models in engineering and the corresponding compression algorithms will be investigated. In Chapter 5, sources from molecular biology will be treated. In the following, the theoretical limits of source coding established by Shannon shall be briefly introduced.

Information Source

According to Shannon, "any stochastic process which produces a discrete sequence of symbols chosen from a finite set may be considered a discrete source" [SSW93]. Thus, a source $\boldsymbol{X}$ can be modelled as a sequence of random variables $\boldsymbol{X} = (X_1, X_2, \dots)$ generating symbols $\boldsymbol{x} = (x_1, x_2, \dots)$ with $x_i \in \mathcal{X}, \forall i$. In general, there may be arbitrary dependencies among the random variables X_i constituting the source. Typically, the source is characterized by the joint PMF $P_{X_1,X_2\dots X_N}(\cdot), N \in \mathbb{N}$. If the joint PMF is time invariant $P_{X_1,X_2\dots X_N}(\cdot) = P_{X_{o+1},X_{o+2}\dots X_{o+N}}(\cdot), \forall o \in \mathbb{N}$, the source is referred to as stationary. The source is ergodic if the sample and statistical mean coincide. More precisely, if the observed relative frequency of a particular subsequence converges to its probability for a sufficiently long observation of the source. If the random variables X_i are statistically independent $P_{X_i|X_1\dots X_{i-1}}(\cdot|\cdot) = P_{X_i}, \forall i$, then the source is memoryless. A special case of sources with memory is the Markov source. The n-th order Markov source is characterized by dependence of the current symbol from n previous symbols $P_{X_i|X_1\dots X_{i-1}}(\cdot|\cdot) = P_{X_i|X_{i-n}\dots X_{i-1}}(\cdot|\cdot)$.

Entropy Rate of the Source

Given the joint entropy of the source, the source's entropy rate is defined as the symbolwise entropy

$$\mathcal{H}(\boldsymbol{X}) = \lim_{N\to\infty} \frac{1}{N} H(X_1, X_2 \dots X_N). \tag{2.30}$$

For a memoryless source this simplifies to $\mathcal{H}(\boldsymbol{X}) = \frac{1}{N}\sum_{i=1}^{N} H(X_i)$, for an IID source $\mathcal{H}(\boldsymbol{X}) = H(X)$ and for an IUD source $\mathcal{H}(\boldsymbol{X}) = \log(|\mathcal{X}|)$.

Shannon's Source Coding Theorem

Shannon's fundamental source coding theorem defines a lower bound on the source code rate R in bits/symbol achievable when losslessly compressing an infinitely long symbol sequence $\boldsymbol{x}$ generated by the source $\boldsymbol{X}$:

$$R \geq \mathcal{H}(\boldsymbol{X}). \tag{2.31}$$

The lower bound is only achievable by an optimal compressor for the given source and an infinitely long observation. In practice the messages to be compressed have a finite length and the aim is to device a compressor that compresses different messages generated by the source with an average rate close to the entropy of the source.

2.2.3 Information Transmission

In the context of information transmission depicted in Figure 2.2, the message $\boldsymbol{x} = (x_1, x_2 \dots x_N)$ generated by a discrete information source $\boldsymbol{X}$ described by $P_{\boldsymbol{X}}(\boldsymbol{X})$ is transmitted over a noisy channel. The receiver receives a noisy version of the original message $\boldsymbol{y} = (y_1, y_2 \dots y_N)$. The channel is characterized by the conditional PDF $p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{Y}|\boldsymbol{X})$ if it is continuous and by the conditional PMF $P_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{Y}|\boldsymbol{X})$ if being value discrete. The depicted model is a simplified discrete time model of the physical channel, where the received symbols y_i represent the time sampled output of a matched filter.

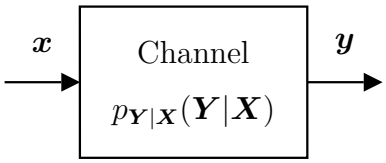


Figure 2.2: Transmission over a noisy channel.

Channel Capacity

The amount of information about $\boldsymbol{X}$ that can be transmitted over a channel is described by the mutual information $I(\boldsymbol{X};\boldsymbol{Y})$, quantifying the reduction of uncertainty about $\boldsymbol{X}$ by observing $\boldsymbol{Y}$. In general, it is more convenient to resort to the information rate being the symbolwise mutual information describing the number of bits transmitted on average per channel use

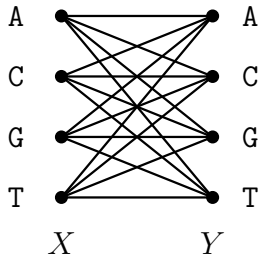
$$\mathcal{I}(\boldsymbol{X};\boldsymbol{Y}) = \lim_{N\to\infty} \frac{1}{N} I(\boldsymbol{X};\boldsymbol{Y}) = \mathcal{H}(\boldsymbol{X}) - \mathcal{H}(\boldsymbol{X}|\boldsymbol{Y}). \tag{2.32}$$

The channel capacity C refers to the maximum achievable information rate for a given channel. With the characteristic conditional channel distribution being given and fixed, the maximization is done over all possible input distributions

$$C = \max_{P_{\boldsymbol{X}}(\boldsymbol{X})} \mathcal{I}(\boldsymbol{X};\boldsymbol{Y}). \tag{2.33}$$

Channel Models

In the following, several channel models relevant to this thesis shall be presented. All of the used channels are assumed to be memoryless $p_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{i=1}^{N} p_{Y_i|X_i}(Y_i|X_i)$ and invariant $p_{Y_i|X_i}(Y_i|X_i) = p_{Y|X}(Y|X), \forall i$ unless otherwise stated.



$$\boldsymbol{P}_{Y|X} = \begin{pmatrix} P(\mathtt{A}|\mathtt{A}) & P(\mathtt{C}|\mathtt{A}) & P(\mathtt{G}|\mathtt{A}) & P(\mathtt{T}|\mathtt{A}) \\ P(\mathtt{A}|\mathtt{C}) & P(\mathtt{C}|\mathtt{C}) & P(\mathtt{G}|\mathtt{C}) & P(\mathtt{T}|\mathtt{C}) \\ P(\mathtt{A}|\mathtt{G}) & P(\mathtt{C}|\mathtt{G}) & P(\mathtt{G}|\mathtt{G}) & P(\mathtt{T}|\mathtt{G}) \\ P(\mathtt{A}|\mathtt{T}) & P(\mathtt{C}|\mathtt{T}) & P(\mathtt{G}|\mathtt{T}) & P(\mathtt{T}|\mathtt{T}) \end{pmatrix}$$

Figure 2.3: Discrete memoryless channel. In this example $\mathcal{X} = \mathcal{Y} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$.

A discrete memoryless channel (DMC) is completely specified by the sets $\mathcal{X}$, $\mathcal{Y}$ and the PMF $P_{Y|X}(Y|X)$. In Figure 2.3 the PMF is described in form of a transition probability matrix $\boldsymbol{P}_{Y|X}$. The channel is called strongly symmetric if the values in the rows as well as the columns of the probability transition matrix are permutations of each other. The channel capacity in this case is achieved for uniformly distributed channel input and equals

$$C = \log|\mathcal{X}| - \sum_{y\in\mathcal{Y}} P_{Y|X}(y|x) \log \frac{1}{P_{Y|X}(y|x)}. \tag{2.34}$$

A special case of the DMC is the binary symmetric channel (BSC). It is completely characterized by the symbol error probability p. The channel capacity is in this case $C_{BSC} = 1 - H_b(p)$, where $H_b(p)$ is the so called binary entropy function.

A typical continuous memoryless channel model is the average white Gaussian noise channel (AWGN). This additive channel perturbs the transmitted symbols with white noise $Y = X + N$, where N has the Gaussian distribution $\mathcal{N}(0, \sigma_n^2)$. Let the average transmission power be $\sigma_x^2 = E_s$ and the noise variance be the two-sided power spectral density of the noise normalized by the average transmission power $\sigma_n^2 = \frac{N_0}{2}$. If X is continuous and has a Gaussian distribution, the channel capacity becomes $C_{AWGN} = \frac{1}{2}\log(1 + \frac{2E_s}{N_0})$. For discrete X the capacity has to be computed numerically under consideration of the modulation scheme.

Channel Coding

According to Shannon's channel coding theorem, the digital information messages to be transmitted over a noisy channel can be encoded in a way that they can be reconstructed without error at the receiver if encoded at a rate R smaller than the channel capacity $R < C$. The channel encoder maps the messages of length K to longer codewords $N > K$, which leads to a rate $R = K/N$ and corresponds to systematically adding redundancy to the information messages. Channel coding aims at finding suitable codes almost achieving the channel capacity. The reader is referred to [Bos99] for further reading.

3

Fundamentals in Genetics

Genetics traces its origins back to the inheritance studies conducted by Georg Mendel [Men86] around 1860. Modern genetics has become molecular and seeks to study not only the molecular principles of inheritance but also the structure and function of the genetic material at the molecular level. It comprises elements from several different disciplines, particularly biology, biochemistry and biophysics. From information theoretic perspective, insights gained on the molecular level have revealed that all genetic information passed on to progeny, representing the blueprint of an organism, is stored as a long linear sequence of letters from a four letter alphabet. This observation raises questions about parallels to digital data storage, processing, synchronization and transmission techniques used in communications engineering. The theoretical framework and the methods developed to design and analyse digital communication systems can be used to study how nature deals with genetic information on the molecular level. In this chapter, the fundamentals of molecular information storage, processing and transmission are presented and complemented with a communications engineer's view and interpretation. The main mechanisms are very similar across all living organisms. However, there exist notable differences between the two main domains of life - the prokaryotes and eukaryotes - that have separated very early in evolutionary history. In the following, these two domains of life are briefly introduced in Section 3.1. Subsequently, knowledge about the storage of genetic information gained by molecular structural biology is presented in Section 3.2. Afterwards, processing of genetic information researched primarily by molecular biology is treated in Section 3.4. Finally, the insights of evolutionary genetics about the transmission of genetic information are covered in Section 3.6. The presented overview is not aiming at being exhaustive and it is abstracted to a certain degree, since its main purpose is to help non-biologists to better understand the subsequent chapters. For further details, the reader is referred to [AJW+08, Lew04, Kni06].

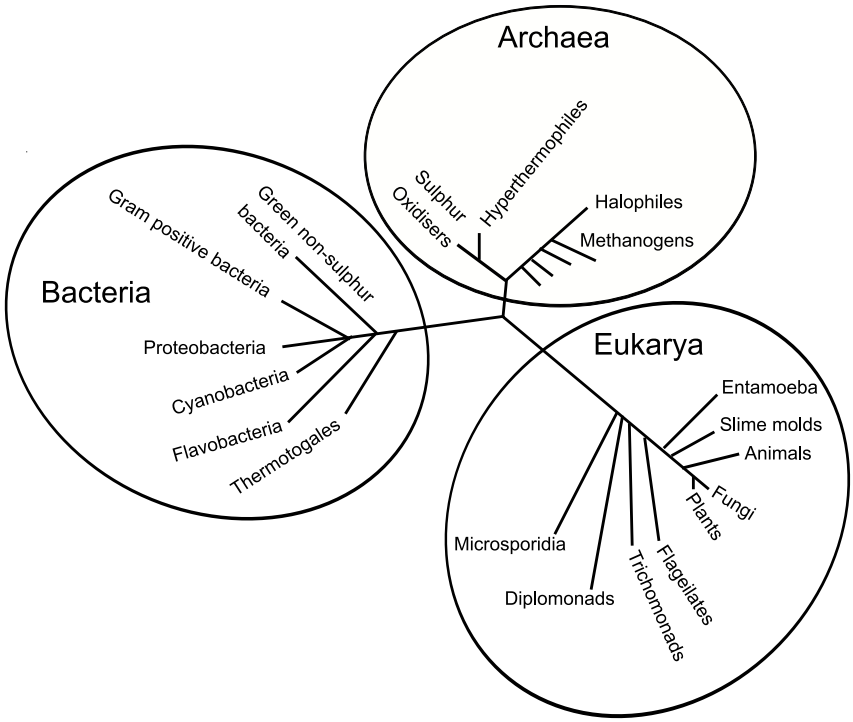


Figure 3.1: An evolutionary tree depicting the three domains of life. Modified from [Wik08].

3.1 Domains of Life

In biological taxonomy the highest taxonomic rank of organisms is a domain. The currently widely accepted tree of life as depicted in Figure 3.1 consists of the following three domains: archaea, bacteria and eukarya [WKW90]. The domains reflect the fundamental evolutionary differences in the genomes of the three taxa. In the context of this work the reduced tree of life, grouping archaea and bacteria together as prokaryotes based on their outward appearance, is used. On the molecular level, archaea organisms resemble bacteria in terms of metabolism and energy conversion, however they use genetic information processing machinery similar to eukaryotes [AJW+08].

3.1.1 Prokaryotes

Prokaryotes are the older and less complex, yet a very successful unicellular life form. Prokaryote cells typically live as independent individuals or in loosely organized colonies. They are no more than a few micrometers large. Usually, they comprise a tough protective cell wall that encloses a cytoplasmic compartment containing loosely floating DNA, RNA and proteins. The prokaryotic genomes are shorter and simpler than those of eukaryotes. However, opposed to their relatively simple structure, prokaryotic cells have adopted to an enormous variety of ecological niches. In general, prokaryotes reproduce asexually. Nonetheless, horizontal gene transfer takes place between prokaryote cells enhancing the spread of advantageous traits in the population. The generation cycles of prokaryotes are in general far shorter than those of eukaryotes and their population sizes are much larger.

3.1.2 Eukaryotes

Opposed to prokaryotes, eukaryotes represent a complex form of life. Eukaryotic cells are more elaborate and about 10^3 times larger in volume than those of prokaryotes. They keep their DNA in a distinct, membrane-enclosed compartment referred to as the nucleus. Eukaryotic genomes are densely packed and far larger in size. Often eukaryotic cells form multicellular organisms. Eukaryotic organisms usually reproduce sexually. Thus, eukaryotic cells possess a diploid copy of the genome, inheriting one copy from each parent. It is hypothesized that eukaryotes have evolved from predator cells in the primordial world engulfing and digesting other cells [AJW+08]. This hypothesis helps to explain why eukaryotes protect their DNA by storing it in a separate nucleus and also the fact that they contain small organelles that resemble bacteria within their cytoplasm (i.e. mitochondria in animals and chloroplasts in plants). Today, these organelles are responsible for the energy production in a eukaryotic cell. They possess their own genomes and processing machinery. These organelles are believed to have been engulfed by an ancestral eukaryotic cell and to have evolved in symbiosis with it by escaping digestion. The high level of complexity of eukaryotic organisms leads to smaller populations and longer generation cycles.

3.1.3 Information Theoretic Implications

The fundamentally different survival strategies of the two domains of life have strong implications on the engineering aspects of the corresponding species. The minimalistic and simplistic approach of prokaryotes with large populations and short generation cycles imposes energetic constraints on individual organisms. This implies reduced complexity of the molecular processes with focus on few simple and robust mechanisms. The amount of genetic information passed from generation to generation is limited and restrained to the minimum, resulting in small genomes. Adaptation to changing environments is achieved by natural selection in large populations. Thus, the faithfulness of genetic information passed by an individual to its progeny is not as crucial for the survival of the species. In eukaryotes, on the other hand, the evolution seems to aim at achieving robustness via increased complexity and redundancy on the level of individual organisms. The energetic constraint upon the amount of genetic information passed on to the offspring is far weaker and the genome size is generally much larger. However, due to long generation cycles and small population sizes, the genetic information needs to be passed on to the next generation at a high level of fidelity. Therefore, if there exist any error correcting means in the sense of channel coding on the sequence level, it can be hypothesized that they are more likely to be found in eukaryotes.

3.2 Structural Biology - Genetic Information Storage

Structural biology is a branch of molecular biology, biochemistry, and biophysics concerned with the molecular structure of biological macromolecules, especially DNA, RNA and proteins. Besides studying the composition and structure, it also focuses on how struc-

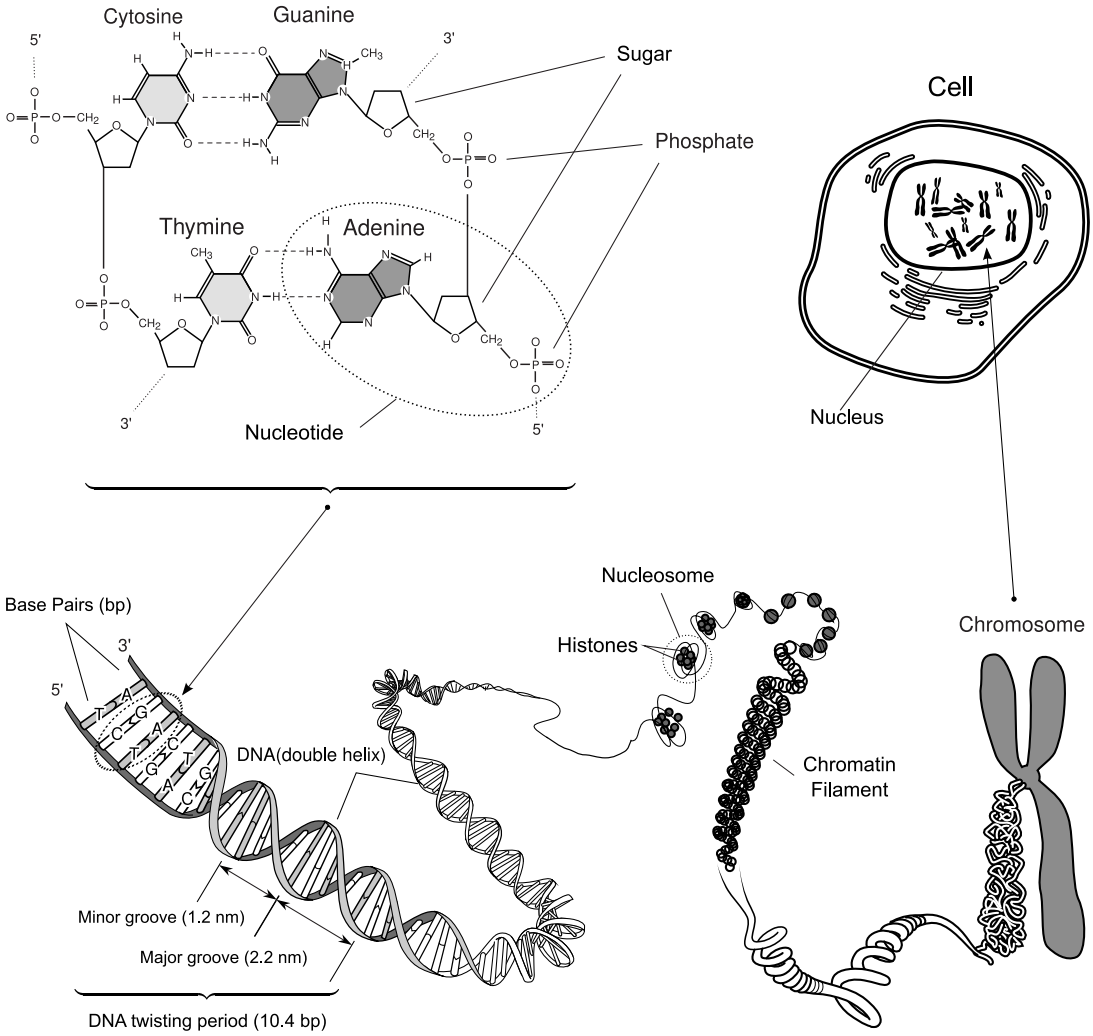


Figure 3.2: Basic chemical components of double stranded DNA and its double helix B-DNA conformation (left), DNA packaging in eukaryotes (right). Modified from [Nat08].

tures are acquired and how alterations in the structure influence the specific function of these macromolecules. This subject is of great interest to biologists since macromolecules carry out most of the functions of the living cells, and because it is only by coiling into specific three-dimensional shapes that they are able to perform these functions. The actual three-dimensional conformation of a molecule is called tertiary structure.

3.2.1 DNA

The primary carrier of genetic information in all living cells is the deoxyribonucleic acid (DNA) molecule - a long unbranched double stranded polymer chain formed of monomers of four different types. These monomers are the so called nucleotides each consisting of a deoxyribose sugar, a phosphate group and a base being either `A`-adenine, `C`-cytosine, `G`-guanine or `T`-thymine. In a single DNA strand, the sugar of one nucleotide is linked to the phosphate of the next one by strong covalent bonds resulting in an alternating directed sugar phosphate backbone. Since the bases are not involved in

the binding along a strand, the nucleotide sequence of a single strand can be arbitrary. The two strands of double stranded DNA, however, are held together by hydrogen bonds between opposing bases from each strand. Since the bonding is in general only possible between A=T and C≡G, the two DNA strands are complementary. The base pairs (bp) A=T and C≡G are held together by 2 and 3 hydrogen bonds respectively. Thus, based on bonding strength, the bases can can be subdivided into weak W={A,T} and strong S={G,C}. The binding between the two DNA strands is therefore stronger for GC rich regions. Another grouping is possible according to the size of the bases. From chemical point of view, the bases represent nitrogenous heterocyclic aromatic organic compounds. While the larger purines denoted by R={A,G} possess two aromatic rings, the pyrimidines denoted by Y={C,T} only have a single ring [TBS09]. Thus, a valid base pair always contains 3 rings, see upper left portion of Figure 3.2.

W={A,T} - weak	R={A,G} - purine
S={C,G} - strong	Y={C,T} - pyrimidine

The sequence of nucleotides in the DNA is referred to as the primary structure. The complementary structure of the two DNA strands is the so called secondary structure. The tertiary structure is the double helix and was discovered in 1953 [WC53]. In the double helix, the direction of the nucleotides in one strand is opposite to their direction in the other strand. This arrangement is called anti-parallel. The asymmetric ends of DNA strands are referred to as the 5' (five prime) and 3' (three prime) ends, with the 5' end being that with a terminal phosphate and the 3' end being the one with a terminal sugar. In its natural state, the DNA double helix assumes a conformation referred to as B-DNA, which is a right handed spiral with a periodicity of about 10.4 bp or 3.4 nm. The spaces between the twisted strands along the rotational axis of the double helix are referred to as the grooves. Since the strands are not directly opposite to each other, the grooves are unequally sized. The minor groove is 1.2 nm and the major groove is 2.2 nm large [WDT$^+$80], see lower left portion of Figure 3.2. The accessibility and distinguishability of bases is better from the major groove. Therefore, sequence specific DNA binding takes place primarily in the major groove [PS84]. DNA specific binding will be studied in more detail in Section 7.1.2.

DNA packaging

While in prokaryotes the DNA usually freely floats in the cytoplasm, forming a closed ring, in eukaryotes it is typically densely packed via chromatin. The packaging is commonly referred to as the chromatin structure. The smallest packaging unit is a nucleosome composed of a histone proteins octamer wrapped around by double-stranded DNA. It makes the bases facing the histone octamer inward less accessible than those facing it outward. The arrangement resembles "beads-on-a-string" and is folded in a helical structure called the chromatin filament, which is arranged in chromosomes in the eukaryotic cell nucleus [AJW$^+$08]. The described eukaryotic DNA packaging is depicted in the right portion of Figure 3.2. It plays an important role in the gene regulation.

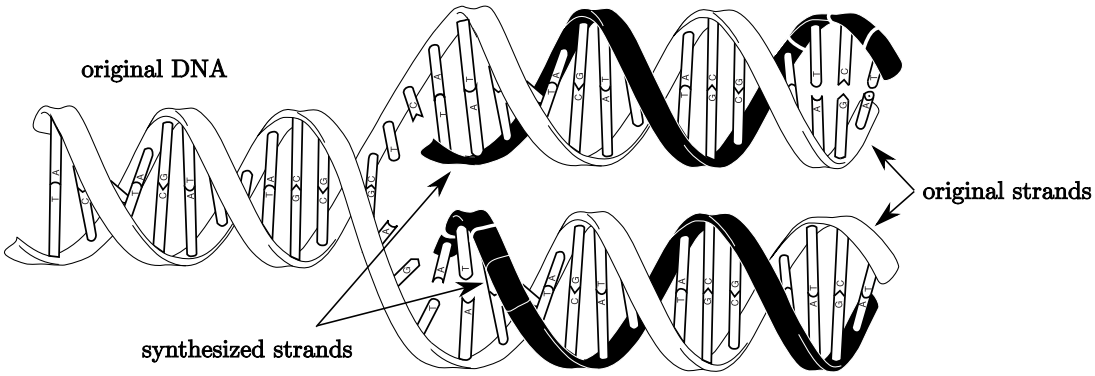


Figure 3.3: DNA replication: Each of the original strands is used as a template and complemented by a newly synthesized strand. Modified from [Nat08].

DNA replication

The complementary base pair binding allows for a simple DNA copying mechanism and easy replication of the double stranded DNA, using the two complementary DNA strands as templates, see Figure 3.3. Each of the two replica contains one original and one synthesized strand. Thus, if a mismatch base pair was present in the original double stranded DNA, the two replica are not going to contain a mismatch any more but they will differ at the position of the original mismatch base pair.

3.2.2 RNA

For further processing, DNA is transcribed into ribonucleic acid (RNA), see Section 3.4.1. RNA is very similar to single stranded DNA. However, instead of `T`-thymine it uses the base `U`-uracil, likewise binding to `A`-adenine. Additionally, the sugar is a ribose instead of a deoxyribose, making the sugar phosphate backbone less stable [TBS09]. The best studied function of RNA is serving as messenger RNA (mRNA) in the process of translation of the protein coding DNA regions into proteins, as described in Section 3.4.3. The mRNA is a complementary copy of the linear sequence of protein coding nucleotides and serves as a template for the protein synthesizing machinery. Apart from mRNA, especially in eukaryotes a lot of non-coding RNA (ncRNA) is transcribed [MM06]. A portion of the ncRNA serves regulatory purposes by binding complementary to mRNA, preventing its translation. However, the function of most eukaryotic ncRNA is subject of ongoing research. Partially, RNAs contain self-complementary sequences that allow parts of the RNA to fold and pair with itself. Thereby, it forms short double helices, packed together into structures akin to proteins, serving complex functions due to their specific tertiary form. Best studied examples of such folded functional RNA are the ribosomal RNA (rRNA) and the transfer RNA (tRNA), involved in mRNA to protein translation, see Section 3.4.3. Figure 3.4 depicts the folded tertiary (left) and the secondary (right) structure of a tRNA, where the secondary structure refers to the scaffold of the tertiary structure, depicting the sequence and the partial complementary binding that takes place.

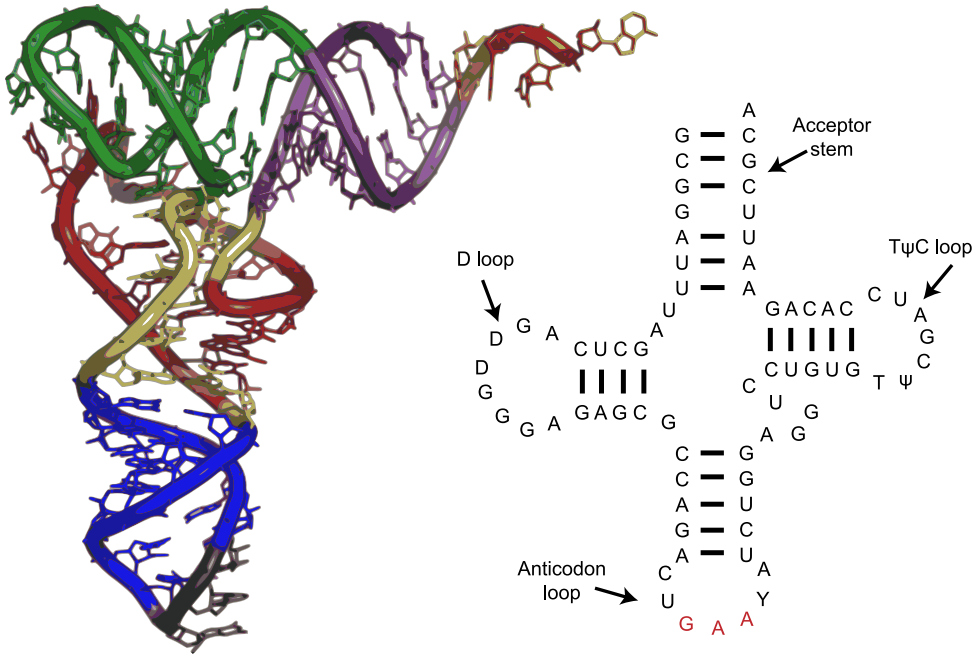


Figure 3.4: The tertiary (left) and secondary (right) structure of a tRNA molecule. Modified from [Wik08] and [Nat08].

3.2.3 Proteins

Most functions in a cell are performed by macromolecules known as the proteins, whose primary structure is a long polymer chain formed of 20 different amino acid monomers held together by peptide bonds. In general, the amino acid sequence can be arbitrary except for the first monomer, which is always methionine. The exact aminoacid sequence of a protein is encoded in the DNA coding for that protein. The process of protein synthesis and the mapping rule from the nucleotide to the aminoacid sequence is described in Section 3.4.3. The 20 aminoacids have different biochemical properties. In particular, depending on the polarity of the side chain, amino acids vary in their hydrophilic or hydrophobic character. These properties are important in protein structure and protein-protein interactions [Cre93]. In order for the protein to be able to fulfil its function within the cell it has to assume its characteristic 3-D form. The process underlying this transition is referred to as protein folding. Even though no complementary binding does take place between amino acids, the folding patterns follow certain regularities. Two of the most easily achievable folding patterns of a protein chain correspond to the most common forms of protein secondary structure, the α-helix and the β-pleated sheets.These patterns are very stable and held together by tight hydrogen bonds [MGH$^+$03]. The three-dimensional arrangement of the folding patterns is the tertiary structure of a protein. Figure 3.5 depicts an exemplary tertiary spatial structure of a protein, showing the α-helices (red) and the β-sheets (yellow).[1] Determining the spatial structure of a protein is a tedious task that can be accomplished by complicated crystallography experiments. The pronounced

[1]Depicted is the N-terminal growth factor-like domain (GFLD) in amyloid precursor protein (PDB ID 1MWP)

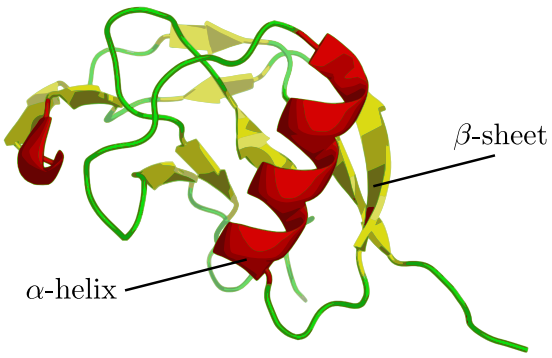


Figure 3.5: Tertiary spatial structure of a protein. Modified from [Wik08].

tertiary structures of proteins give them the ability to specifically and tightly bind to other molecules including DNA, RNA and other proteins. The region of the protein involved in the binding is the so called active site. Often several proteins bind together forming a protein complex. A common role of proteins is to act as enzymes, catalysing chemical reactions involved e.g. in metabolism, transcription, DNA repair and replication. Other roles involve cell signalling and ligand binding and structural tasks like forming different types of tissues. In the context of this work, proteins involved in gene expression, sequence specifically binding to the DNA, will play a central role, see Chapter 7.

3.3 Genetic Information Storage and Engineering

3.3.1 DNA Damage Repair and Error Correction

The complementary binding between the two DNA strands in the double helix makes the DNA macromolecule very stable. Additionally, the complementarity of the two strands can be seen as a simple repetition code. In the cell, DNA is subject to many physical and chemical agents known as mutagens (e.g. ultra violet light, radiation, free radicals). They attack the DNA and chemically modify the bases. Thousands of such modifications take place in the DNA of a human cell every day, however only a few accumulate as mutation in the DNA sequence. The chemically modified bases are efficiently recognized, removed and replaced by DNA excision repair proteins. The intact complementary base is used as template for the replacement [FWS95]. In other words, the four bases represent valid codewords, whereas the chemically modified bases do not, and are recognized as erasures and replaced by using the "repeated" valid codeword on the complementary strand. How important DNA repair is for living organisms, can be seen on the fact that bacteria, possessing a sparse genome, use several percent of it to encode the DNA repair proteins [AJW+08].

3.3.2 DNA Mismatch Repair and Error Correction

DNA mismatch repair is responsible for correcting mismatch errors in the DNA, most of which are introduced during DNA replication. The DNA replication process is noisy in the sense that wrong bases can be complemented. The damage is repaired by DNA mismatch

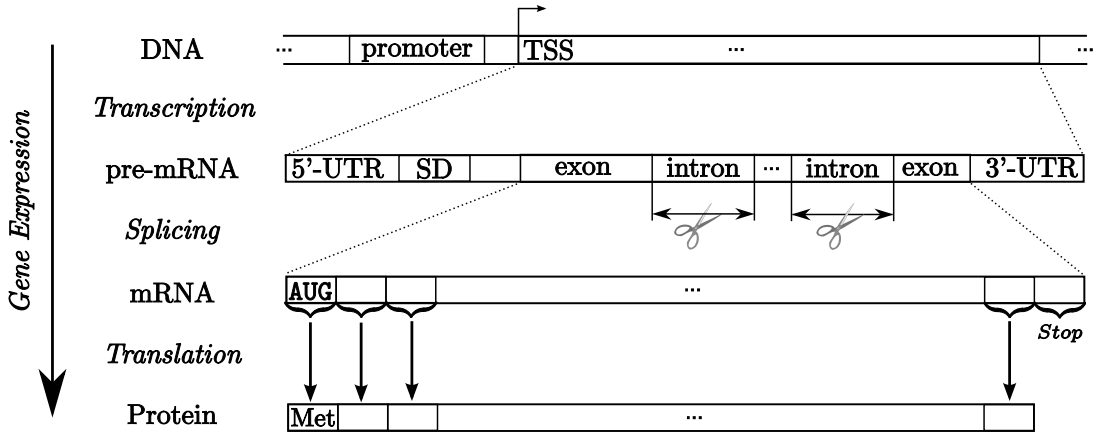


Figure 3.6: Schematic illustration of gene expression. Note that in prokaryotes no splicing takes place since prokaryotic protein coding genes consist of a single exon.

repair proteins capable of recognizing the deformation caused by the mismatch. During replication, the template strand is known to the mismatch repair machinery, allowing it to distinguish the wrongly incorporated nucleotide, excise it and replace it with the correct one.

In [Mac02, Don03], it has been attempted to interpret the donor/acceptor pattern of the three hydrogen bonds in a complementary base pair and the purine/pyrimidine property of a base as a parity check code. The binary interpretation of the three residues and the purine/pyrimidine character would theoretically allow for 16 different bases. In fact, many of these can be successfully synthesized and were found to be processed by the DNA replication machinery. Thus, it can be hypothesized that the nature has chosen the particular subset of the four naturally occurring bases for coding theoretic purposes, in order to minimize the probability of replication errors. In the proposed scheme, the naturally occurring bases represent a subset of codewords with the highest minimum distance to non-complementary counterparts.

3.4 Molecular Biology - Genetic Information Processing

Molecular biology is the study of biology on the molecular level focusing on the interactions between biological macromolecules DNA, RNA and proteins and how these interactions are regulated. Molecular biology has evolved around the so called central dogma of molecular biology, stating that the information transfer follows only in one direction from DNA to mRNA via transcription and subsequently from mRNA to proteins in the process referred to as translation. The whole process of protein production from the information coded in the DNA is referred to as gene expression. The three main molecular processes involved in gene expression - transcription, splicing and translation - are described in detail in the following and are schematically depicted in Figure 3.6.

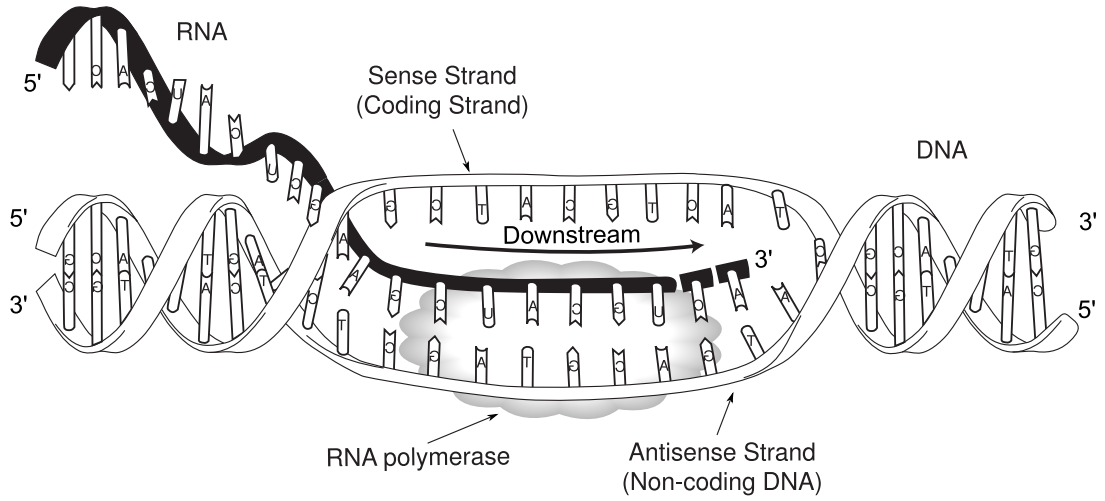


Figure 3.7: Transcription of DNA to RNA by the RNA polymerase. Modified from [Nat08].

3.4.1 Transcription

During transcription, an RNA copy of one of the two DNA strands is synthesized. The DNA is unwound in the transcribed region and the strand complementary to the so called coding strand is used as a template to synthesize the RNA. The copying and synthesis is performed sequentially by the RNA polymerase protein. Transcription is directed. The DNA template strand is read in the 3' to 5' direction, thus the new RNA strand is synthesized from the 5' to the 3' end. Obviously, the transcribed RNA has the same sequence as the coding strand except that `U`-uracil is used instead of `T`-thymine. The situation is depicted in Figure 3.7. The first DNA position that is transcribed, is called the transcription start site (TSS). In molecular biology, directed processing is said to follow downstream along the sequence. Thus, the sequence preceding the TSS is said to be upstream and the transcribed sequence is seen as downstream of the TSS with respect to transcription.

The Transcription Process

Transcription is divided into three main stages: initiation, elongation and termination. Initiation refers to the assembly of the transcription machinery around the TSS and the transcription start. Elongation is the process of transcribing itself. Termination concludes the transcription. The RNA synthesis is stopped and the transcription machinery disassociates from the DNA.

Upstream of the TSS is the so called promoter region designating the transcription initiation site, see Figure 3.6. In prokaryotes the promoter region contains two sequence specific DNA binding sites, recognized by the σ-subunit of the RNA polymerase, situated upstream of the TSS. Transcription initiation is far more complex in eukaryotes. The main difference is that eukaryotic polymerases do not directly recognize their core promoter sequences. In eukaryotes a collection of initiation proteins mediates the binding of

the RNA polymerase and the initiation of transcription. Only after they have attached to the promoter, does the RNA polymerase bind to it. The completed assembly is called the transcription initiation complex. The eukaryotic promoter region contains upstream of the TSS the so called `TATA` box and a pyrimidine or GC-rich initiator region directly around the TSS. The elongation is interrupted at the terminator site. Prokaryotes use two different strategies for transcription termination.

Transcription Regulation

Transcription has to be regulated, since every cell carries a copy of the whole genome, but needs to express different genes at different concentration levels depending on its specialization or current environmental conditions. Transcriptional regulation follows several different mechanisms and takes place in several regulation layers. The basic mechanism is regulation by the promoter sequence, e.g. in bacteria different σ-subunits, binding specifically to different promoter sequences, are used under normal conditions and during heat stress response. This way different groups of genes can be expressed as a reaction to environmental stress. Direct regulation of individual genes is possible by regulatory DNA binding proteins called transcription factors. These can bind to the DNA at specific binding sites around the TSS, acting either as silencers or enhancers. The regulation mechanisms are very varied, from blocking the access of the RNA polymerase to the promoter by overlapping binding, to promoting transcription by assisting the RNA polymerase to bind. Obviously, a particular transcription factor can only regulate genes that have a corresponding binding site. The genes coding for transcription factors are themselves regulated. Additionally, the expression products of different genes (mostly proteins) interact with each other and with other substances in the cell, thereby influencing the transcription rates. These interactions are described by the so called gene regulatory networks. More recently, it has become apparent that a lot of transcriptional regulation is governed by epigenetic (non-DNA-sequence specific) mechanisms, altering the accessibility of DNA to proteins and thereby modulating transcription. Examples include DNA packaging (in particular the chromatin structure in eukaryotes, see Section 3.2.1), chemical modification of DNA (e.g. DNA methylation that can be passed onto daughter cells and plays a key role in cell specialization), as well as non-sequence specific DNA binding proteins. In Chapter 7 the binding sites of proteins involved in transcription regulation will be analysed and their synchronization properties studied.

3.4.2 Splicing

If the transcribed RNA codes for a protein, it is called messenger RNA (mRNA). In eukaryotes gene expression involves an extra step before translation. Transcribed eukaryotic mRNA initially contains segments called introns and exons, whereas only exons are protein coding and the introns have to be spliced out prior to translation. In this state, the mRNA is referred to as pre-mRNA, see Figure 3.6. The splicing of the introns and subsequent joining of the remaining exons is done by the spliceosome. Within the intron, a 3' splice site, a 5' splice site, and a branch site mark sequence specific cut points and the binding site for the spliceosome [Bla03].

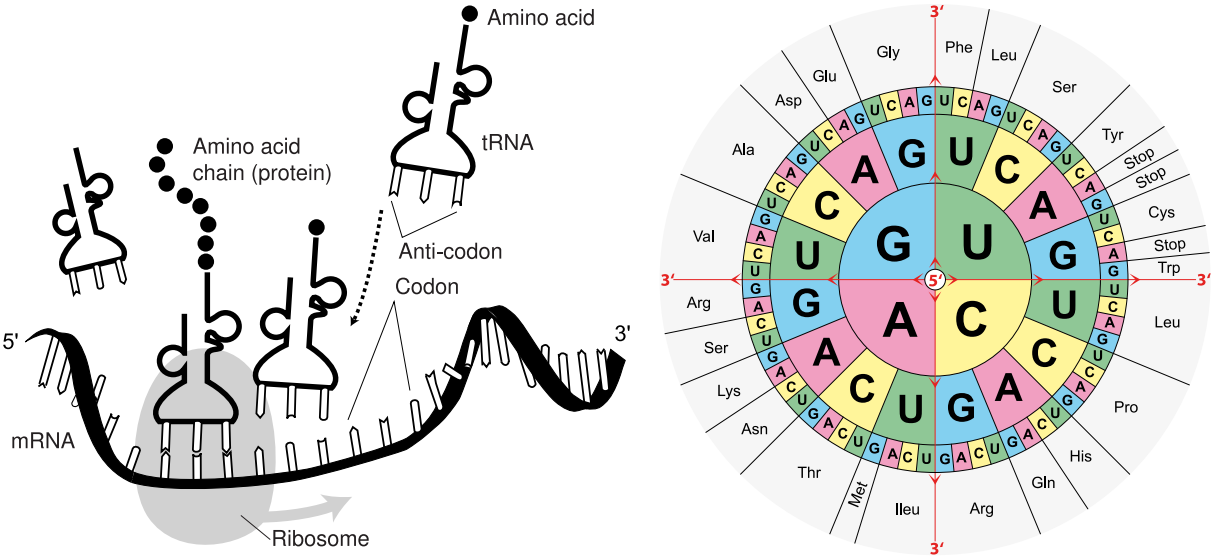


Figure 3.8: Translation of mRNA to protein (left), genetic code (right). Modified from [Nat08] and [Wik08].

3.4.3 Translation

As described in Section 3.2.3, proteins are formed by a linear sequence of amino acids of 20 different types. The process of mapping the mRNA sequence to the amino acid sequence is called translation. Translation takes place in the cytoplasm on a large ribonucleoprotein assembly called the ribosome. The sequence of nucleotides in the mRNA molecule is read consecutively in groups of three referred to as codons. Thus, there exist $4^3 = 64$ different codons. However, there are only 20 different amino acids. The mapping, also called genetic code, is redundant and some codons encode the same amino acid, see Figure 3.8 (right). Adaptor tRNA molecules are responsible for the mapping, see Figure 3.8 (left). An amino acid is first attached to a tRNA molecule, which by complementary base pair binding recognizes the appropriate codon. The genetic code redundancy implies that there are either as many tRNA molecules as there are codons, or that some tRNAs can base pair with more than one codon. In fact, some tRNAs are constructed in a way that they require accurate base pairing only at the first two positions of the codon and can tolerate a mismatch (or wobble) at the third position. Note that many of the alternative codons for an amino acid differ only in their third nucleotide, see Figure 3.8 (right). While the tRNAs are responsible for the mapping of codons to amino acids, it is the ribosome that synthesizes the amino acid polypeptide chain. The translation takes place in the 5' to 3' direction. It begins with the start codon `AUG` that is recognized by a unique initiator tRNA molecule, which encodes for methionine (`Met`), see Figure 3.6. During the elongation phase the mRNA is processed block wise codon by codon. In each step, the ribosome waits for the appropriate tRNA to bind to the currently processed codon. Subsequently, the amino acid attached to the tRNA is added to the growing polypeptide chain before the ribosome moves to process the next codon, see Figure 3.8 (left). The process is repeated until one of the three stop codons `UAA, UAG, UGA` is reached. A release factor then binds to the ribosome, releasing the completed polypeptide chain.

Only the mRNA sequence between the start and the stop codon is translated. The mRNA contains an untranslated region (UTR) on the 5' and the 3' end, see Figure 3.6. The 5' end contains a sequence specific binding site that facilitates the recognition of the translation start. In prokaryotes it is the Shine-Dalgarno (SD) sequence, in eukaryotes the so called Kozak sequence.

3.5 Genetic Information Processing and Engineering

3.5.1 Transcription and Error Correction

During transcription elongation simple proofreading mechanisms ensure the faithfulness of the transcribed RNA. However, they are fewer and less effective than those involved in DNA replication resulting in a lower copying fidelity. While DNA replication achieves an error rate of 10^{-10} per base pair, transcription has an error rate around 10^{-6} [Rad01] per nucleotide. This seems like meaningful trade-off between accuracy and efficiency, since transcribed RNA does not serve the purpose of passing genetic information to progeny. Interestingly, the rate at which the DNA polymerase replicates the DNA is much higher than the rate at which the RNA polymerase transcribes the DNA. However, at the high speed, the DNA polymerase replicates with an error rate of $10^{-3} - 10^{-5}$ and it is the DNA mismatch repair, see Section 3.3.2, that effectively reduces the overall error rate of DNA replication.

The DNA damage repair described in Section 3.3.1 has been found to show higher activity in transcribed regions [FKC+02]. The reason is that the RNA polymerase molecule, performing the transcription, is itself capable of recognizing damaged nucleotides on the template strand and recruit DNA damage repair proteins to it. This guarantees that important, often transcribed, DNA regions are error checked and corrected more often and ensures that they experience less mutations than other regions under the assumption of identical damage rates per time unit across the genome. This partially resembles error correction in magnetic platter hard disk drives. The information signal on the magnetic track is subject to degradation. Present-day drives store the data with high information density using efficient channel codes. If an information block on the drive is found to be damaged on read access but can still be decoded, it is re-encoded and written to a different sector on the drive. Blocks which are read more frequently get error checked and corrected more often.

3.5.2 Translation and Error Coding

The original involvement of information theorists with molecular genetics goes back to the discovery of the genetic code. In the period between the discovery of the DNA structure in 1953 and the decipherment of the genetic code 1961-1969, when no actual DNA sequences and only very few amino acid sequences were known, several different coding schemes describing the mapping of the DNA sequence (4 letter alphabet) to a protein (amino acid sequence from a 20 letter alphabet) were proposed by coding theory experts. Some of

them had high information density, while others had foreseen error correction capabilities. A review of the proposed codes can be found in [Hay98]. The experimental discovery of the actual genetic code (the mapping rule of the $4^3 = 64$ DNA sequence triplets to the 20 amino acids and a stop symbol) was a disappointment for the coding community, since it does not seem to implement any of the two. From this time, there has been little interaction between the two communities until recently. The question why the genetic code has evolved the way it is remains open. Remarkably, it is identical across all domains of life with a few minor exceptions. Recent studies suggest the optimality of the code in terms of error minimization using metrics based on physio-chemical properties of the resulting amino acids like their hydrophobicity [FWK03]. Obviously, errors are differently weighted in terms of their effect on the function, which is not being accounted for by traditional models from communications engineering trying to minimize the overall error number. Other results indicate that the genetic code is optimal in terms of allowing an additional information signal to be modulated on top of it [IA07], e.g. the signal for sequence specific binding sites. Apparently, evolution imposes additional constraints on the optimization of the genetic code, which makes its modelling rather peculiar. One of the constraints seems to be the minimization of the effect of frame shift errors [BVK07]. Since the genetic code is a block code, a frame shift error leads to a completely different amino acid sequence. The used codon mapping ensures that a stop codon is very likely to be encountered soon after a frame shift error. The resulting shortened faulty protein is quickly degraded.

3.5.3 Gene Expression and Marker Synchronization

In all the steps of gene expression, it is necessary that it can be recognized where to start and stop the processing. In transcription the transcription start site has to be found. In splicing the exon intron borders have to be identified and in translation the coding part of the mRNA has to be recognized. This information is typically encoded in the sequence, often in form of sequence specific binding sites as described in this section (e.g. promoter, Shine-Dalgarno sequence, start and stop codons). In engineering, this process strongly resembles the use of marker sequences in synchronization. Binding sites of regulatory proteins can be analogously interpreted as synchronization markers. Parallels between binding site recognition and synchronization, as well as the synchronization properties of selected binding patterns, will be studied in Chapter 7.

3.6 Evolutionary Genetics - Genetic Information Transmission

That the main driving force of evolution and speciation is adaptation to changing environmental conditions by means of natural selection was discovered by Darwin in 1859 [Dar59], long before anything was known about the molecular mechanisms underlying it. Darwin has postulated the theory of universal common descent, stating that any two species have a common ancestor and that speciation follows along a binary evolutionary tree (phyloge-

netic tree) starting from the common ancestor, where each node of the tree corresponds to a speciation event. The main laws of genetic heredity from parents to offspring were discovered by Mendel in 1886 [Men86]. Evolutionary genetics refers to the research field that helped to unite both theories. The link is established by studying evolution in terms of gene changes within a population and by uncovering the processes that convert the variation within population into permanent variations between species. Modern molecular evolutionary genetics studies the evolutionary processes acting upon genomes on the molecular level.

3.6.1 Genome Evolution

According to evolutionary theory, a certain degree of mutation is necessary to allow for adaptation of different species to changing environmental conditions. Propagation of evolutionary disadvantageous mutations is hindered by natural selection in contrast to neutral and the rare advantageous mutations. The living cells do not possess any mechanism for controlled mutation of their genome. On the molecular level the evolution rather depends on "mistakes" followed by non-random survival. Most of the genetic changes simply result from failures in the normal mechanisms of DNA replication and damage repair. Thus, the mutations are the result of DNA damage that has not been repaired and has been passed on to the offspring along the germline. DNA replication and repair mechanisms have been found to operate at an extraordinary high fidelity rate. Once a mutation has been fixed in the species population it becomes permanent. When comparing two evolutionary closely related genomes like human and chimpanzee, the differences will be rather small due to the short evolutionary distance to their common ancestor. However, conserved regions between more distant species like human and mouse are likely to be a result of purifying selection suppressing mutations interfering with their function. The more distant the species are, the more difficult it becomes to identify homologous regions sharing common ancestry since they are likely to have diverged too much.

3.6.2 Types of Mutations

Mutations can be categorized using different criteria. In the context of this work, we want to focus on inheritable mutations affecting the germline cells and not mutations in somatic cell lines (body cells) leading e.g. to cancer. In terms of effects on evolutionary fitness, it can be differentiated between the most frequent neutral mutations occurring at a steady rate that have no harmful or beneficial effect on the organism, deleterious mutations that have a negative effect decreasing the fitness of an organism and rare advantageous mutations that have a positive effect. Only neutral and advantageous mutations have a chance of spreading in the population and becoming permanent. Permanent mutations lead to structural differences in the genomes of separate species. While large scale mutations lead to chromosomal and genome rearrangements, small scale mutations affect one or few nucleotides.

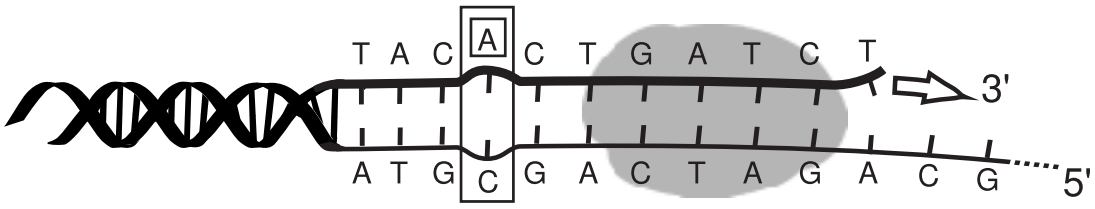


Figure 3.9: Substitutions are primarily caused by mismatch errors introduced during DNA replication. If left uncorrected, in the next replication round the introduced mismatch will lead to a mutation in one of the DNA copies. Modified from [Nat08].

Small Scale Mutations

There are three types of small-scale mutations affecting the DNA: substitutions, insertions and deletions. A substitution exchanges a DNA nucleotide by another one. Substitutions result primarily from DNA replication mismatch errors, see Figure 3.9. These are more likely to occur if the replicated nucleotides are chemically modified due to untreated DNA damage caused by mutagens. A quantitative analysis of evolutionary rates has found that the substitution rate among the human, rat and mouse genome is around 0.65 substitutions per site for DNA positions that evolve neutrally (sites that are not under natural selection) [CBS+04]. The small scale insertion and deletion (InDel) mutations insert/remove a single or multiple nucleotide(s) into/from the DNA strand. InDels are far less likely than substitutions and were found to occur at a rate being only 5% of the substitution rate [CBS+04]. InDels are particularly problematic in exons. If the length of the affected nucleotides is not a multiple of 3, they cause a frame shift in the codon reading frame changing the following amino acid sequence and typically leading to a premature translation stop. The main cause of InDels is believed to be the affinity of single stranded DNA to fold and form loops. The biochemical processes giving rise to substitution and InDel mutations are fundamentally different, and it can be assumed that these events occur independently from each other.

Large Scale Mutations

The major cause of large-scale mutations are DNA double strand breaks. These can result in a chromosome part being deleted, translocated or duplicated. The translocation can be seen as a deletion of a larger region followed by a subsequent insertion of the deleted part into the same or a different chromosome. Analogously a duplication is the insertion of a duplicated portion in the same or a different chromosome. Such large scale duplications play an important role in genome evolution. In particular, gene duplications allow genes to evolve towards new functionality while retaining the original function [Zha03]. Large scale duplications may involve whole chromosomes or even entire genomes [Spr02]. Since DNA is double stranded, an insertion can happen in two different ways, see the right portion of Figure 3.10. Note that the two possibilities a) and b) are flipped realizations of each other. Thus, with respect to the reference genome, the insertion of a duplicated region manifests either as an exact copy or a palindromic copy being the reverse complement. The situation shall be explained in more detail using an example.

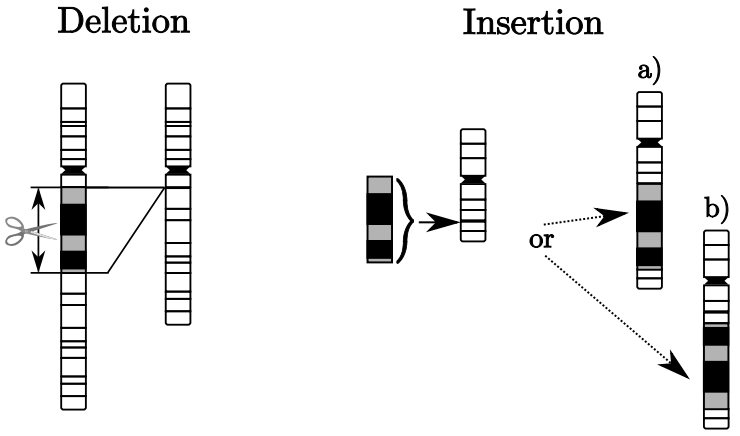


Figure 3.10: Large scale deletion (left). Large scale insertion (right) can take place in two different ways leading either to an exact a) or a palindromic copy b). Modified from [Nat08].

Example: Assume a DNA region with the reference sequence `5'-gtgtag-3'` is duplicated and inserted at another position in the genome in the middle of a region with the reference sequence `5'-...ATCTCCAGTCAGCT ...-3'`. The double stranded DNA snippet duplicate

```
5'-gtgtag-3'
3'-cacatc-5'
```

can be inserted into the new region in the following two ways

```
5'-...ATCTCCAgtgtagGTCAGCT ...-3'    5'-...ATCTCCActacacGTCAGCT ...-3'
3'-...TAGAGGTcacatcCAGTCGA ...-5'    3'-...TAGAGGTgatgtgCAGTCGA ...-5'
```

With respect to the reference sequence (first row), the left insertion leads to an exact duplication and the flipped insertion on the right results in a palindromic duplication. The sequence `ctacac` inserted into the reference strand by the palindromic duplication is the reverse complement of the original duplicated reference sequence `gtgtag`. ∗

When comparing the human and the mouse genome a total of around 180 large scale genome rearrangements are found [AJW+08]. Large scale mutations are thus by far not as frequent as small scale mutations but they influence large regions. In particular, duplications can even involve entire chromosomes or the whole genome. The duplicated copies are subject to small scale mutations after a duplication event and accordingly diverge over evolutionary time. Because of large scale mutations, different genomes can greatly differ in length. Parts that exist in one genome may not be present in another and vice versa. Additionally, parts that both genomes inherited from their common ancestor may be located at different positions in the genomes and may exhibit different nucleotide compositions because of small scale mutations.

3.7 Genetic Information Transmission and Engineering

In the following genetic information transmission over evolutionary time scales shall be modelled. The focus is placed on evolutionary substitutions for which there exist well established statistical models. In Chapter 5 the derived channel model is used to compress the nucleotide portion of multiple genome alignments, in Chapter 7 it is used to study the synchronization properties of mutating binding sites.

3.7.1 Substitution Channel - Continuous Time Markov Process

The evolution of a single nucleotide over time shall be modelled in the following. The nucleotide alphabet is denoted by $\mathcal{A} = \{\texttt{A},\texttt{C},\texttt{G},\texttt{T}\}$. Let the random variable X_0, with realization $x_0 \in \mathcal{A}$, model a single ancestral DNA site, and let X_t represent this site after time t. The time dependent evolutionary discrete memoryless channel described by the probabilities $P(x_t = j|x_0 = i) = p_{ij}(t),\ i,j \in \mathcal{A}$ has to be characterized. Let $X_{t'}$ represent the nucleotide at some intermediate time $0 < t' < t$. Given that the nucleotide mutation rates are not changing over time, it is reasonable to assume that the transition $X_0 \to X_{t'} \to X_t$ form a homogeneous continuous time Markov chain $\{X_t\}_{t\geq 0},\ t \in \mathbb{R}^+$, i.e. for all $h, t \geq 0$, X_{t+h} is conditionally independent of $\{X_s\}_{s\leq t}$ given X_t. Such Markov chain is characterized by a rate matrix $\boldsymbol{R} = [r_{ij}],\ i,j \in \mathcal{A}$ that is stable and conservative [Nor97]

$$0 \leq r_{ij} < \infty \quad \forall i \neq j \qquad \text{(stable)} \tag{3.1a}$$

$$\sum_j r_{ij} = 0 \quad \forall i \qquad \text{(conservative)} \tag{3.1b}$$

Let the probabilities $P(x_t = j|x_0 = i) = p_{ij}(t)$ be the elements of the matrix $\boldsymbol{P}(t)$ describing the discrete memoryless subtitution channel, see Figure 3.11.

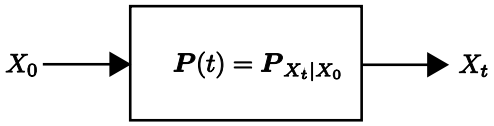


Figure 3.11: The evolutionary time dependent DMC channel.

The rate matrix $\boldsymbol{R}$ and the channel matrix $\boldsymbol{P}(t)$ are related as follows

$$\frac{d}{dt}\boldsymbol{P}(t) = \boldsymbol{P}(t)\boldsymbol{R}, \tag{3.2}$$

The time dependent channel matrix $\boldsymbol{P}(t)$ can thus be computed as [Nor97]

$$\boldsymbol{P}(t) = e^{t\boldsymbol{R}}, \tag{3.3}$$

where $e^{t\boldsymbol{R}}$ denotes the matrix exponential of $t \cdot \boldsymbol{R}$. In Appendix B.1 it is shown how to evaluate the exponential of a matrix. Many properties of the scalar exponential extend to the matrix exponential as well. In particular, note that

$$\boldsymbol{P}(t_1 + t_2) = e^{(t_1+t_2)\boldsymbol{R}} = e^{t_1\boldsymbol{R}} e^{t_2\boldsymbol{R}}. \tag{3.4}$$

The homogeneity of the continuous time Markov process also implies that it is memoryless and stationary with the equilibrium distribution $\boldsymbol{\pi} = (\pi_{\mathtt{A}}, \pi_{\mathtt{C}}, \pi_{\mathtt{G}}, \pi_{\mathtt{T}})$. The equilibrium distribution fulfils

$$\boldsymbol{\pi}\boldsymbol{R} = \mathbf{0} \tag{3.5a}$$
$$\boldsymbol{\pi}\boldsymbol{P}(t) = \boldsymbol{\pi}. \tag{3.5b}$$

Additionally, for any initial distribution $\boldsymbol{\lambda}$ the continuous time process generated by $\boldsymbol{R}$ converges to the equilibrium distribution [Nor97]

$$\lim_{t\to\infty} \boldsymbol{\lambda}\boldsymbol{P}(t) \to \boldsymbol{\pi}. \tag{3.6}$$

In the framework of evolutionary modelling the continuous time Markov process is assumed to be reversible [Yan06]. In other words, the expected amount of substitutions from nucleotide i to nucleotide j is equal to the amount of change from j to i in steady state $\forall i, j \in \mathcal{A}$, hence:

$$\pi_i r_{ij} = \pi_j r_{ji} \tag{3.7a}$$
$$\pi_i p_{ij}(t) = \pi_j p_{ji}(t). \tag{3.7b}$$

With $\boldsymbol{\Pi} = \operatorname{diag}(\boldsymbol{\pi})$ it can be equivalently written

$$\boldsymbol{\Pi}\boldsymbol{R} = \boldsymbol{\Pi}(\boldsymbol{S}\boldsymbol{\Pi}) = (\boldsymbol{S}\boldsymbol{\Pi})^T\boldsymbol{\Pi} = \boldsymbol{R}^T\boldsymbol{\Pi}, \tag{3.8}$$

where the rate matrix $\boldsymbol{R} = \boldsymbol{S}\boldsymbol{\Pi}$ is the product of a symmetric matrix $\boldsymbol{S}$ and $\boldsymbol{\Pi}$. Since $\boldsymbol{\pi}$ is a PMF it has 3 free parameters. Due to the reversibility (3.7a) and conservativeness constraint (3.1b), the symmetric matrix $\boldsymbol{S}$ has 6 free parameters. Thus, the general time reversible evolutionary model (GTR) has exactly 9 free parameters and

$$\boldsymbol{R}_{GTR} = \begin{pmatrix} \star & \pi_{\mathtt{C}}\alpha & \pi_{\mathtt{G}}\beta & \pi_{\mathtt{T}}\gamma \\ \pi_{\mathtt{A}}\alpha & \star & \pi_{\mathtt{G}}\delta & \pi_{\mathtt{T}}\epsilon \\ \pi_{\mathtt{A}}\beta & \pi_{\mathtt{C}}\delta & \star & \pi_{\mathtt{T}}\zeta \\ \pi_{\mathtt{A}}\gamma & \pi_{\mathtt{C}}\epsilon & \pi_{\mathtt{G}}\zeta & \star \end{pmatrix}, \tag{3.9}$$

where the $\star$ symbols on the diagonal are dependent parameters to be computed according to the conservativeness constraint (3.1b) such that the row sums equal zero. The model can be further simplified by imposing the following two additional constraints. First, the steady state substitution rates between purines ($\mathtt{A} \leftrightarrow \mathtt{G}$) as well as between pyrimidines ($\mathtt{C} \leftrightarrow \mathtt{T}$) are identical. Note that the purines are denoted by $\mathtt{R} = \{\mathtt{A},\mathtt{G}\}$ and pyrimidines by $\mathtt{Y} = \{\mathtt{C},\mathtt{T}\}$. This kind of substitutions are called transitions. Second, the transversion rates between purines and pyrimidines ($\mathtt{R} \leftrightarrow \mathtt{Y}$) are identical, see Figure 3.12. Imposing

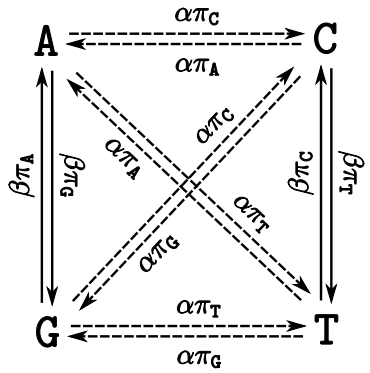


Figure 3.12: Graphical representation of the HKY model. Full lines are transitions and dashed lines represent transversions.

these constraints reduces the overall number of free parameters to 5. The corresponding substitution model is called the Hasegawa, Kishino, and Yano evolutionary model (HKY)

$$\boldsymbol{R}_{HKY} = \begin{pmatrix} \star & \pi_{\mathtt{C}}\alpha & \pi_{\mathtt{G}}\beta & \pi_{\mathtt{T}}\alpha \\ \pi_{\mathtt{A}}\alpha & \star & \pi_{\mathtt{G}}\alpha & \pi_{\mathtt{T}}\beta \\ \pi_{\mathtt{A}}\beta & \pi_{\mathtt{C}}\alpha & \star & \pi_{\mathtt{T}}\alpha \\ \pi_{\mathtt{A}}\alpha & \pi_{\mathtt{C}}\beta & \pi_{\mathtt{G}}\alpha & \star \end{pmatrix}. \tag{3.10}$$

The HKY model accounts for the fact that the rate of transitions has been found to often exceed the rate of transversions $r_{Ts} > r_{Tv}$ [PH99]. This is surprising, since for identical substitution rates $\alpha = \beta$, the transition/transversion ratio is rather expected to equal $r_{Ts}/r_{Tv} = 0.5$. As can be seen from Figure 3.12

$$\frac{r_{Ts}}{r_{Tv}} = \frac{\beta(\pi_{\mathtt{A}} + \pi_{\mathtt{C}} + \pi_{\mathtt{G}} + \pi_{\mathtt{T}})}{\alpha(2\pi_{\mathtt{A}} + 2\pi_{\mathtt{C}} + 2\pi_{\mathtt{G}} + 2\pi_{\mathtt{T}})} = \frac{\beta}{2\alpha} = 0.5. \tag{3.11}$$

In fact, in bacteria the ratio can be as high as $r_{Ts}/r_{Tv} \approx 2$ [Och03]. It is hypothesized, that this is related to the fact that bases of the same type are more similar in the physiochemical sense. Thus, the corresponding mismatches are more difficult to recognize for the mismatch repair machinery. The HKY model has been found to be a reasonable simplification [HRY+03] of the GTR model for most evolutionary studies. In some cases, further simplification is possible by assuming that the mutation rates are identical and thereby reducing the number of free parameters to 4. The resulting model is the so called Felsenstein evolutionary model (FEL)

$$\boldsymbol{R}_{FEL} = \begin{pmatrix} \star & \pi_{\mathtt{C}}\alpha & \pi_{\mathtt{G}}\alpha & \pi_{\mathtt{T}}\alpha \\ \pi_{\mathtt{A}}\alpha & \star & \pi_{\mathtt{G}}\alpha & \pi_{\mathtt{T}}\alpha \\ \pi_{\mathtt{A}}\alpha & \pi_{\mathtt{C}}\alpha & \star & \pi_{\mathtt{T}}\alpha \\ \pi_{\mathtt{A}}\alpha & \pi_{\mathtt{C}}\alpha & \pi_{\mathtt{G}}\alpha & \star \end{pmatrix}. \tag{3.12}$$

Further simplification is possible by assuming that the stationary distribution is equiprobable $\pi_{\mathtt{A}} = \pi_{\mathtt{C}} = \pi_{\mathtt{G}} = \pi_{\mathtt{T}} = 1/4$. This constraint, when applied to the HKY model, leads to the so called Kimura 2 parameter evolutionary model (K2P). Imposed on the FEL model, it leads to the single free parameter Jukes Cantor evolutionary model (JC)

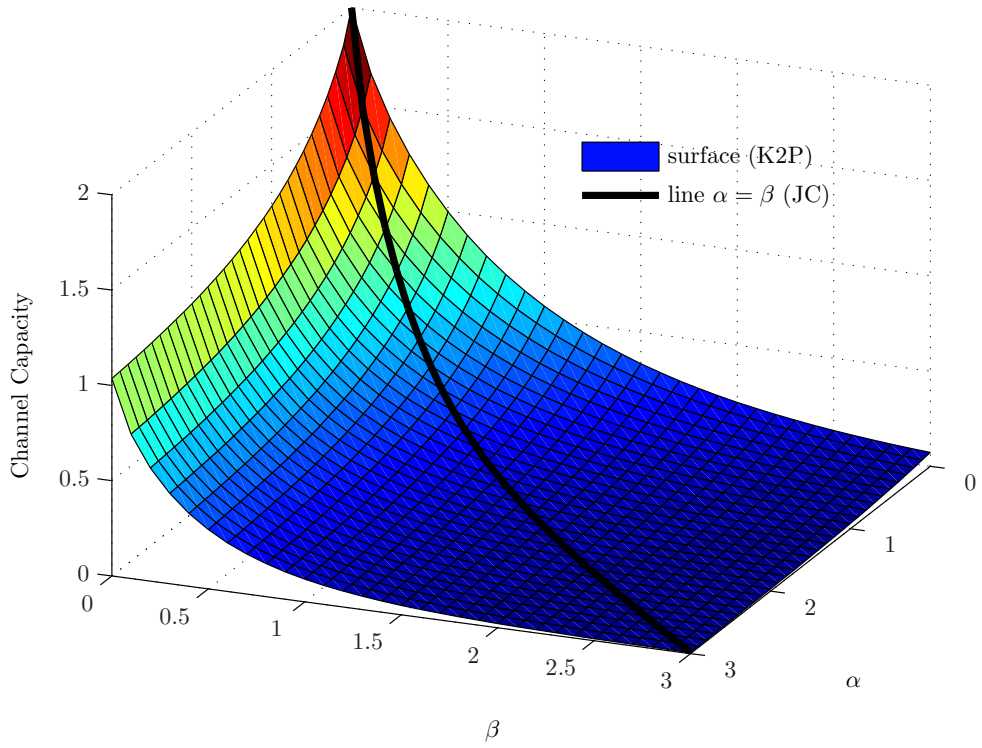


Figure 3.13: Channel Capacity of the K2P and the JC evolutionary model for $t = 1$.

$$\boldsymbol{R}_{K2P} = \frac{1}{4}\begin{pmatrix} \star & \alpha & \beta & \alpha \\ \alpha & \star & \alpha & \beta \\ \beta & \alpha & \star & \alpha \\ \alpha & \beta & \alpha & \star \end{pmatrix} \qquad \boldsymbol{R}_{JC} = \frac{1}{4}\begin{pmatrix} \star & \alpha & \alpha & \alpha \\ \alpha & \star & \alpha & \alpha \\ \alpha & \alpha & \star & \alpha \\ \alpha & \alpha & \alpha & \star \end{pmatrix}. \tag{3.13}$$

The corresponding channel matrices can be evaluated [Yan06]. For the K2P model, the channel matrix $\boldsymbol{P}_{K2P}(t)$ is characterized by

$$p_{ij}(t) = \begin{cases} \frac{1}{4} + \frac{1}{4}e^{-\alpha t} + \frac{1}{2}e^{-\frac{(\alpha+\beta)}{2}t} & \text{if } i = j \quad \text{no mutation} \\ \frac{1}{4} + \frac{1}{4}e^{-\alpha t} - \frac{1}{2}e^{-\frac{(\alpha+\beta)}{2}t} & \text{if } i \to j \quad \text{transition} \\ \frac{1}{4} - \frac{1}{4}e^{-\alpha t} & \text{if } i \to j \quad \text{transversion} \end{cases}. \tag{3.14}$$

For the JC model $\boldsymbol{P}_{JC}(t)$ is characterized by

$$p_{ij}(t) = \begin{cases} \frac{1}{4} + \frac{3}{4}e^{-\alpha t} & \text{if } i = j \text{ no mutation} \\ \frac{1}{4} - \frac{1}{4}e^{-\alpha t} & \text{if } i \neq j \text{ mutation} \end{cases}. \tag{3.15}$$

The rows and columns of the resulting channel matrices $\boldsymbol{P}(t)$ are permutations of each other. Thus, the K2P and the JC evolutionary channel models are symmetric DMC channels and their capacity can be computed using (2.34). Figure 3.13 depicts the channel capacity for the K2P model (surface) and the JC model (line). It can be seen that the capacity is exponentially decreasing. A typical K2P scenario with a transition/transversion ratio $r_{Ts}/r_{Tv} > 0.5$ would correspond to a point on the plane portion left of the JC line.

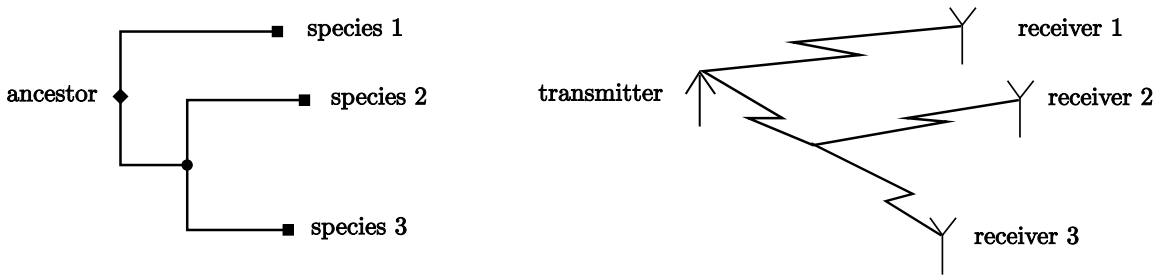


Figure 3.14: Analogy between nucleotide evolution on a phylogenetic tree and the SIMO transmission model.

3.7.2 Nucleotide Evolution is SIMO Transmission

In terms of communications engineering, evolution can be regarded as a single input multiple output (SIMO) transmission system. The common ancestor is the single transmitter and the currently living species are the receivers. The genome of the common ancestor is the transmitted message and the genomes of the currently living species are the received messages. The phylogenetic tree describing the evolutionary relationships corresponds to the signal paths from the transmitter to the receivers. Speciation events, being the inner nodes in the phylogenetic tree, are the scattering points in the signal paths, see Figure 3.14. The SIMO transmission model is particularly well suited to describe the evolution of single homologous nucleotides. However, it is not suited to describe the evolution of whole genomes, in particular it cannot account for large scale mutations like duplications and translocations leading to rearrangements in the transmitted genome. The modelling capability for small scale InDel mutations within homologous regions, sharing common ancestry, is also limited. Probabilistic modelling of the InDel events is difficult. In information theory, the insertion and deletion channel is hard to treat analytically, e.g. the capacity for the general case is still unknown. Additionally, the InDel events typically stretch across several consecutive nucleotides, preventing symbol-wise independent channel models as used for substitutions in general acting upon single nucleotides. However, since the InDel and the substitution process seem to be largely uncorrelated and independent, they can be modelled separately. In [CBS+04] the independence was shown for the human, rat and mouse. For the purpose of this work, the SIMO channel will be solely used to model the evolution of homologous nucleotides.

Substitution SIMO channel

Let us assume that N homologous nucleotides can be observed in N species. According to the evolutionary theory, their ancestor has diverged into two independent nucleotides some time ago, and any of those two nucleotides could have become the ancestor of another two and so on. The generalized homologous nucleotide evolution model is depicted in Figure 3.15. The relationship of the observed nucleotides is described by the phylogenetic tree being a tree like graph $\mathcal{T}$ with nodes representing DNA nucleotides and edges the phylogenetic relationship. An edge is drawn from node u to v if u is an ancestor of v. The node u is the *parent* of v, and v is the *child* of u. A phylogenetic tree is binary, i.e. every node has either one parent or none, and every node has either two children

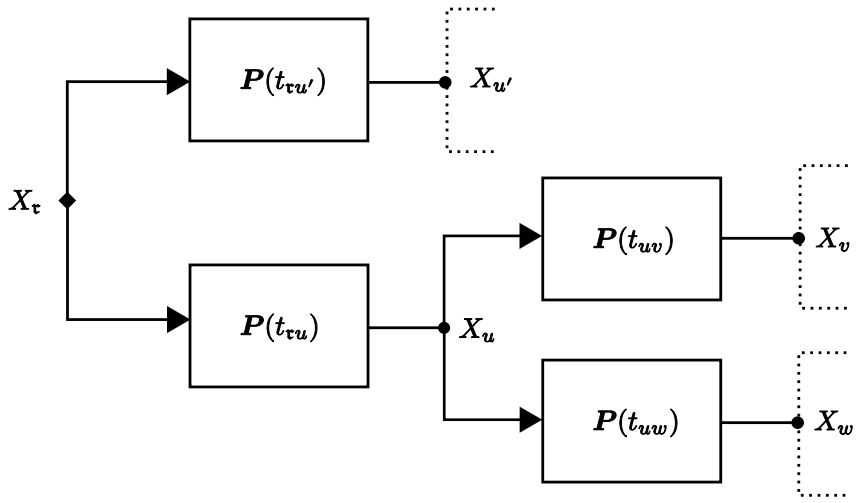


Figure 3.15: The SIMO channel model for homologous nucleotides.

or none. The single node without parents is called the *root* of the tree and is denoted by $\mathfrak{r}$. Nodes without children are called leaves and represent the observed homologous nucleotides in species that exist today. Each branch in $\mathcal{T}$ has a certain length representing the evolutionary distance between the connected nodes associated with it. The transition probabilities between any two nodes u and v in $\mathcal{T}$ connected by a branch are consequently given by

$$\boldsymbol{P}(t_{uv}) = e^{t_{uv}\boldsymbol{R}}, \tag{3.16}$$

where $t_{uv} \in \mathbb{R}^+$ is the evolutionary distance between u and v.

The evolution of a single nucleotide on a tree $\mathcal{T}$ with branch lengths $\tau = \{t_{uv} : (u, v) \in \mathcal{T}\}$ and N leaves can be described as follows. Starting from nucleotide $x_{\mathfrak{r}}$ in the common ancestor, at any node u the nucleotide x_u is transmitted to each child v with transition probabilities specified by (3.16). This process continues until the nucleotides x_{ℓ_i}, $i = 1, .., N$ are observed at the leaves. Due to stationarity, the nucleotide distribution at any node in the tree is identical to the equilibrium distribution $\boldsymbol{\pi}$ of the process.

3.7.3 Estimation on the Tree - Felsenstein Algorithm

In order to determine the common ancestor, the nucleotide, most likely to have been transmitted over the SIMO channel, has to be estimated given the nucleotides at the leaves and the evolutionary model $\{\boldsymbol{R}, \boldsymbol{\pi}, \mathcal{T}, \tau\}$. Let $\boldsymbol{x}_\ell = [x_{\ell_1}, ..x_{\ell_N}]$ denote the observable nucleotides at the leaves. The maximum likelihood estimate of the common ancestor nucleotide $\hat{x}_{\mathfrak{r}}$ is then equal to

$$\hat{x}_{\mathfrak{r}} = \arg\max_{x_{\mathfrak{r}}} p(\boldsymbol{x}_\ell | x_{\mathfrak{r}}). \tag{3.17}$$

Due to the stationarity assumption of the underlying Markov substitution process $p(x_{\mathfrak{r}}) = \pi_{x_{\mathfrak{r}}}, \forall x_{\mathfrak{r}} \in \mathcal{A}$, the maximum a-posteriori estimate can also be computed as

$$\hat{x}_{\mathfrak{r}} = \arg\max_{x_{\mathfrak{r}}} \left(p(\boldsymbol{x}_\ell | x_{\mathfrak{r}}) \pi_{x_{\mathfrak{r}}} \right), \tag{3.18}$$

The probability of observing a single nucleotide at leaf ℓ_i is simple and given as

$$p(x_{\ell_i}) = \sum_{x_\mathfrak{r} \in \mathcal{A}} p(x_{\ell_i}|x_\mathfrak{r}) p(x_\mathfrak{r}) = \sum_{x_\mathfrak{r} \in \mathcal{A}} p(x_{\ell_i}|x_\mathfrak{r}) \pi_{x_\mathfrak{r}}. \tag{3.19}$$

Because of the property expressed in (3.4), the transition probabilities are calculated as $e^{t_{\mathfrak{r}\to\ell_i}\boldsymbol{R}}$, where $t_{\mathfrak{r}\to\ell_i}$ is just the sum of the distances between all nodes leading from the root $\mathfrak{r}$ to leaf ℓ_i. The calculation of the joint probability $p(\boldsymbol{x}_\ell)$ is more complicated and has to take into account the phylogenetic dependencies among nucleotides. An efficient procedure for calculating the likelihood was presented by Felsenstein in 1981 [Fel81]. Felsenstein's algorithm is an application of the belief propagation sum-product algorithm to tree-like graphs. Belief propagation algorithms were developed independently by communications engineers for channel decoding purposes [JBCR74, Pea82]. Let $\boldsymbol{x}_{\ell(u)}$ denote the observations at the leaves of the subtree rooted at u, and let v and w be children of u, see Figure 3.15. Note that $\boldsymbol{x}_{\ell(\mathfrak{r})} = \boldsymbol{x}_\ell$ and $\boldsymbol{x}_{\ell(u)} = [\boldsymbol{x}_{\ell(v)}, \boldsymbol{x}_{\ell(w)}]$. Since X_v and X_w are conditionally independent, given X_u, the conditional probability $p(\boldsymbol{x}_{\ell(u)}|x_u)$ can be expressed recursively as

$$\begin{aligned} p(\boldsymbol{x}_{\ell(u)}|x_u) &= p(\boldsymbol{x}_{\ell(v)}, \boldsymbol{x}_{\ell(w)}|x_u) \\ &= p(\boldsymbol{x}_{\ell(v)}|x_u) p(\boldsymbol{x}_{\ell(w)}|x_u) \\ &= \left(\sum_{x_v \in \mathcal{A}} p(\boldsymbol{x}_{\ell(v)}|x_v) p(x_v|x_u) \right) \times \left(\sum_{x_w \in \mathcal{A}} p(\boldsymbol{x}_{\ell(w)}|x_w) p(x_w|x_u) \right), \end{aligned} \tag{3.20}$$

where the transition probabilities $p(x_v|x_u)$ and $p(x_w|x_u)$ are known from $\boldsymbol{P}(t_{uv})$ and $\boldsymbol{P}(t_{uw})$, see (3.16), and $p(\boldsymbol{x}_{\ell(v)}|x_v)$ and $p(\boldsymbol{x}_{\ell(w)}|x_w)$ were calculated in previous iterations. The algorithm starts at the leaves with initial probabilities

$$p(x|x_{\ell_i}) = \begin{cases} 1 & \text{if } x = x_{\ell_i} \\ 0 & \text{otherwise} \end{cases}, \tag{3.21}$$

for a leave node ℓ_i and proceeds up to the root in an iterative fashion according to (3.20). Finally, the common ancestor can be computed according to (3.17) or (3.18). Also the probability of the observed N nucleotides can be computed as

$$p(\boldsymbol{x}_\ell) = \sum_{x_\mathfrak{r} \in \mathcal{A}} p(\boldsymbol{x}_{\ell(\mathfrak{r})}|x_\mathfrak{r}) \pi_{x_\mathfrak{r}}. \tag{3.22}$$

3.7.4 Substitution Space Time Model

The substitution model described above accounts for the *temporal* evolution of a single DNA site leading to multiple observations. Computational genome analysis and biological experiments show that different sites of DNA exhibit different rates of substitutions. The substitution rates were found to be generally lower at functional sites due to natural selection pressure hindering mutation accumulation. To model the variable rates, a discrete process Θ_j is introduced, where the index j corresponds to the nucleotide site position

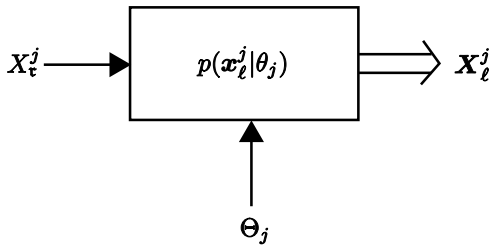


Figure 3.16: Time-Space model of the nucleotide evolution accounting for the spatial rate variation.

in the ancestor sequence. This spatial process *interacts* with the time-process by scaling the distances of the tree at each position j by the factor θ_j, which is a realization of the process. The corresponding nucleotide position then evolves along the phylogenetic tree with transition probabilities between the nodes u and v depending on θ_j

$$\boldsymbol{P}(t_{uv})^j = e^{\theta_j t_{uv} \boldsymbol{R}}. \tag{3.23}$$

The space-time transmission model accounting for variable rates is depicted in Figure 3.16. Different models for Θ_j were proposed [Yan06,FC96,MFP05]. Typically, Θ_j is assumed to have a Gamma like distribution and show Markov type dependencies based on the biological observation that neighbouring sites of the DNA sequences generally have correlated rates. In a sense, the behaviour of Θ_j resembles that of fading in slow fading channels. The rate variation θ_j can be estimated by using numerical optimization methods with the Felsenstein estimator [Nie97].

3.7.5 Summary of the Nucleotide Evolution Model

The parameters used to model the nucleotide evolution resulting from substitution mutations are summarized in Table 3.1.

Parameter	Description	Variability
$\mathcal{T}$	phylogenetic tree	invariable
τ	tree branch lengths	invariable
$\boldsymbol{R}$	substitution rate matrix	invariable for large sections
$\boldsymbol{\pi}$	stationary distribution	invariable for large sections
θ_j	rate variation	variable between sites

Table 3.1: Overview of parameters specifying the nucleotide evolution model.

3.8 Summary

This Chapter has provided a brief overview of the various aspects of genetic information storage, processing and transmission from an engineer's perspective. The most important insights for the follow-up chapters shall be summarized briefly.

The entire genetic information of an organism is stored in its DNA as a discrete signal from a quaternary nucleotide alphabet $\mathcal{A} = \{\texttt{A,C,G,T}\}$. The two strands of the DNA double helix are complementary. Thus, the entire genome can be described by using one of the strands, namely the reference strand. The complementary binding takes place between the base pairs A=T and C≡G. With respect to binding strength, it can be distinguished between weak W={A,T} and strong S={C,G} bases. In terms of size, bases are subdivided in purines R={A,G} and pyrimidines Y={C,T}. While the RY-distribution seems random along the genome, the WS-distribution, the so called GC-content, varies for different regions.

The best studied genetic information processing mechanism is gene expression - the process of producing proteins from the information encoded in the DNA. The processing steps differ in the two domains of life - the simple single cellular prokaryotes comprising bacteria and archaea, and the more complex generally multicellular eukaryotes comprising animals and plants. In both domains, DNA is first transcribed into RNA using complementary binding. The RNA resembles single stranded DNA. In eukaryotes, the transcribed RNA contains non-coding regions that have to be spliced out before translation. During translation the RNA is sequentially mapped in blocks of three nucleotides to an amino acid sequence following a fixed mapping rule called the genetic code. In order to assume its function, the amino acid sequence of a protein has to fold into a certain 3-D shape. The steps of gene expression are regulated by different regulatory proteins binding sequence specifically to the DNA or RNA and acting either as repressors or activators. Sequence specific binding sites are also used to mark the spot for the gene expression machinery, where to start the processing. This very much resembles the use of markers in synchronization in engineering and will be studied in detail in Chapter 7.

Nature seeks a compromise between the degree of fidelity of the genetic information transmission from generation to generation and the mutational level necessary for evolutionary adaptation to changing environmental conditions. The living cells do not possess any mechanisms for directed mutational adaptation. The source of mutations are random DNA damage and DNA replication errors. Adaptation takes place via natural selection affecting which mutations become permanent in a population. There exist efficient mechanisms correcting most of the DNA damage. Likewise, the noisy DNA replication process is complemented by highly efficient mismatch repair. Both error correcting mechanisms greatly reduce the overall error rate. Changes in the DNA sequence that become fixed are called mutations. In terms of the effect on the genome structure, it is distinguished between large scale mutations (i.e. deletions, duplications and translocations) causing genome rearrangements and more frequent small scale mutations (i.e. short insertions, deletions and substitutions) acting locally. The substitution process, being the most frequent type of mutation, can be modelled as a discrete memoryless channel using continuous time Markov models. Thus, the nucleotide evolution resembles the single input multiple output (SIMO) transmission systems. The mutational models of genome evolution play an important role when devising compression algorithms for genomic data. This shall be discussed in detail in Chapter 5.

Part I

Source Coding Aspects in Genetics

4

Compression in Engineering

The main motivation for digital data compression is to reduce the amount of space necessary to store or transmit a message. The Morse code, invented in 1838 for the use in telegraphy, is an early example of digital data compression. In telegraphy short and long pulses are used for data transmission. A binary pulse sequence is used for the transmission of each letter. Morse has come up with the idea to use shorter binary codewords for the transmission of letters such as "e" and "t" that are more frequent in the English language, thereby reducing the average transmission time of a message. Modern work on data compression began in the late 1940s with the introduction of information theory focusing solely on statistical properties of the encoded data. In the framework of information theory the letters are regarded as symbols generated by a source being the language and having certain statistical properties. With the source coding theorem, Shannon has established that lossless compression of symbols from a source is possible as long as the compression rate is higher or equal to the entropy of the source [Sha48]. In 1949, Shannon and Fano devised a systematic way to assign codewords to encoded symbols based on their distribution. An optimal assignment method was found by Huffman [Huf52] in 1951. The compression efficiency can be further increased by combining several symbols to blocks. Unfortunately, the maximum block length is highly limited for Huffman coding since its complexity grows exponentially with increasing block length. In 1979, arithmetic coding was introduced by Rissanen [RL79]. Although slightly inferior to Huffmann in terms of compression rate, its complexity scales linearly with increasing block length. Both Huffman and arithmetic coding belong to the category of entropy coders, asymptotically achieving entropy for large blocks of symbols. However, they both require prior knowledge of the symbol distribution. Symbolwise entropy coding can be combined with statistical prediction algorithms, using higher order context models to refine the coding distribution for each encoded symbol. Different prediction algorithms, based on learn-

ing variable order Markov models and using suitable model mixing strategies to make a prediction, have been developed. Particularly well performing in terms of compression efficiency, computational complexity and memory requirement are Prediction by Partial Matching introduced by Cleary and Witten [CW84] and Context Tree Weighting invented by Willems et al. [WST95]. The latter is among the very few algorithms offering both, theoretical guaranteed asymptotic and good practical performance. Since these prediction algorithms do not require a-priori knowledge of the statistics of the underlying Markov models and are capable of learning it during compression, they are referred to as universal compressors. Another class of algorithms, belonging to the category of universal compressors, are dictionary based algorithms using the idea of encoding the message by pointers to repeating patterns in the already encoded portion of the message. Originally introduced by A. Lempel and J. Ziv [ZL77] in 1977, the dictionary based algorithms are particularly well suited for the compression of text data. All the mentioned compression algorithms are *lossless*, meaning that the original message can be reconstructed error free from the compressed output. Many of the basic algorithms exist in several refined variants. The overview provided in the following is by far not exhaustive. It is restricted only to selected representatives, variants and concepts that will be required in Chapter 5 in order to develop lossless compression algorithms suitable for genetic data.

4.1 Entropy Coding

An analysis of entropy coding including detailed derivations and proofs can be found for example in [CT06]. In data compression the basic idea is to assign short codewords to symbols (or blocks of symbols) with high frequency and long codewords to seldom ones. The message to be encoded is assumed to have been generated by an IID source $\boldsymbol{X}$ modelled as a series of random variables $\boldsymbol{X} = (X_1, X_2, \dots)$ generating symbols $\boldsymbol{x} = (x_1, x_2, \dots)$, where $x_i \in \mathcal{X}, \forall i$. A source code C is a mapping of $\mathcal{X}$ to the set of finite length strings of symbols from a discrete alphabet. Throughout this work, the codewords are assumed to be binary. Let $C(x)$ denote the codeword corresponding to x and $|C(x)|$ denote its length measured in bit. The expected length of the code $E\{|C(x)|\}$ is

$$E\{|C(x)|\} = \sum_{\forall x \in \mathcal{X}} P_X(x)|C(x)|. \tag{4.1}$$

In order to make instantaneous decoding possible, the set of codewords has to be prefix-free, meaning that no codeword is a prefix of any other codeword. Prefix-freedom makes the end of each codeword instantaneously recognizable in the encoded data stream, even though the codewords have different lengths. Short codewords cannot be assigned to all source symbols if the code is to be prefix-free. The set of possible codeword lengths for instantaneous codes is limited by the Kraft inequality [Kra49]

$$\sum_{\forall x \in \mathcal{X}} 2^{-|C(x)|} \leq 1. \tag{4.2}$$

The expected length of the code in (4.1) is minimized under the constraint imposed by the Kraft inequality if codeword lengths $|C(x)|$ are chosen according to the PMF of X,

ideally $|C(x)| = \operatorname{ld} \frac{1}{P_X(x)}$. However, the codeword length has to be rounded up to the nearest integer due to the restriction of having to use integer codeword lengths. Thus, for an optimal binary prefix-free code

$$|C(x)| = \left\lceil \operatorname{ld} \frac{1}{P_X(x)} \right\rceil . \tag{4.3}$$

The difference of the actual codeword length to the ideal one is the codeword redundancy

$$\delta(x) = |C(x)| - \operatorname{ld} \frac{1}{P_X(x)}. \tag{4.4}$$

For optimal binary prefix-free codes $0 \leq \delta(x) < 1$ bit. The performance of a code is measured in terms of the average amount of bits used per source symbol also referred to as the code rate R. For symbolwise coding and IID sources the code rate $R = E\{|C(x)|\}$ bits/symbol and is lower bounded by the entropy $H(X) \leq R$. The assignment of codeword lengths according to (4.3) leads to a code rate R within 1 bit of the symbol entropy

$$H(X) \leq R < H(X) + 1. \tag{4.5}$$

Obviously, the lower bound is achieved if $\forall x \in \mathcal{X}\ \exists n \in \mathbb{N} : P_X(x) = \frac{1}{2^n}$. Especially codes for highly non-uniform distributions are likely to result in high code rates up to the overhead of 1 bit/symbol. By encoding blocks of N symbols jointly $\boldsymbol{x}^N = (x_1 \ldots x_N)$, the codeword redundancy is $0 \leq \delta(\boldsymbol{x}^N) < 1$ bit. The average amount of bits used per source symbol is $R = E\{|C(\boldsymbol{x}^N)|\}/N$ and it is lower bounded by the entropy rate $\mathcal{H}(\boldsymbol{X}^N)$. For N IID symbols encoded jointly $\mathcal{H}(\boldsymbol{X}^N) = \frac{H(\boldsymbol{X}^N)}{N} = H(X)$ and thus

$$H(X) \leq R < H(X) + \frac{1}{N}. \tag{4.6}$$

Hence, using large block lengths, a code rate R arbitrarily close to the symbolwise entropy $H(X)$ can be achieved for IID sources.

4.1.1 Huffman Coding

An optimal prefix code for a given distribution can be constructed by a simple algorithm developed by Huffman [Huf52]. The algorithm works by creating a binary tree of nodes. The tree is build from the leaves to the root. Each symbol of the source alphabet is represented by a leaf node, which is labelled with the probability of the corresponding symbol. First, the two nodes with the lowest probability are determined. Subsequently, their parent node is created and the sum of the probability weight of its children is assigned to it. One branch to a child is designated with a binary 1, the other with a binary 0. The number of free nodes, i.e. leaves and nodes without a parent is reduced by one. This merging step is performed until there is only one parent node left. This node is the root of the tree and has a probability of one. The prefix-free binary codeword $C(x)$ for each symbol $x \in \mathcal{X}$ is the sequence of the binary zeros and ones assigned to the branches of the tree on the way from the root to the leaf corresponding to the symbol x.

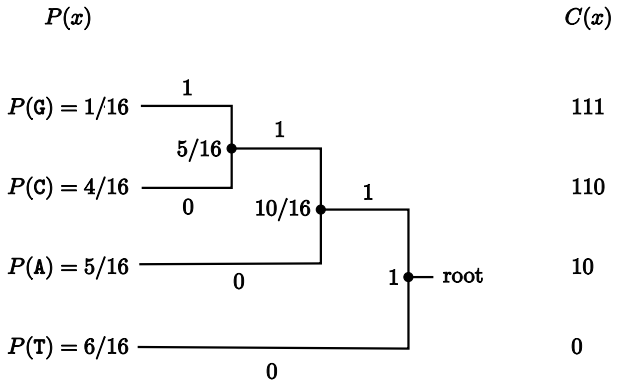


Figure 4.1: Huffman tree and prefix-free Huffman code for the sequence `GATTTACCATTTACCA`.

Example: Figure 4.1 depicts a Huffman tree constructed for the 16 symbols long sequence `GATTTACCATTTACCA` from a four symbol alphabet using blocks of size one. The symbol probabilities were estimated using frequency counts $P(\texttt{G}) = 1/16, P(\texttt{C}) = 4/16, P(\texttt{A}) = 5/16, P(\texttt{T}) = 6/16$. Under the assumption that they correspond to the background distribution the entropy evaluates to $H(X) = 1.81$ bits/symbol. The encoded sequence requires 31 bits as opposed to 32 bits needed without compression and $R = 1.94$ bits/symbol. $*$

Different codes can be built depending on the labelling of the tree branches and on the combination of nodes with the same weights. However, they all achieve the same code rate R. While at the encoder the symbol probabilities of the source can be estimated from the frequency counts in the message to be encoded, the decoder has to know the symbol probabilities a-priori. Thus, these have to be transmitted together with the encoded data. Even though, Huffman coding works optimally and can reach entropy arbitrarily close by combining symbols to large blocks, it is limited by the memory requirement growing exponentially with increasing block length. The reason is that the entire PMF $P(\boldsymbol{X}^N)$ for all $|\mathcal{X}|^N$ possible realizations of a block has to be stored in order to compute the codeword mapping.

4.1.2 Arithmetic Coding

When grouping symbols to blocks, the memory limitation of the Huffman code can be avoided by using arithmetic coding (AC), whose complexity scales linearly with increasing block length. Arithmetic coding is based on the Shannon-Fano-Elias coding, which is slightly inferior to Huffman coding in terms of the codeword redundancy and thus the achievable code rate.

Shannon-Fano-Elias Coding

The Shannon-Fano-Elias coding is based on the idea of computing the codeword from the cumulative distribution function

$$F_X(x) = \sum_{\forall x' : x' \leq x} P_X(x'). \tag{4.7}$$

The calculation of the CDF requires that some kind of fixed ordering is imposed on the symbols in $x \in \mathcal{X}$. This ordering can be arbitrary, e.g. the natural lexicographic ordering can be imposed. Since X is a discrete random variable, $F_X(x)$ is a step function. In order to describe an x, it is sufficient to pick a real number $r \in I(x)$ in the interval $I(x) = [F_X(x) - P_X(x), F_X(x))$. The intervals $\{I(x)\}_{x\in\mathcal{X}}$ are non-overlapping $I(x) \cap I(x') = \emptyset, \forall x' \in \{\mathcal{X}\backslash x\}$ and subintervals of the unit interval $I(x) \subset [0,1), \forall x \in \mathcal{X}$. The binary expansion of the chosen real number r, stripped of the leading '0.', can then be used as a codeword $0.C(x) = (r)_2$. However, an expansion of most real numbers in the binary alphabet is of infinite length, e.g. $(\frac{1}{3})_2 = 0.011101\ldots$. In order to minimize the codeword redundancy, the shortest codeword $C(x)$ in the interval has to be found. This can be accomplished by choosing r initially as the midpoint of the interval

$$r = F_X(x) - \frac{1}{2}P_X(x), \tag{4.8}$$

and truncating $T(.)$ its binary representation to the first $\lceil \mathrm{ld}(2/P_X(x)) \rceil$ bits, see [CT06]

$$0.C(x) = T((r)_2), \qquad |C(x)| = \left\lceil \mathrm{ld}\,\frac{2}{P_X(x)} \right\rceil = \left\lceil \mathrm{ld}\,\frac{1}{P_X(x)} \right\rceil + 1. \tag{4.9}$$

The resulting codewords $C(x), \forall x \in \mathcal{X}$ are prefix-free. The coding redundancy is $1 \leq \delta(x) < 2$ bits and the code achieves a code rate

$$H(X) + 1 \leq R < H(X) + 2. \tag{4.10}$$

By encoding blocks of N IID symbols $\boldsymbol{x}^N = (x_1 \ldots x_N)$ jointly, the bounds of the code rate become tighter and shift towards the entropy

$$H(X) + \frac{1}{N} \leq R < H(X) + \frac{2}{N}. \tag{4.11}$$

Again, by using large block lengths, a code rate arbitrarily close to the symbolwise entropy can be achieved. The described Shannon-Fano-Elias coding procedure is always inferior to Huffman coding in terms of the code rate, see (4.6). However, the difference decreases with increasing block length and both schemes converge to the entropy for $N \to \infty$. Compared to Huffman coding, the knowledge of the PMF of all possible realizations of a block $P(\boldsymbol{X}^N)$ is not necessary. The codeword $|C(\boldsymbol{x}^N)|$ can be computed only from $P(\boldsymbol{x}^N)$ and $F(\boldsymbol{x}^N)$, which can be computed recursively even for sources with memory. For $n = 1 \ldots N$

$$P(\boldsymbol{x}^n) = P_{X_n|\boldsymbol{x}^{n-1}}(x_n|\boldsymbol{x}^{n-1})P(\boldsymbol{x}^{n-1}) \tag{4.12a}$$

$$F(\boldsymbol{x}^n) = F(\boldsymbol{x}^{n-1}) - P(\boldsymbol{x}^{n-1}) \sum_{\forall x' : x' > x_n} P_{X_n|\boldsymbol{x}^{n-1}}(x'|\boldsymbol{x}^{n-1}). \tag{4.12b}$$

Thus, only the conditional PMFs $P_{X_n|\boldsymbol{x}^{n-1}}(X_n|\boldsymbol{x}^{n-1})$ have to be known or estimated for all $n = 1 \ldots N$ in order to be able to apply the Shannon-Fano-Elias coding. Note that for memoryless sources $P_{X_n|\boldsymbol{x}^{n-1}}(X_n|\boldsymbol{x}^{n-1}) = P(X_n)$. The presented recursion assumes arbitrary accuracy for the computation. However, in practice the computation needs to be implementable in fixed-point arithmetic. A fixed-point implementable algorithm was first presented in [RL79] and is referred to as arithmetic coding.

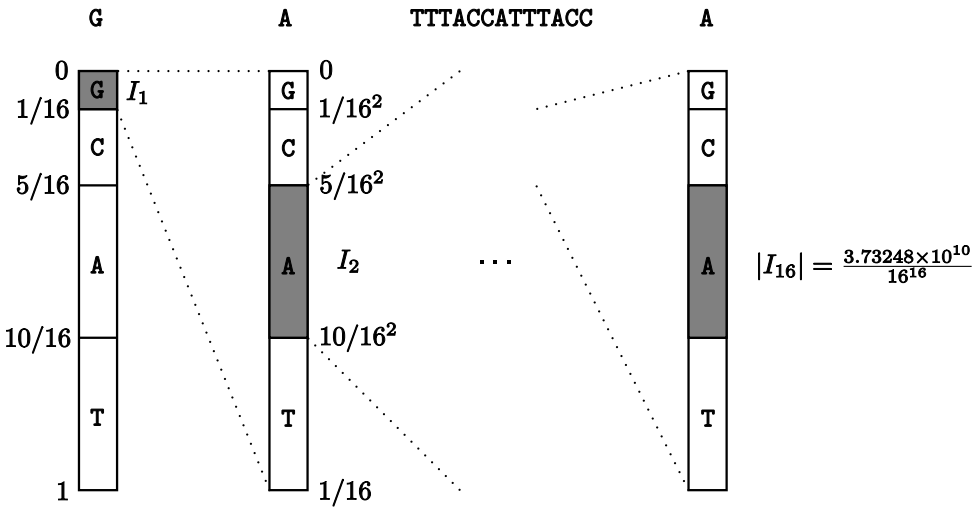


Figure 4.2: Arithmetic coding applied to the sequence GATTTACCATTTACCA.

Arithmetic Coding

The key to an implementation in fixed-point arithmetic is not to consider infinite precision points for the cumulative distribution function but subintervals in the unit interval [RL79]. Let $F(\boldsymbol{x}^N)$ denote the CDF of block probabilities and $F(x_n)$ the CDF of the n-th symbol in the block. The solution relies on a recursive computation of the interval $I_N = [I_N^{low}, I_N^{high}] = [F(\boldsymbol{x}^N) - P(\boldsymbol{x}^N), F(\boldsymbol{x}^N))$ that the codeword must lie. The recursion is defined as

$$I_0 = [0, 1) \tag{4.13a}$$

$$I_n = \left[I_{n-1}^{low} + (F(x_n) - P(x_n)) \cdot |I_{n-1}|, I_{n-1}^{low} + F(x_n) \cdot |I_{n-1}|\right) \qquad n = 1 \ldots N, \tag{4.13b}$$

where $|I_n| = I_n^{high} - I_n^{low} = P(\boldsymbol{x}^n)$ denotes the size of the interval. The first identical bits in the binary representations of the intervals correspond to the codeword $C(\boldsymbol{x}^N) \in I_N$ that Shannon-Fano-Elias coding would assign to $\boldsymbol{x}^n$. Thus, the codeword length satisfies

$$|C(\boldsymbol{x}^N)| = \left\lceil \mathrm{ld} \frac{1}{P(\boldsymbol{x}^N)} \right\rceil + 1, \tag{4.14}$$

and the coding redundancy $1 \leq \delta(\boldsymbol{x}^N) < 2$ bit. With every new input symbol x_n, the corresponding interval I_n becomes shorter. The low and high end of the interval get closer with increasing n and begin to agree in the first few bits of their binary representation. These are also the first few bits of the final codeword $C(\boldsymbol{x}^N)$. They are not going to change with increasing n and thus can be shifted out of the calculation. By effectively scaling the interval in every step, the entire calculation can be done with fixed-point arithmetic. Implementation details are described e.g. in [BCW90].

Example: Figure 4.2 shows the encoding process for the 16 symbols long sequence $\boldsymbol{x}^N$ =GATTTACCATTTACCA. To store this sequence from a four symbol alphabet would require 32 bits when left uncoded. The symbol probabilities were estimated using frequency counts $P(\mathtt{G}) = 1/16$, $P(\mathtt{C}) = 4/16$, $P(\mathtt{A}) = 5/16$, $P(\mathtt{T}) = 6/16$. The final interval

after encoding all symbols is of size $|I_{16}| = P(\boldsymbol{x}^N) = 3.73248 \times 10^{10}/16^{16}$, leading to $|C(\boldsymbol{x}^N)| = \lceil 28.8806 \rceil + 1 = 30$ bits, which is 1 bit less than the 31 bits needed with symbolwise Huffman coding, see Example in Section 4.1.1. This performance gain comes from the fact that arithmetic coding encodes the symbols of the sequence jointly as one block. Note that Huffman coding for the whole block would be infeasible. ∗

The decoding process is similar. The first symbol $\hat{x}_1$ is the sub-interval of the unit interval in which the floating point codeword falls. In other words $\hat{x}_1$ is the symbol with the lowest ordinal number, whose CDF is greater than the encoded real-valued number $r = (0.C(\boldsymbol{x}^N))_{10}$. In every step the bounds of the interval are updated analogously to the encoder and the section of the new interval in which the codeword lies is searched

$$I_0 = [0, 1) \tag{4.15a}$$

$$\hat{x}_n = \arg\min_{x' \in \mathcal{X}} \left(\left(I_{n-1}^{low} + F_{X_n}(x') \cdot |I_{n-1}| \right) - r \right) \tag{4.15b}$$

$$I_n = \left[I_{n-1}^{low} + (F_{X_n}(\hat{x}_n) - P_{X_n}(\hat{x}_n)) \cdot |I_{n-1}|, I_{n-1}^{low} + F_{X_n}(\hat{x}_n) \cdot |I_{n-1}| \right), \tag{4.15c}$$

The recursion has to be terminated after N steps. Thus, N must be a-priori known or additionally transmitted to the decoder. The complexity of encoding and decoding is symmetric.

4.2 Universal Statistical Prediction Based Coding

With arithmetic coding a nearly optimal, memory and computationally efficient solution to encoding long messages was found. However, the PMFs $P_{X_n|\boldsymbol{x}^{n-1}}(X_n|\boldsymbol{x}^{n-1}), \forall n \in \{1 \dots N\}$ characterizing the source have to be known to the encoder/decoder. In order to make arithmetic coding universal, it has to be combined with some statistical prediction algorithm capable of predicting the coding distribution of the next symbol from symbols already encoded/decoded in the past. In case of an IID source, this is known as adaptive arithmetic coding. The prediction based approach was successfully extended even to sources with memory. Such predictors use a mixture of adaptive higher order context models to estimate the coding distribution of the next symbol given the preceding ones. The predicted distribution is used by arithmetic coding to encode the next symbol. The first algorithm of this kind, called Prediction by Partial Matching, was introduced in [CW84]. In [WST95], the Context Tree Weighting method was presented and analytically shown to have desirable asymptotic as well as practical coding properties.

4.2.1 Adaptive Arithmetic Coding

First, let the source be IID, i.e. $P(X_n) = P(X), \forall n \in \{1 \dots N\}$. A suitable adaptive predictor for the coding distribution of the next symbol from symbols already encoded in the past is needed. Such estimators are generally based on the symbol occurrence frequency counts of past symbols. However, using the empirical frequency count directly as predictor neglects unobserved events, e.g. an unlikely symbol might not appear at all in a short message generated by the source and thus would have an empirical frequency

count of zero. This would be detrimental for compression purposes since the log loss, defined as $\log(1/p(x))$, of an unobserved but possible event x is infinite. One possibility to alleviate the situation, is to add a constant pseudo-count to every possible symbol, e.g. the Laplace predictor uses a pseudo-count of 1. Another predictor from the class of the add-constant predictors is the Krichevsky Trofimov estimator (KT) [KT81], which uses a pseudo-count of 1/2. The KT estimator was shown to possess certain optimality properties when the number of possible elements is fixed and the sample size increases to infinity.

Example: Let the source be an IID DNA sequence source and the 16 symbols long sequence $\boldsymbol{x} = \texttt{GATTTACCATTTACCA}$ be the observed sample. Being from a 4 symbol alphabet and using a pseudo-count of 1/2, the overall amount of symbols is $|\boldsymbol{x}|+4\cdot 1/2 = 18$ and the KT predicted source distribution is going to be $\hat{P}(\texttt{A}|\boldsymbol{x}) = \frac{5+1/2}{18}$, $\hat{P}(\texttt{C}|\boldsymbol{x}) = \frac{4+1/2}{18}$, $\hat{P}(\texttt{G}|\boldsymbol{x}) = \frac{1+1/2}{18}$, $\hat{P}(\texttt{T}|\boldsymbol{x}) = \frac{6+1/2}{18}$. ∗

For the purpose of universal compression of IID sources the PMF estimate to be used by the arithmetic encoder is updated with each encoded symbol. Using the KT estimator the PMF estimates are

$$\hat{P}_{X_n|\boldsymbol{x}^{n-1}}(x_n = x'|\boldsymbol{x}^{n-1}) = \frac{f_{x'}(\boldsymbol{x}^{n-1}) + 1/2}{|\mathcal{X}|/2 + n - 1}, \qquad x' \in \mathcal{X}, \tag{4.16}$$

where $f_{x'}(\boldsymbol{x}^{n-1})$ is the count of x' in the subsequence $\boldsymbol{x}^{n-1}$. The estimated probability $\hat{P}(\boldsymbol{x}^n) = \hat{P}(x_n|\boldsymbol{x}^{n-1})\hat{P}(\boldsymbol{x}^{n-1})$ can be computed recursively and the estimated probability of the block $\boldsymbol{x}^N$ to be encoded evaluates to

$$\hat{P}(\boldsymbol{x}^N) = \prod_{n=1}^{N} \hat{P}(x_n|\boldsymbol{x}^{n-1}). \tag{4.17}$$

Example: Let $\boldsymbol{x}^N = 01110$ and the source be binary $x \in \{0, 1\}$

$$\hat{P}(x_n = 1|\boldsymbol{x}^{n-1}) = \frac{f_1(\boldsymbol{x}^{n-1}) + 1/2}{n}, \qquad \hat{P}(x_n = 0|\boldsymbol{x}^{n-1}) = \frac{f_0(\boldsymbol{x}^{n-1}) + 1/2}{n}.$$

The estimated probability $\hat{P}(01110) = \frac{1}{2}\cdot\frac{1}{4}\cdot\frac{3}{6}\cdot\frac{5}{8}\cdot\frac{3}{10} = \frac{3}{256}$. ∗

Instead of the actual probability $P(\boldsymbol{x}^N) = \prod_{\forall x'\in\mathcal{X}} P_X(X = x')^{f_{x'}(\boldsymbol{x}^N)}$, the estimated probability $\hat{P}(\boldsymbol{x}^N)$ is used with arithmetic coding to compute the codeword. Thus,

$$|C(\boldsymbol{x}^N)| = \left\lceil \operatorname{ld} \frac{1}{\hat{P}(\boldsymbol{x}^N)} \right\rceil + 1, \tag{4.18}$$

and an additional parameter estimation redundancy $\operatorname{ld}(P(\boldsymbol{x}^N)/\hat{P}(\boldsymbol{x}^N))$ is introduced into the codeword redundancy. It is upper bounded by

$$\operatorname{ld}(P(\boldsymbol{x}^N)/\hat{P}(\boldsymbol{x}^N)) \leq \frac{|\mathcal{X}| - 1}{2} \operatorname{ld} N + \operatorname{ld}|\mathcal{X}|, \tag{4.19}$$

see [BEY06] for a proof. Therefore, the codeword redundancy of the KT estimator is upper bounded by

$$\delta(\boldsymbol{x}^N) = |C(\boldsymbol{x}^N)| - \operatorname{ld}\frac{1}{P(\boldsymbol{x}^N)} \tag{4.20a}$$

$$< \operatorname{ld}\frac{P(\boldsymbol{x}^N)}{\hat{P}(\boldsymbol{x}^N)} + 2 \tag{4.20b}$$

$$\leq \underbrace{\frac{|\mathcal{X}|-1}{2}\operatorname{ld} N + \operatorname{ld}|\mathcal{X}|}_{\text{parameter estimation}} + \underbrace{2}_{\text{coding}}. \tag{4.20c}$$

Example: Let the source be binary $x \in \{0,1\}$ and $N = 1024$, the individual codeword redundancy is upper bounded by $\delta(\boldsymbol{x}^{1024}) = \frac{1}{2}\operatorname{ld} 1024 + 1 + 2 = 8$ bit. Maximum 2 bits are lost due to the coding redundancy of arithmetic coding and only a maximum of 6 bits is used by the parameter estimation, which is considerably less than would be necessary to store the PMFs separately. ∗

4.2.2 Context Tree Weighting

Up to now, the source was assumed to be memoryless. In the following, the Context Tree Weighting (CTW) prediction algorithm for sources with memory will be presented.

Tree Source

Suppose that the PMF, according to which the next symbol is generated, depends on the preceding generated symbols. A tree can be used to describe such a source. Each node of the tree has exactly $|\mathcal{X}|$ branches and each edge is associated with a symbol from the source alphabet. Thus, every path from a leaf to the root is associated with a unique sequence $\boldsymbol{s}$ and all the leaf node sequences build a set $\mathcal{S}$ of variable length suffix-free sequences, meaning that no sequence $\boldsymbol{s} \in \mathcal{S}$ is a suffix of any other. Consequently, the tree has exactly $|\mathcal{S}|$ leaves and $\frac{|\mathcal{S}|-1}{|\mathcal{X}|-1}$ inner nodes. Each leaf sequence is associated with a PMF $P(X|\boldsymbol{s})$. These PMFs are the parameters of the tree model defined by the set $\mathcal{S}$. In a message $\boldsymbol{x}^N$ generated by a tree source, every realization x_n is, due to the suffix-freedom of $\mathcal{S}$, unambiguously associated with a preceding suffix $\boldsymbol{s}(n) \in \mathcal{S}$ and thus also with a corresponding PMF $P(X_n|\boldsymbol{s}(n))$ according to which it has been generated. An example binary tree source is depicted in Figure 4.3. In order to be able to determine the suffix of the first D symbols $x_n, \forall n = 1 \ldots D$, where $D = \max_{\boldsymbol{s}\in\mathcal{S}}(|\boldsymbol{s}|)$ is the maximum tree depth, the past D symbols preceding $\boldsymbol{x}^N$ have to be known. These past symbols have to be transmitted separately.

Example: Let $\boldsymbol{x}^N = 0100110$ be a sequence generated by the tree source depicted in Figure 4.3 and the sequence 10 be the past symbols. The probability $P(\boldsymbol{x}^N)$ evaluates to

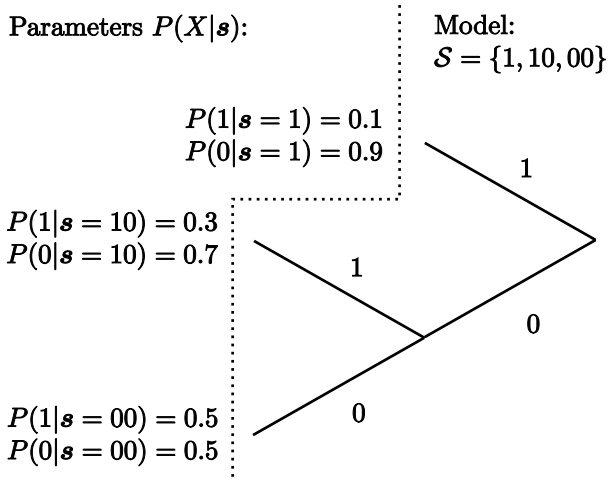


Figure 4.3: Tree source with parameters and model.

past	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
10	0	1	0	0	1	1	0	
$\boldsymbol{s}(n) =$	10	00	1	10	00	1	1	
$P(\boldsymbol{x}^N) =$	0.7 ·	0.5 ·	0.9 ·	0.7 ·	0.5 ·	0.1 ·	0.9	$\approx 10^{-2}$.

*

Given the tree source model $\mathcal{S}$ and the corresponding PMFs, arithmetic coding can be used to encode a message generated by that source. The aim, however, is to device a universal compressor for tree sources capable of learning the tree source parameters (the PMFs associated with each $\boldsymbol{s} \in \mathcal{S}$) as well as its model (the set $\mathcal{S}$ defining the topology).

Tree Source with Unknown Parameters

Let us assume that the model $\mathcal{S}$ is known, whereas the parameters are not. All symbols x_n that have the same suffix $x_n \in \boldsymbol{x}^N : \boldsymbol{s}(n) = \boldsymbol{s}$ form a memoryless subsequence $\boldsymbol{x}_{\boldsymbol{s}}^N$, whose statistics is determined by the PMF $P(X|\boldsymbol{s})$. Thus, a separate estimator $\hat{P}(\boldsymbol{x}_{\boldsymbol{s}}^N)$ can be used for every subsequence of symbols with the same preceding suffix in order to compute $\hat{P}(\boldsymbol{x}^N) = \prod_{\boldsymbol{s}\in\mathcal{S}} \hat{P}(\boldsymbol{x}_{\boldsymbol{s}}^N)$. The codeword length then becomes

$$|C(\boldsymbol{x}^N)| = \left\lceil \mathrm{ld} \frac{1}{\prod_{\boldsymbol{s}} \hat{P}(\boldsymbol{x}_{\boldsymbol{s}}^N)} \right\rceil + 1. \tag{4.21}$$

Example: Let $\boldsymbol{x}^N = 0100110$ be again a sequence generated by a tree source with the model $\mathcal{S}$ depicted in Figure 4.3 and the sequence 10 be the past symbols. However, this time the model parameters are assumed to be unknown. The estimated probability $\hat{P}(\boldsymbol{x}^N)$ is computed as

past	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
10	0	1	0	0	1	1	0	
$\boldsymbol{x}_1^N =$			0			1	0	$= 010$
$\boldsymbol{x}_{10}^N =$	0			0				$= 00$
$\boldsymbol{x}_{00}^N =$		1			1			$= 11$
$\hat{P}(\boldsymbol{x}_1^N) =$			$\frac{1}{2}$	·		$\frac{1}{4}$ ·	$\frac{3}{6}$	$= \frac{1}{16}$
$\hat{P}(\boldsymbol{x}_{10}^N) =$	$\frac{1}{2}$ ·			$\frac{3}{4}$ ·				$= \frac{3}{8}$
$\hat{P}(\boldsymbol{x}_{00}^N) =$		$\frac{1}{2}$ ·			$\frac{3}{4}$ ·			$= \frac{3}{8}$
$\hat{P}(\boldsymbol{x}^N) =$	$\frac{1}{2}$ ·	$\frac{1}{2}$ ·	$\frac{1}{2}$ ·	$\frac{3}{4}$ ·	$\frac{3}{4}$ ·	$\frac{1}{4}$ ·	$\frac{3}{6}$	$= \frac{9}{1024}$

and the codeword length $|C(\boldsymbol{x}^N)| = \lceil 6.83 \rceil + 1 = 8$ bit. ∗

The codeword redundancy is upper bounded by

$$\delta(\boldsymbol{x}^N) < \operatorname{ld} \frac{P(\boldsymbol{x}_{\boldsymbol{s}}^N)}{\prod_{\boldsymbol{s}} \hat{P}(\boldsymbol{x}_{\boldsymbol{s}}^N)} + 2 \tag{4.22a}$$

$$\leq |\mathcal{S}| \gamma \left(\frac{N}{|\mathcal{S}|} \right) + 2, \tag{4.22b}$$

where $|\mathcal{S}|\gamma(\frac{N}{|\mathcal{S}|})$ is the parameter estimation redundancy and

$$\gamma(z) = \begin{cases} z \cdot \operatorname{ld} |\mathcal{X}| & \text{for } 0 \leq z < 1 \\ \frac{|\mathcal{X}|-1}{2} \operatorname{ld} z + \operatorname{ld} |\mathcal{X}| & \text{for } 1 \leq z \end{cases} . \tag{4.23}$$

A detailed proof can be found in [BEY06].

Tree Source with Unknown Parameters and Model

If the tree model is not known, a so called context tree can be used to compute the appropriate coding distributions. A context tree has a topology of a fully blown source tree of chosen depth D, restricted to paths that correspond to contexts (suffices) of length D that have already been observed in the encoded portion $\boldsymbol{x}^n$ of the message $\boldsymbol{x}^N$. Thus, every node of the tree is associated with a unique context $\boldsymbol{c}$ representing the path from that node to the root. Potentially every leaf and inner node of the context tree could be a leaf of the actual source tree model. For such nodes $\mathcal{H}_0 : \boldsymbol{c} \in \mathcal{S}$ the KT estimated probability $P_{\boldsymbol{c}}(n|\mathcal{H}_0) = \hat{P}(\boldsymbol{x}_{\boldsymbol{c}}^n)$ would be appropriate for their description. The alternative $\mathcal{H}_1 : \boldsymbol{c} \notin \mathcal{S}$ that the node is an inner node of the actual source tree, is best described by the product of the probabilities associated with its children $P_{\boldsymbol{c}}(n|\mathcal{H}_1) = \prod_{\forall c \in \mathcal{X}} P_{c\boldsymbol{c}}(n)$. The contexts $c\boldsymbol{c}, \forall c \in \mathcal{X}$ correspond to the set of children of the node with context $\boldsymbol{c}$, e.g. $\{010, 110\}$ are the children of $\boldsymbol{c} = 10$ in the binary case. Since $\mathcal{S}$ is not known, we do not know which of the two hypothesis $\mathcal{H}_0$ and $\mathcal{H}_1$ is valid for $P_{\boldsymbol{c}}(n)$. Thus, the

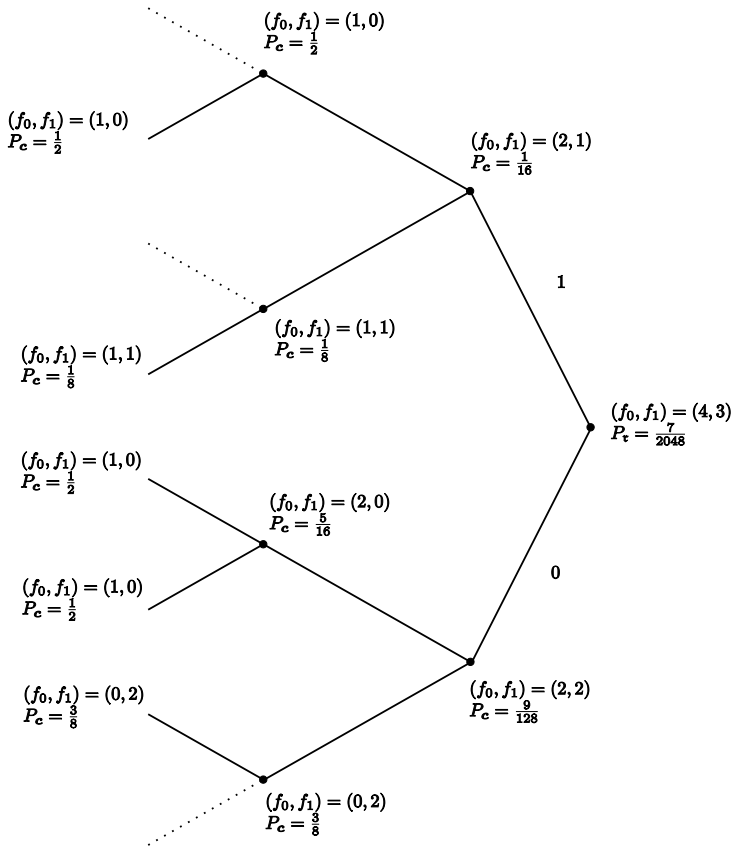


Figure 4.4: Weighted context tree for the message 0100110, passed symbols were 110.

weighted average of the two hypothesis is a good description. Different weighting can be used at different nodes if a-priori knowledge about the tree source is available. With no a-priori knowledge, it is best to use equal weighting of both hypothesis at each node, thus $P_{\boldsymbol{c}}(n) = \frac{P_e(n|\mathcal{H}_0)+P_e(n|\mathcal{H}_1)}{2}$. The following recursive update rule is obtained for the node probabilities $P_{\boldsymbol{c}}(n)$ on the context tree

$$P_{\boldsymbol{c}}(n) = \begin{cases} \hat{P}(\boldsymbol{x}_{\boldsymbol{c}}^n) & \text{if } \boldsymbol{c} \text{ is a leave} \\ \frac{1}{2}\left(\hat{P}(\boldsymbol{x}_{\boldsymbol{c}}^n) + \prod_{\forall c \in \mathcal{X}} P_{c\boldsymbol{c}}(n)\right) & \text{if } \boldsymbol{c} \text{ is an inner node} \end{cases} . \tag{4.24}$$

The recursively obtained probability $P_{\mathfrak{r}}(n)$ at the root $\mathfrak{r}$ of the context tree is the so called context tree weighting estimate $\hat{P}_{CTW}(\boldsymbol{x}^n) = P_{\mathfrak{r}}(n)$ for the probability of the observed sequence $\boldsymbol{x}^n$. In addition to the probabilities $P_{\boldsymbol{c}}(n)$, the frequency count $f_x(\boldsymbol{x}_{\boldsymbol{c}}^n), \forall x \in \mathcal{X}$ has to be stored for every node of the context tree in order to be able to compute $\hat{P}(\boldsymbol{x}_{\boldsymbol{c}}^n)$ using the KT estimator. Given the context tree at $n-1$, computing $\hat{P}_{CTW}(\boldsymbol{x}^n)$ requires only the recursive update from the leaf to the root of the $D+1$ nodes on the path of the context tree that corresponds to the context $x_{n-D} \ldots x_{n-1}$. The PMFs $P(X_n|\boldsymbol{x}^{n-1}), \forall n = 1 \ldots N$ required by the arithmetic encoder can be calculated as

$$\hat{P}_{X_n|\boldsymbol{x}^{n-1}}(x_n = x|\boldsymbol{x}^{n-1}) = \frac{\hat{P}_{CTW}(x\boldsymbol{x}^{n-1})}{\hat{P}_{CTW}(\boldsymbol{x}^{n-1})}, \qquad \forall x \in \mathcal{X}. \tag{4.25}$$

Example: Let 110 be the past symbols and $\boldsymbol{x}^N = 0100110$. The context tree for $\boldsymbol{x}^N$ is depicted in Figure 4.4. Examplary in the node $\boldsymbol{c} = 10$, the subsequence of $\boldsymbol{x}^N$ with 10 as context $\boldsymbol{x}_{10}^N = x_1 x_4$. Consequently, $f_0(\boldsymbol{x}_{10}^N) = 2$ and $f_1(\boldsymbol{x}_{10}^N) = 0$. Thus, the probability $\hat{P}(\boldsymbol{x}_{10}^N) = \frac{3}{8}$ and $P_{10}(N) = \frac{3/8+1/2\cdot 1/2}{2} = 5/16$. ∗

Additional redundancy is needed to encode the tree model. This redundancy is upper bounded by the complexity of the actual source tree that is best described by the overall number of nodes in the source tree being $\frac{|\mathcal{X}|\cdot|\mathcal{S}|-1}{|\mathcal{X}|-1}$. Thus, the codeword redundancy of CTW is composed of model, parameter and coding redundancy and upper bounded by

$$\delta(\boldsymbol{x}^N) < \underbrace{\frac{|\mathcal{X}|\cdot|\mathcal{S}|-1}{|\mathcal{X}|-1}}_{\text{model}} + \underbrace{|\mathcal{S}|\gamma\left(\frac{N}{|\mathcal{S}|}\right)}_{\text{parameter}} + \underbrace{2}_{\text{coding}}. \tag{4.26}$$

The codeword redundancy is only dependent on the number of parameters $|\mathcal{S}|$ and the sequence length N. It approaches the Rissanen lower bound [Ris76] for large N

$$\lim_{N\to\infty} \delta(\boldsymbol{x}^N) = \frac{|\mathcal{X}|-1}{2} \operatorname{ld} N. \tag{4.27}$$

This is true for any tree source. The CTW can be considered to show minimum description length (MDL) [GR07] behaviour if one accepts that redundancy is needed to describe the source model and parameters.

4.2.3 Decomposition Context Tree Weighting

The CTW algorithm was originally derived for binary sources and performs particularly well for such sources. Application of CTW to multi-alphabets can be accomplished in two different ways. Either the generalized multi-alphabet CTW, as described in Section 4.2.2, can be used or the binary version is applied to the binary representation of the alphabet symbols. The sooner has been found to perform relatively poorly with American Standard Code for Information Interchange (ASCII) encoded texts. A slight improvement can be achieved using the so called Good-Turing estimator instead of KT [BEY06]. Using binary CTW requires several adjustments for good performance [TVW97]. Direct application of binary CTW to the binary representation of ASCII texts is not advisable, since the different bit positions of the binary representation of an ASCII symbol have different underlying statistics. Thus, in case of byte encoded ASCII symbols each byte is decomposed into 8 bits and a different CTW tree is used for every bit position in the byte. The decomposition reduces the total number of parameters, and thus the redundancy. Assuming that the depth of the context has been set to two preceding ASCII symbols, the first bit has the two preceding ASCII as context. The second bit has the two ASCII and the first bit as context, etc. The underlying tree source can only have leaves at ASCII borders. In order to decrease the model redundancy and prevent overfitting, the binary trees should only be weighted at the ASCII borders. The inner ASCII bit borders cannot be leaves of the underlying tree source, thus their weights are computed as $P_{\boldsymbol{c}}(n) = P_{0\boldsymbol{c}}(n)P_{1\boldsymbol{c}}(n)$.

The estimation redundancy depends on the number of parameters. However, after decomposition many of the parameters are 0 or 1 because of non-occurring ASCII symbols, e.g. certain bit positions remain zero all the time if only upper case letters are used. Unfortunately, the KT estimator shows the highest prediction error for such deterministic sequences. This can be dealt with by using the zero-redundancy estimator instead of KT

$$\hat{P}_{ZR}(\boldsymbol{x}^n) = \frac{1}{2}\hat{P}_{KT}(\boldsymbol{x}^n) + \frac{1}{4}\vartheta(f_0(\boldsymbol{x}^n) = 0) + \frac{1}{4}\vartheta(f_1(\boldsymbol{x}^n) = 0), \tag{4.28}$$

where $\vartheta(\cdot)$ is the indicator function with $\vartheta(\text{true}) = 1$ and $\vartheta(\text{false}) = 0$. This limits the redundancy for deterministic sequences (all zeros or all ones) to 2 bits and increases the upper bound for the parameter redundancy of non-deterministic sequences only by 1 bit compared to KT. The decomposition based CTW approach can be successfully applied to the compression of DNA datasets, as described in Chapter 5.

4.3 Universal Dictionary Based Coding

Another very popular class of universal compression schemes is universal dictionary based coding, originally introduced by Lempel and Ziv in [ZL77, ZL78]. Dictionary based coders are simple to implement and asymmetric in the sense that decompression is far less complex than compression, making them particularly interesting for data distribution. In addition, they are especially well suited for data containing variable length repeating patterns, like natural texts or binary computer programs. These properties have resulted in high popularity and widespread use of adaptive dictionary based coding. Their compression rate has been shown to asymptotically achieve the entropy rate of the source for any stationary ergodic source. The key idea of dictionary based coding is to parse the message that is to be encoded into subsequences and to replace repeating subsequences by pointers to already encoded past occurrences.

In the two seminal papers [ZL77, ZL78], Lempel and Ziv (LZ) have proposed two distinct versions of the first dictionary based compression algorithm: a sliding window and a tree-structure based version referred to as LZ77 and LZ78 respectively. Meanwhile, dictionary based coding has been extended and refined in many aspects. However, the basic concept remained unchanged and shall be discussed in more detail in the following as it builds the basis for the most successful DNA sequence compression algorithms described in Section 5.1.1.

4.3.1 Sliding Window Lempel Ziv - LZ77

The algorithm described in [ZL77] uses a buffer of predefined depth D storing the past symbols $x_{n-D}x_{n-D+1}\dots x_{n-1}$. The longest match to the following symbol sequence $x_n x_{n+1}\dots$ is searched for in the buffer. The match is encoded by an offset pointer o to its location within the window, its length l and the symbol x_{n+l} that follows the match. Thus, $x_{n-o}\dots x_{n-o+l-1} = x_n \dots x_{n+l-1}$ and the triplet to be encoded is (o, l, x_{n+l}), whereas $0 \leq l \leq o \leq D$. If no match is found the triplet $(0, 0, x_n)$ is encoded. For the next iteration, n is set to $n = n + l + 1$. Huffman coding is used to compress the triplets.

Example: The sequence GATTTACCATTTACCA shall be compressed. Matching sequences in each encoding step are underlined. The respective x_n carries a hat, for example in the second step Â.

Step	G A T T T A C C A T T T A C C A	(o, l, x_{n+l})
1	Ĝ A T T T A C C A T T T A C C A	$(0, 0, \mathtt{G})$
2	G Â T T T A C C A T T T A C C A	$(0, 0, \mathtt{A})$
3	G A T̂ T T A C C A T T T A C C A	$(0, 0, \mathtt{T})$
4	G A T T̂ T A C C A T T T A C C A	$(1, 1, \mathtt{T})$
5	G A T T T Â C C A T T T A C C A	$(4, 1, \mathtt{C})$
6	G A T T T A C Ĉ A T T T A C C A	$(1, 1, \mathtt{A})$
7	G A T T T A C C A T̂ T T A C C A	$(7, 6, \mathtt{A})$

Huffman coding is used to encode the triplets whereas the pairs (o, l) and the symbols x_{n+l} are encoded using separate Huffman trees. ∗

The algorithm uses a dictionary that consists of all subsequences of the sliding buffer window. A slightly modified version introduced in [SS82] is used in most practical implementations today, e.g. *gzip, pkzip*. Increasingly popular state of the art LZ77 based algorithm that achieves good compression rates is Lempel-Ziv-Markov chain algorithm (LZMA). It uses a refined dictionary scheme and an adaptive coding scheme similar to arithmetic coding to encode the pointers. LZMA is used e.g. in the popular *7z* implementation.

4.3.2 Tree Structured Lempel Ziv - LZ78

The approach presented in [ZL78] parses the message into subsequences, where each subsequence is the shortest subsequence not seen earlier. A dictionary is build in form of a tree, where the paths from the root to each node represent subsequences observed so far. The path from the root to a leaf of the dictionary tree, matching the subsequence to be encoded next, represents the longest match. The matched subsequence plus the following symbol x represent the shortest new subsequence, unobserved so far. The path corresponding to the matched subsequence is prolonged using the symbol x. Therefore, one node is added to the dictionary tree in every step imposing an order on the nodes. Each new subsequence is encoded by the symbol x following the match and a pointer to the match in terms of an offset o in subsequences. Thus, the pair to be encoded is (o, x).

Example: The sequence GATTTACCATTTACCA would be parsed into the subsequences G,A,T,TT,AC,C,AT,TTA,CC,A and encoded using the pairs (0,G)(0,A)(0,T)(1,T)(3,C)(0,C)(5,T)(4,A)(3,C)(0,A). ∗

The most commonly used variant of LZ78 is the Lempel-Ziv-Welch algorithm (LZW) used by the Unix *compress* utility.

4.4 Summary

In this chapter, different lossless source coding schemes have been presented. The Huffman algorithm was introduced as the optimal symbolwise entropy coding scheme for memoryless sources with known statistics loosing at most 1 bit per symbol. Encoding blocks of symbols jointly reduced the loss to 1 bit per block, however the complexity was found to grow exponentially with increasing block length. Arithmetic coding was presented as the entropy coder of choice for longer messages generated by memoryless sources with known statistics. While it needs 1 bit extra per block compared to the optimal Huffman coding, the complexity grows only linearly with increasing block size. Subsequently, universal coding schemes capable of adapting to the source during the encoding process were presented. The universal compressors use either an adaptive statistical or a dictionary based prediction model to adjust to the source. The remaining redundancy after prediction is encoded using one of the entropy coding schemes. The presented representatives of both types of universal algorithms were found to asymptotically achieve the entropy rate for stationary ergodic sources. However, their practical performance differs for different kinds of sources due to the limited message length and computational complexity. While the statistical prediction algorithms like Context Tree Weighting perform particularly well for variable order Markov model type of sources, the Lempel-Ziv dictionary based compressors are particularly suited for sources producing repetitive subsequence patterns, e.g. written texts. Thus, the prediction based universal algorithms are universal in the sense of adapting to the actual source statistics, however they make certain a-priori assumptions about the general statistical model of the class of sources they have been designed for.

In Chapter 5, the presented universal prediction based source coding schemes will serve as basis for devising suitable universal compressors for voluminous sequence and alignment datasets generated and used in molecular biology.

5

Compression in Genetics

Due to recent progress in high-throughput DNA sequencing technology the amount of data available in public sequence databases e.g. GenBank at the National Center for Biotechnology Information (NCBI) [BKML+10], the University of California Santa Cruz (UCSC) Genome Browser [KBD+03] or the Ensembl database [HBB+02] is growing exponentially. Large sequence databases typically provide a limited web-interface and partial direct access to their database sufficient for the analysis of specific short genomic regions e.g. particular genes. However, genome-wide statistical analysis is only possible using a local copy of the data raising the problem of efficient storage and distribution.

Over the past few years the sequencing of many complex species, having large genomes, has been completed. In Section 5.1 state-of-the-art DNA sequence compression algorithms are discussed. However, the compressibility of genomic DNA sequence data seems to be limited. The availability of multiple genomes has given rise to the field of comparative genomics. In order to be able to compare whole genomes they have to be aligned first. The resulting whole genome alignments represent one of the largest sequence datasets in molecular biology. Genomes are subject to large scale mutations like translocations and duplications, and many small scale mutations including insertions, deletions and substitutions. Therefore whole genome alignments comprise many locally aligned blocks of regions from different genomes that share common ancestry, see Section 5.2. In Section 5.3 a highly efficient lossless compression scheme for such alignment blocks, relying on evolutionary models and techniques from lossless binary image compression is introduced. The whole genome alignments are distributed in form of multiple alignment format files containing the alignment blocks and position information about the aligned regions. In Section 5.4 the first compression algorithm for such multiple alignment format files is introduced. It is capable of reducing the file size tenfold and is two times more efficient than alternative universal compression algorithms.

5.1 DNA Sequence Compression

One representative of voluminous static datasets used in molecular biology are the reference DNA sequences of whole genomes. The human genome for example comprises approximately $3 \cdot 10^9$ nucleotides. A new human reference sequence is only released every few years (i.e. hg17 in 2004, hg18 in 2006 and the current version hg19 in 2009). Uncompressed DNA sequence data requires 2 bits/symbol due to the four nucleotides alphabet $\mathcal{A} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$. Single genomic DNA sequences seem largely incompressible using traditional compression algorithms as will be shown in Section 5.1.2. New compression schemes have been developed specifically for genomic DNA sequences, best of which up to now attain compression rates of around 1.8 bits/symbol [CDAM07]. A common property of these algorithms is that they use knowledge about genome evolution and composition for refining the DNA source model. Selected DNA compression algorithms are presented in Section 5.1.1. However, the improvement of roughly 10% against the uncompressed state has so far been insufficient for a broad adoption of such compression schemes for distribution and storage purposes. Nonetheless, DNA specific compressors can be successfully applied in phylogenetic classification [LBC+01] and to clustering of genomic data studied by the author in [HDG+04, DHHM05]. DNA compressors can also be used to study local DNA information content [DPA+07].

5.1.1 DNA Compression Algorithms

As detailed in Section 3.6.2, throughout evolution genomes are subject to mutations. Most frequent are substitutions, where single nucleotides transform into a different ones. Less frequent are duplications of longer sequence segments, where a duplicate is inserted at some other position in the genome either as an exact or palindromic copy (flipped reverse complement). Both types of replica (copy and palindromic copy) will be referred to as a repeat in the following. Genomic sequences thus contain point mutated repeats, which can be exploited to achieve compression. This regularity is particularly well captured by extended Lempel-Ziv like dictionary based compression schemes. Along the genome there exist regions where the bonding between the two strands is stronger and where it is weaker. Since the `G`≡`C` bond is stronger than an `A=T` bond such regions are going to have a high `GC` (low `AT`) and a low `GC` (high `AT`) content respectively. This regularity appears particularly well suited for statistical prediction based compression.

Palindromes

The first regularity exploited by compression algorithms specifically designed for DNA sequences were palindromes. Similar to Lempel-Ziv *Biocompress* [GT93] and *Biocompress 2* [GT94] are dictionary based. Exact copies and exact palindromic copies above a certain length are stored using an offset pointer and length. An extra flag-bit is encoded additionally per copy indicating whether it is palindromic or not. A large buffer is used in order to catch distant copies. The only difference between the two versions is in the way they store the remaining bases. While Biocompress leaves them uncompressed, Biocom-

press 2 uses adaptive arithmetic coding of order 2. A refined two-pass approach under the name *Cfact* was proposed in [RDDD96].

Example: The following sequence contains one longer palindromic repeat starting at position 31. It is encoded using the triplet $(offset, length, palindrome) = (21, 11, true)$.

$$\texttt{TGTAACCGAG}_{10}\texttt{ATTATTACCA}\texttt{GCTGGACATGT}_{31}\texttt{GGTAATAATC}\texttt{GTCGCGATA...} \qquad *$$

Approximate Repeats

Chen et al. [CKL01] were with *GenCompress* among the first to suggest approximate repeats instead of exact repeats. In this manner large scale duplications that have been modified by small scale mutations can be recognized and harnessed for compression as long as they have not diverged too much. In addition to encoding the offset pointer, the length and the palindrome-flag for the approximate repeat, also the number and the set of small scale edit operations necessary to describe the differences in the approximate repeat has to be encoded. Each edit operation is itself described by one of the three types insertion, deletion, substitution, by the position offset within the repeat and the base if necessary. An approximate repeat is only encoded as such if a compression gain is achieved.

Example: The following sequence contains one longer approximate palindromic repeat starting at position 31 containing one substitution. The approximate repeat is encoded using the quadruplet $(offset, length, palindrome, edits) = (21, 11, true, 1)$ and the substitution edit operation $(type, offset, base) = (substitution, 5, \texttt{C})$

$$\texttt{TGTAACCGAG}_{10}\texttt{ATTATTACCA}\texttt{GCTGGACATGT}_{31}\texttt{GGT}\hat{\texttt{C}}\texttt{ATAATC}\texttt{GTCGCGATA...} \qquad *$$

In [CLMT02] an improved version was introduced under the name *DNAcompress* using a two pass approach, finding significant approximate repeats in one pass and encoding these in another pass. The remaining bases are encoded using arithmetic coding of order-2.

Model Weighting

The currently best performing group of genomic DNA compression algorithms uses concepts from dictionary as well as statistical prediction based coding described in Section 4.2. The *DNA-eXpertModel* compressor [CDAM07] encodes a sequence symbol by symbol. A probability distribution is predicted for each symbol and encoded using arithmetic coding. The prediction is a weighted mixture of predictions from different models, called experts. Experts based on Markov type dependencies, local and global empirical symbol distributions, as well as dictionary based repeats are used. The weighting of the models is adapted according to the local prediction performance of each expert. The weights of experts performing locally well is increased. The positionwise self-information can be used to identify regions with low information content, e.g. repetitive sites [DPA+07].

5.1.2 Performance Comparison of DNA Compressors

The DNA sequence compression performance of different compression algorithms is typically tested on a standard benchmarking corpus comprising various sequence data including regular genes, chloroplast, mitochondria and bacteria genomes. Table 5.1 shows the compression rates achieved on the corpus by different compression algorithms. For comparison purposes Lempel-Ziv (LZ) and Context Tree Weighting (CTW) were chosen as representatives of universal dictionary and statistical prediction based source coders. DNAcompress (DNAc) and DNA-eXpertModel (DNAx) were picked as representatives of DNA specific compressors. While DNAc is an example of an extended dictionary based approach, DNAx is prediction driven and as of now the best DNA source coder. The individual sequences were obtained from the GenBank database [BKML+10]. The sequence size is provided in symbols, whereas a symbol corresponds to a base. The compression rate is in bits/symbol. Note, that the empirical symbolwise background distribution is roughly 2 bits/symbol for all sequences. Note that 2 bits/symbol would also be needed if the symbols were stored uncompressed due to the quaternary nucleotide alphabet.

sequence	length	LZ	CTW	DNAc	DNAx
Genes					
HUMDYSTROP	38,770	2.31	1.92	1.91	1.90
HUMGHCSA	66,495	1.52	1.88	1.03	1.00
HUMHPRTB	56,737	2.18	1.92	1.82	1.75
Chloroplast					
CHMPXX	121,024	2.15	1.82	1.67	1.66
CHNTXX	155,844	2.22	1.93	1.61	1.61
Mitochondria					
MPOMTCG	186,608	2.21	1.96	1.89	1.89
Bacteria					
H.Influenza	1,830,029	2.12	1.91	1.88	1.88
E.Coli	4,630,230	2.14	1.94	1.92	1.92

Table 5.1: Compression results obtained for the standard DNA compression benchmarking corpus - length is provided in symbols (bases), compression rate in bits/symbol.

Remarks on the DNA compression benchmarking corpus

- *Genes* are generally difficult to compress. HUMGHCSA is an exception as it contains a very long approximate repeat coding for two similar human growth hormones.
- *Chloroplast* DNA of higher plants contains a relatively long palindrome leading to good compression ratios for DNA specific compression algorithms.
- *Mitochondria* and *Bacteria* genomes consist primarily of coding sequences and are thus difficult to compress.

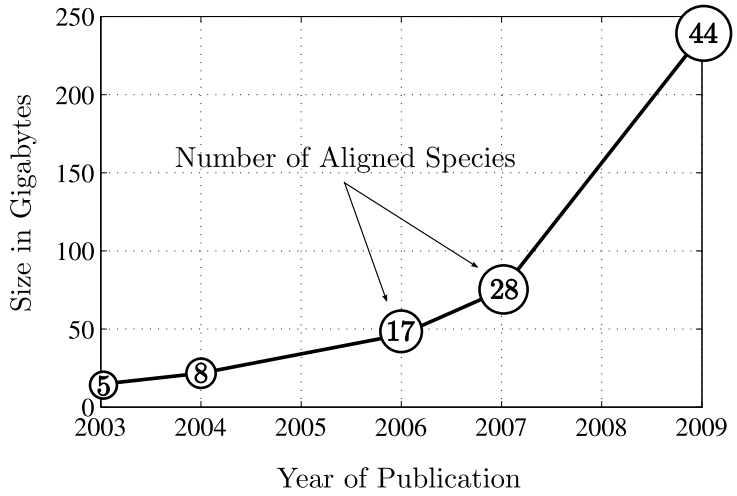


Figure 5.1: Growth in the storage requirement of multiple genome alignment datasets released by UCSC. Circles indicate number of species aligned.

- The corpus only contains small genomes, since the proposed DNA specific compressors are incapable of compressing large genomes for complexity reasons. This is due to the fact that the approximate repeat search is a non-deterministic polynomial-time hard (NP-hard) problem with respect to the sequence length.

Dictionary based LZ type compression algorithms assuming exact local repeats fail to compress genomic DNA data. They lack a strategy for distinguishing between repeats worth encoding and those, which are not. This leads to an expansion instead of compression as can be seen in Table 5.1. Statistical prediction based algorithms like *CTW* generally manage to slightly compress the DNA sequences, which can be partially attributed to the exploitation of the regionwise variation in the sequence `GC` content. However, pure statistical prediction based compressors are unable to profit from palindromic or distant approximate repetitions. The best compressors specifically designed for genomic DNA also use such dependencies. They achieve the best compression ratios for DNA sequence data at the cost of high complexity. However, the compression gain is limited. This has so far prevented a broad adoption of DNA specific compression algorithms for transmission and storage purposes.

5.2 Multiple Genome Alignment

A particularly voluminous static dataset in molecular genetics are whole genome alignments. They are essential for computational genome annotation as well as studies of evolution and variation. Genomes of different species are aligned using an extra symbol representing the missing entries. Both, the UCSC comparative genomics group and the Ensembl Compara project [CAB+03] provide multiple genome alignment datasets of vertebrate species. They are roughly hundred gigabytes large, rapidly growing in size with every release incorporating newly sequenced genomes, see Figure 5.1. In this work, the first compression algorithm for multiple genome alignment datasets is developed.

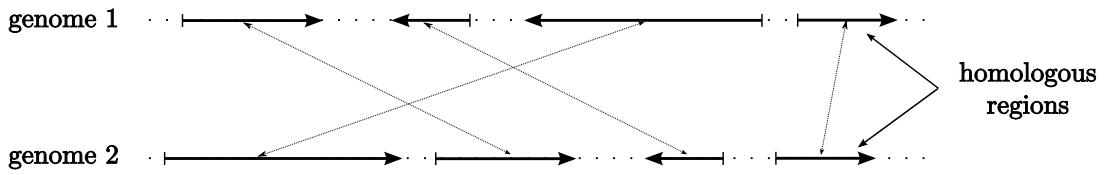


Figure 5.2: Paralog homologous regions are depicted. Their ordering and direction in different species can change during evolution due to large scale mutations causing genome rearrangements.

5.2.1 Motivation for Multiple Genome Alignments

Many areas of molecular biology research, in particular computational genome annotation and evolutionary genomics, have benefited from multiple genome alignment data generated by comparative genomics [DH04, Bla07]. Comparative analysis of homologous DNA sequences that share common ancestry helps to obtain a functional map of the human genome. It is the driving force behind the sequencing efforts of other vertebrate species. Homologous regions are likely to encode similar functions, and a function that was experimentally verified in one species can be mapped to homologous sequences in related species. Conservation studies on multiple genome alignments of vertebrates have revealed many highly conserved and thus likely functional regions outside of the protein coding and known functional DNA [DRS$^+$03]. The current knowledge about the human genome is quite limited. The role of most putative functional regions remains poorly understood. Comparing the human genome to that of other species and studying how different regions have evolved provides valuable additional knowledge simplifying the identification of the putative function even if the actual function is unknown in all the aligned species. For example, coding regions, RNA genes and regulatory sites are subject to different evolutionary constraints that leave distinguishable "evolutionary signatures". Thus, even conserved homologous regions with unknown function can be pre-assigned a putative functional category based on the observed evolutionary footprint [Bla07].

5.2.2 The Multiple Genome Alignment Problem

The challenge in constructing multiple genome alignments is to meaningfully compare the available sequenced genomes to each other by identifying homologous regions of common ancestry. However, the identification of homologous sequences is a non-trivial task since genomes of different species can greatly differ due to genomic mutations explained in Section 3.6.2. While small-scale mutations affect the DNA locally, altering a single or several nucleotides, large-scale mutations lead to a reorganization of the genome, see Figure 5.2. The further diverged the sequences are, the more difficult it is to obtain an evolutionary correct alignment. Once homologous sequences have diverged beyond a certain point, they cannot be reliably distinguished from non-homologous ones. Therefore, it is not possible to align whole genomes, but only the still identifiable homologous regions, resulting in a set of many locally aligned multiple sequence alignment (MSA) blocks.

In fact, multiple genome alignments are a mixture of global and local alignments. In a global alignment all the input sequences are aligned in a single alignment. Thus, global

alignments assume that the orthologous regions are co-linear and no rearrangements have taken place, which is unrealistic for whole genomes even for closely related species. A local alignment approach first determines homologous segments of the input sequences and only aligns those. This results in a mosaic of homology predictions in which the aligned segments are not necessarily co-linear, see Figure 5.2. However, by giving up the co-linearity assumption the risk of aligning regions that are not true homologs increases. Therefore, multiple genome alignment algorithms typically use a mixed approach. First, sets of pairwise homologous regions are identified for all species. Subsequently, the obtained pairwise sets are assembled into chains of co-linear sets. Short homologous pairwise regions, breaking locally the co-linearity in the assembly, are most likely false positives instead of true homologs and are thus removed from the dataset. The remaining homologous regions are aligned into MSA blocks using a progressive alignment scheme relying on pairwise alignments, see Section 5.2.4. The accuracy of the obtained MSA blocks largely depends on the used pairwise alignment procedure.

5.2.3 Pairwise Alignment - Smith Waterman Algorithm

Assume that two homologous sequences $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ are given. During evolution, both sequences experience small scale mutations, i.e. substitution and short insertion/deletion events. The latter result in sequences of unequal lengths, and it is necessary to reconstruct the homology of the nucleotides by aligning the sequences. A special symbol "`-`", called the "gap", is introduced to mark missing entries due to insertion/deletion events. Assume that the sequences have lengths $|\boldsymbol{s}_1|$ and $|\boldsymbol{s}_2|$, then a pairwise sequence alignment is a matrix $\boldsymbol{A} = \mathcal{A}^{2\times L}$ with $\mathcal{A} = \{\texttt{A,C,G,T,-}\}$ having the following properties:

1. $\max\{|\boldsymbol{s}_1|, |\boldsymbol{s}_2|\} \leq L \leq |\boldsymbol{s}_1| + |\boldsymbol{s}_2|$,
2. no column of $\boldsymbol{A}$ contains only gap symbols,
3. the ith row of $\boldsymbol{A}$ with gaps removed is identical to $\boldsymbol{s}_i$.

The Smith-Waterman algorithm (SW) [SW81] is capable of performing optimal pairwise alignment of two homologous sequences. It has to be provided with substitution and gap penalty scores $S(x, x')$ that fulfil

$$S(x,x') \begin{cases} \geq 0 & \text{for } x = x' \text{ no mutation} \\ \leq 0 & \text{for } x \neq x' \text{ mutation} \end{cases} . \tag{5.1}$$

Given this scoring system, the SW algorithm determines the optimal pairwise alignment using dynamic programming. A matrix $\boldsymbol{M}^{|\boldsymbol{s}_2|\times|\boldsymbol{s}_1|}$, where each row $1 \leq i \leq |\boldsymbol{s}_2|$ corresponds to the respective nucleotide in the sequence $\boldsymbol{s}_2$ and every column $1 \leq j \leq |\boldsymbol{s}_1|$ to the respective nucleotide in $\boldsymbol{s}_1$, is constructed for this purpose. Each element in the matrix represents the two residues of the sequences being aligned at that position, see (5.4). To calculate $M(x_i, x_j)$ one looks at the alignment that has been already made up to that point and finds the best way to continue. Thereby, a diagonal update corresponds

to the assumption of a match or substitution. A vertical update assumes a gap insertion in $\boldsymbol{s}_1$ and a horizontal update a gap insertion in $\boldsymbol{s}_2$. Assuming the initial values $M(x_0, x_0) = M(x_i, x_0) = M(x_0, x_j) = 0, \forall i, j$, the matrix M is constructed recursively according to the following rule

$$M(x_i, x_j) = \max \begin{pmatrix} 0 \\ M(x_{i-1}, x_{j-1}) + S(x_i, x_j) \\ M(x_{i-1}, x_j) + S(x_i, \texttt{-}) \\ M(x_i, x_{j-1}) + S(\texttt{-}, x_j) \end{pmatrix}, \forall i, j. \tag{5.2}$$

After constructing the entire matrix, one can go back starting in the lower right corner from $M(x_{|\boldsymbol{s}_2|}, x_{|\boldsymbol{s}_1|})$ and trace which way through the matrix back to $M(x_1, x_1)$ gives the best alignment by moving back in each step to that position among

$$\begin{array}{ll} M(x_{i-1}, x_{j-1}) & \text{diagonal step (homologous nucleotides)} \\ M(x_{i-1}, x_j) & \text{vertical step (insert gap in } \boldsymbol{s}_1) \\ M(x_i, x_{j-1}) & \text{horizontal step (insert gap in } \boldsymbol{s}_2) \end{array}, \tag{5.3}$$

that has the maximum score. The diagonal direction is preferred for equally weighted steps. An alignment constructed this way is optimal for the given scoring scheme S.

Example: Let the homologous sequences be

$$\boldsymbol{s}_1 = \texttt{ACACACT}$$
$$\boldsymbol{s}_2 = \texttt{AGCACACA}$$

and the penalty scoring scheme be equal to $S(x, \text{gap}) = S(\text{gap}, x) = S(\text{mismatch}) = -1$, $S(\text{match}) = 2$. Following the recursion in (5.2), the matrix $\boldsymbol{M}$ evaluates to

$$\mathbf{M} = \left(\begin{array}{c|ccccccc} & \texttt{A} & \texttt{C} & \texttt{A} & \texttt{C} & \texttt{A} & \texttt{C} & \texttt{T} \\ \hline \texttt{A} & \mathbf{2} & 1 & 2 & 1 & 2 & 1 & 0 \\ \texttt{G} & \mathbf{1} & 1 & 1 & 1 & 1 & 1 & 0 \\ \texttt{C} & 0 & \mathbf{3} & 2 & 3 & 2 & 3 & 2 \\ \texttt{A} & 2 & 2 & \mathbf{5} & 4 & 5 & 4 & 3 \\ \texttt{C} & 1 & 4 & 4 & \mathbf{7} & 6 & 7 & 6 \\ \texttt{A} & 2 & 3 & 6 & 6 & \mathbf{9} & 8 & 7 \\ \texttt{C} & 1 & 4 & 5 & 8 & 8 & \mathbf{11} & 10 \\ \texttt{A} & 2 & 3 & 6 & 7 & 10 & 10 & \mathbf{10} \end{array}\right). \tag{5.4}$$

The optimal path through the matrix is highlighted and corresponds to the alignment

$$\boldsymbol{A} = \begin{array}{l} \texttt{A-CACACT} \\ \texttt{AGCACACA} \end{array} \tag{5.5}$$

*

Although optimal for the alignment of two sequences, the Smith-Waterman algorithm is too slow for long sequences and it does not account for genome rearrangements. This can be achieved using heuristic Blast type algorithms [AGM+90] performing seeded alignments. In the first step, short exact matches are identified in the two genomes to be aligned. Subsequently, they are used as seeds to a Smith-Waterman like extension algorithm expanding the matching seed in both directions into aligned homologous regions.

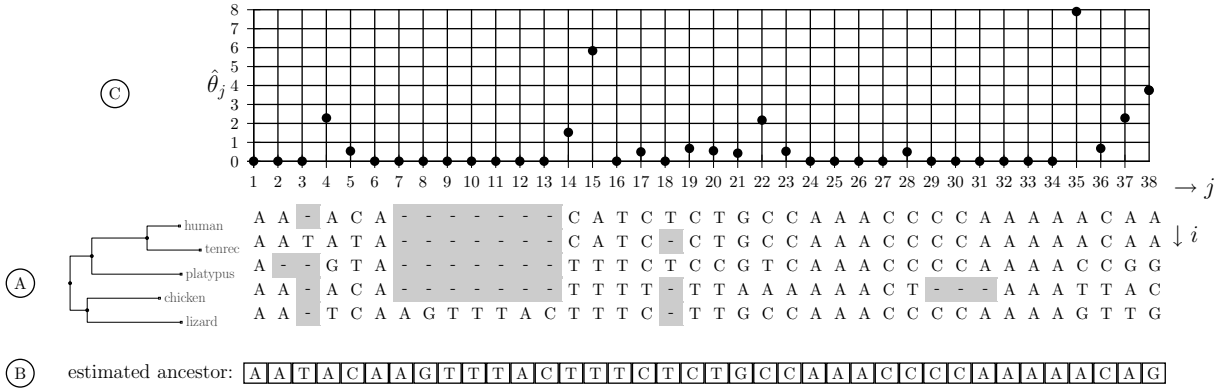


Figure 5.3: (A): a small alignment of 5 species with the corresponding phylogenetic tree. (B): the estimated maximum a posteriori common ancestor nucleotide for each column. (C): maximum likelihood estimate of the rate variation parameter $\hat{\theta}_j$ for each column, see Section 3.7.4.

5.2.4 Multiple Sequence Alignment (MSA)

There exist several possibilities for extending the pairwise alignment scheme to multiple homologous sequences. In the context of multiple genome alignments progressive alignment is used. It relies on the phylogenetic tree describing the evolutionary relationship between the species to be aligned [BKR$^+$04, BP04, BDC$^+$03]. The closest species are aligned first. Their parent node is assigned the resulting subalignment and the corresponding common ancestor sequence estimated using the Felsenstein algorithm, see Section 3.7.3. The ancestral sequence now represents the subalignment. This procedure is repeated until all sequences are aligned. Since the phylogenetic tree is binary, each step consists of the pairwise alignment of a subalignment (group of already aligned sequences), represented by the common ancestor, against another. The subalignment produced for a certain clade is not revised when combined with another one. This can result in suboptimal alignments, especially with respect to gaps. Therefore, post-processing algorithms are often used to fine-tune the position of gaps.

Example: An MSA block of homologous sequences from *human, tenrec, platypus, chicken, and lizard* together with the corresponding phylogenetic tree is shown in Figure 5.3 (A). Gaps in the alignment were caused by insertions/deletions, e.g. the gaps in positions $7-13$ were most likely caused by an insertion in the lizard, whereas in positions $29-31$ by a deletion in the chicken. It can be seen, that the gaps tend to appear in blocks and that the gap pattern shows strong regularities, see the background color of the MSA matrix. This property can be exploited for compression. The Felsenstein estimate of the common ancestor sequence of the depicted alignment is provided in Figure 5.3 (B). The common ancestor nucleotide has been estimated for each column. Note that the actual common ancestor species most likely did not contain the nucleotides 7-13. These have most likely been introduced by an insertion in the lizard branch. Note that Figure 5.3 (C) will be explained in Section 5.3.1. ∗

5.2.5 Whole Genome Alignment Datasets

Computing multiple genome alignments of vertebrates currently requires an enormous capacity of computational resources. Therefore, they are generally pre-computed by large annotation labs and centrally provided for public download by large databases like UCSC [KBD+03] and Ensembl [HAA+09]. Due to genome rearrangements caused by large-scale mutations, genome alignment datasets consist of many MSA blocks. In this work, the multiple genome alignment of 28 vertebrate species [MRH+07] (multiz28way) provided by UCSC serves as test dataset. It comprises alignment blocks containing from 2 up to 28 species. It uses human as reference species, meaning that a human homologous sequence is present in each MSA block. Note that a complete multiple genome alignment would also include MSA blocks unalignable with human. The MSA blocks are stored in several so called multiple alignment format (MAF) files, described in detail in Section 5.4.1. There exists one alignment file for each human chromosome comprising all MSA blocks containing homologs on that human chromosome. In the MAF file each MSA block carries supplementary information about the position of the aligned homologs in the respective genomes and the alignment score. The supplementary information makes up about 20% of the original file size, the remaining 80% are occupied by the MSA blocks. Thus, the overall compression efficiency for the whole genome alignment datasets is primarily dependent on how efficiently the MSA blocks can be compressed.

5.3 Multiple Sequence Alignment Compression

In the following a two step compression scheme for MSA blocks is proposed and analysed. Being from a 5 symbol alphabet, an uncompressed MSA requires $\mathrm{ld}(5) = 2.32$ bits/symbol. For the multiz28way reference dataset the herein proposed compression algorithm reduces the compression rate to approximately 1.0 bits/symbol.

The statistical dependencies of MSA blocks are basically governed by the two underlying and largely independent evolutionary processes, the substitution and the insertion/deletion mutational process, see Section 3.6.2. While there exist well established statistical models for the nucleotide substitution process described in detail in Section 3.7.1, modelling the insertion/deletion mutations is difficult. Therefore, in the proposed evolutionary based compression scheme the nucleotides in an MSA are compressed using the predictions obtained from the nucleotide substitution model, whereas the gaps are encoded independently using techniques from lossless binary image compression. Encoding the nucleotides and the gaps separately is justified by the independence of the two underlying mutational processes and should not introduce an inherent loss to the achievable compression rate. For complexity reasons a suboptimal algorithm has to be used to compress the nucleotide portion of the alignment blocks. However, the proposed suboptimal solution is shown to perform close to the optimum.

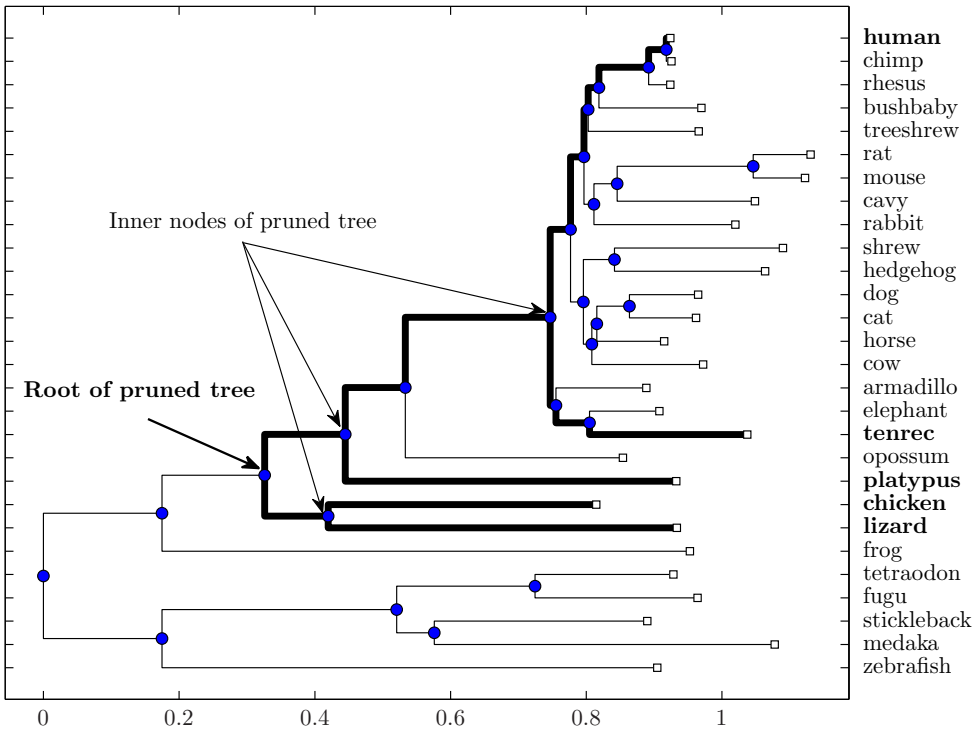


Figure 5.4: Phylogenetic tree depicting the species in the multiz28way dataset and their evolutionary relationships. The subtree corresponding to the species subset human, tenrec, platypus, chicken, lizard used in Figure 5.3 (A) is highlighted. The time (branch length) is measured in number of substitutions/site.

5.3.1 Compression of Nucleotides

As described in Section 3.7.4, the substitution process is a continuous time Markov process characterized by the evolutionary parameters $\{\boldsymbol{R}, \boldsymbol{\pi}, \mathcal{T}, \tau, \theta_j\}$ all of which are fixed except for the positionwise variable rate heterogeneity parameter θ_j, see Table 3.1. The fixed parameters only need to be encoded once for the whole multiple genome alignment dataset and are typically provided along with it. The rate heterogeneity θ_j needs to be encoded for each MSA column separately. For the multiz28way reference dataset the reversible substitution rate matrix equals to

$$\boldsymbol{R} = \begin{pmatrix} -0.991 & 0.179 & 0.491 & 0.321 \\ 0.257 & -1.002 & 0.187 & 0.558 \\ 0.706 & 0.187 & -1.162 & 0.269 \\ 0.321 & 0.388 & 0.187 & -0.896 \end{pmatrix}, \tag{5.6}$$

and the associated background distribution is $\boldsymbol{\pi} = (0.295, 0.205, 0.205, 0.295)$. The phylogenetic tree $\mathcal{T}$ and the corresponding branch lengths τ describing the evolutionary relationship between the species in the dataset are depicted in Figure 5.4.

Figure 5.3 (A) depicts one exemplary MSA block from the dataset and the corresponding pruned phylogenetic tree reduced to the species actually present in the alignment block. The pruned tree is easily derived from the full tree and corresponds to the highlighted

subtree in Figure 5.4. Note that it has a younger common ancestor node than the full tree. In fact, every alignment column is associated with a subtree of the full tree defined by the subset of species in which the homologous nucleotides have actually been found. In the extreme case of a single nucleotide present in an alignment column, see positions 7-13 in Figure 5.3 (A), the subtree is composed of a single root node and the nucleotide corresponds to the common ancestor. The maximum likelihood estimate of the rate heterogeneity parameter $\hat{\theta}_j$ is plotted in Figure 5.3 (C).

Columnwise and Nucleotidewise Approach

Two different encoding approaches for the nucleotide portion of MSAs are proposed in the following. Both rely on the well studied evolutionary substitution model presented in Section 3.7 that describes the evolutionary relationships between homologous nucleotides. The nucleotides in each MSA column are homologous in the sense of sharing a common ancestor. The substitution mutation process acts independently upon each alignment column. The evolutionary parameters characterizing the model are assumed to be given, but omitted from the notation where not needed. For each alignment column j the set of species leaf nodes ℓ corresponding to the homologous nucleotides actually observed in that column is determined first. Subsequently, the gaps are removed leading to the vector of homologous nucleotides $\boldsymbol{x}_\ell^j$ observed in the column. Note that the evolutionary relationship of the species leaf nodes ℓ is described by a subtree of the full phylogenetic tree $\mathcal{T}$.

In Section 3.7.3, the Felsenstein algorithm was introduced as an efficient way of computing the probability $p(\boldsymbol{x}_\ell^j)$ of observing a set of homologous nucleotides in different species given the evolutionary model relating the species. The obtained probabilities for each column can be losslessly encoded using arithmetic coding. This columnwise approach represents an optimal encoding strategy given the evolutionary model. However, it is only feasible for small numbers of nucleotides per column, since the arithmetic encoder used to encode the column probabilities requires the exact knowledge of the complete PMF for all possible column realizations. Their number is of the order $\mathcal{O}(|\mathcal{A}|^N)$, where $|\mathcal{A}|$ is the alphabet cardinality and N the number of nucleotides in the column.

Therefore, an alternative nucleotidewise encoding scheme is proposed. A representative common ancestor nucleotide $\hat{x}_\mathfrak{r}^j$ is encoded for each column together with the set of conditional probabilities $p(x_{\ell_i}^j|\hat{x}_\mathfrak{r}^j), \forall i = 1 \dots N$ of all the leaf nucleotides observed in that column. The representative common ancestor $\hat{x}_\mathfrak{r}^j$ is a function of the column realization $\boldsymbol{x}_\ell^j$ and is chosen in a way minimizing the amount of bits required to encode the column j

$$\hat{x}_\mathfrak{r}^j = \arg\max_{x_\mathfrak{r}} \left(\prod_{i=1}^{N} p(x_{\ell_i}^j|x_\mathfrak{r}) \right). \tag{5.7}$$

The computation neglects the dependencies of the nucleotides in the column introduced by the topology of the phylogenetic tree characterizing the SIMO channel, see Figure 5.5. The distances between the root and the leaves are the same in both schemes. The estimated common ancestor nucleotide $\hat{x}_\mathfrak{r}^j$ corresponds to the maximum likelihood estimate

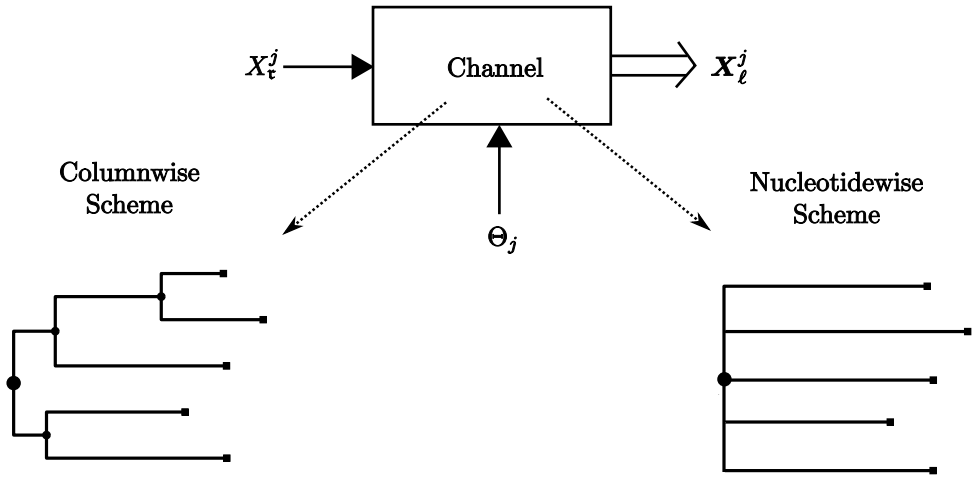


Figure 5.5: SIMO channel models used by the proposed compression schemes. In contrast to the columnwise scheme, the nucleotidewise scheme neglects the topology of the phylogenetic tree.

under the assumption that the nucleotides have evolved independently. It differs from the maximum likelihood estimate obtained with the Felsenstein algorithm (3.17) that accounts for the topology of the tree. In fact, in many SIMO transmission channel models used in communications engineering this is the preferred model, since typically the "scattering topology" of the SIMO channel is unknown or difficult to estimate. For compression purposes the objective is not to reconstruct evolution as accurately as possible, but to minimize the overall compression rate. The brute force search for the representative common ancestor nucleotide $\hat{x}_t^j$ that minimizes the compression rate of a column is easily performed and is actually faster than the computation of the evolutionary correct estimate using the Felsenstein algorithm. At the decompressor $\hat{x}_t^j$ is known. Thus, the nucleotidewise compression scheme is asymmetric. The encoding process is more complex than the decoding. This nicely fits the static multiple genome alignment datasets. The encoding process itself can be computationally demanding and should be optimized in terms of compression efficiency, as it is only performed once by the dataset provider. Therefore, $\hat{x}_t^j$ can be chosen using the brute force search such that it minimizes the compression rate and not by simply selecting the most frequent nucleotide in the alignment column $\boldsymbol{x}_\ell^j$. Decoding is performed by each user of the data separately on a standard workstation. Thus, it should be fast and simple.

Note that due to gaps the number and subset of species actually present in a particular MSA column j is varying, see Figure 5.3. The average compression rate achievable for a particular subset depends on the distance to the common ancestor of all the species in the subset, the number of species and the rate heterogeneity parameter θ. For a small number of species in the subset, the achievable average compression rates can be computed analytically for both compression schemes. For the columnwise compression the average compression rate

$$R_c = \frac{1}{N} H(\boldsymbol{X}_\ell). \tag{5.8}$$

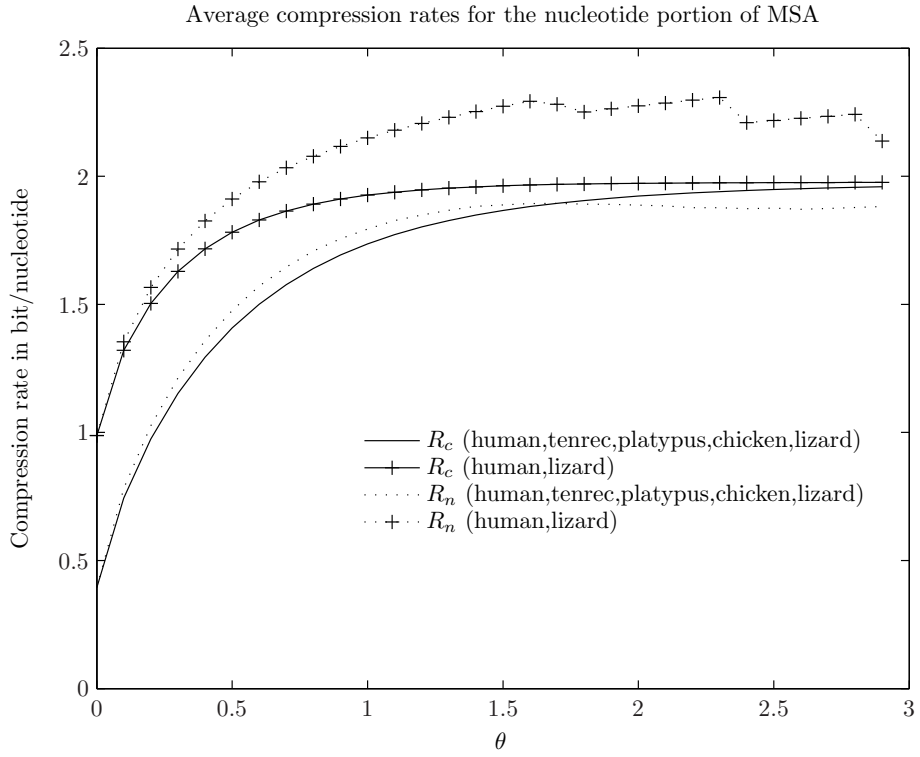


Figure 5.6: Average achievable compression rates per nucleotide for the columnwise $R_c(\theta)$ (full lines) and nucleotidewise $R_n(\theta)$ (dotted lines) compression scheme for a 5 and a 2 species dataset having identical common ancestors.

For the nucleotidewise compression scheme the average compression rate

$$R_n = \frac{1}{N}\left(\sum_{\forall \boldsymbol{x}_\ell} p(\boldsymbol{x}_\ell) \log \frac{1}{\prod_{i=1}^{N} p(x_{\ell_i}|\hat{x}_\mathfrak{r})}\right) + H(\hat{X}_\mathfrak{r}), \tag{5.9}$$

where $\hat{x}_\mathfrak{r}$ is the representative common ancestor computed according to (5.7). Since the alignment columns are assumed to be independent except for the rate heterogeneity, the average amount of bits needed for the encoding of $\hat{x}_\mathfrak{r}$ is $H(\hat{X}_\mathfrak{r})$. For a small number of species the PMF of $\hat{x}_\mathfrak{r}$ is computed as

$$p(\hat{x}_\mathfrak{r}) = \sum_{\forall \boldsymbol{x}_\ell : f(\boldsymbol{x}_\ell)=\hat{x}_\mathfrak{r}} p(\boldsymbol{x}_\ell), \tag{5.10}$$

where $f(\boldsymbol{x}_\ell)$ is the estimator from (5.7). When compressing real MSA data, the representative common ancestor nucleotides can be compressed using some universal compression scheme, e.g. the Context Tree Weighting algorithm, presented in Section 4.2.2.

Figure 5.6 depicts R_c and R_n as a function of the rate heterogeneity θ for two differently large species subsets. Both subsets have the same common ancestor. The corresponding phylogenetic tree can be found in Figure 5.3. It can be seen that for small values of θ the proposed compression schemes can dramatically reduce the compression rates for datasets comprising more species, compare the 2 and the 5 species dataset. As expected, the columnwise compression (full lines) saturates at 2 bits/symbol for high values of θ. The

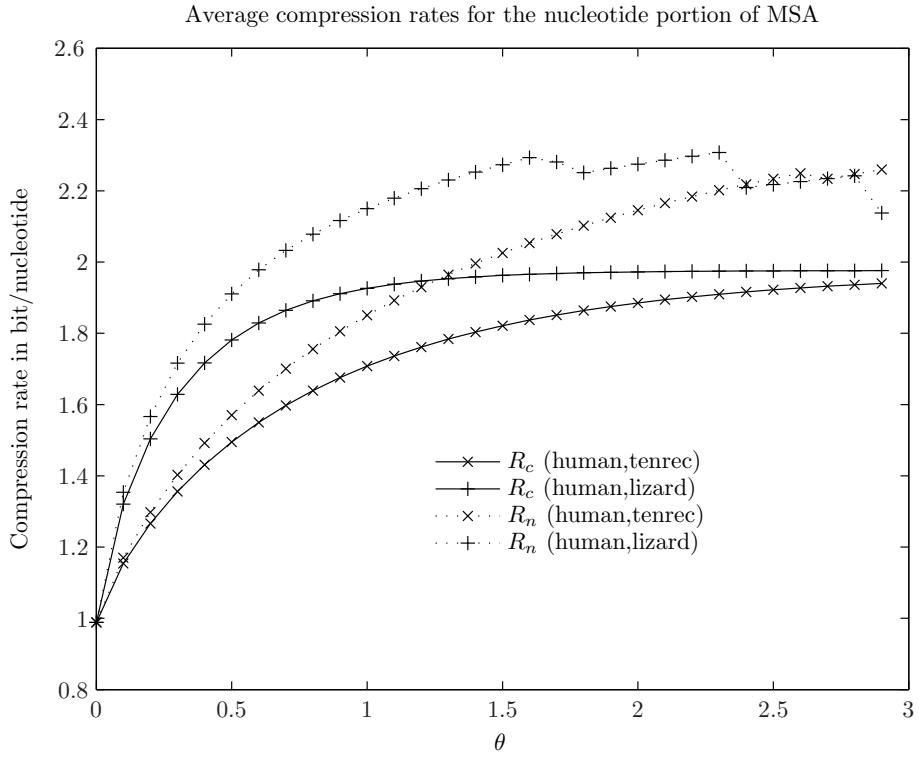


Figure 5.7: Average achievable compression rates per nucleotide for the columnwise $R_c(\theta)$ (full lines) and nucleotidewise $R_n(\theta)$ (dotted lines) compression scheme for 2 species datasets having different common ancestors.

nucleotidewise compression (dotted lines) suffers an inherent loss because the nucleotides in a column are encoded separately as if they have evolved independently instead of on a tree. This effect is particularly well observable for low θ values. Encoding the representative common ancestor nucleotides separately introduces additional loss. The impact is especially strong for a small number of species, see the 2 species dataset. However, for a larger number of species and higher values of θ the nucleotidewise approach benefits from being able to choose the common ancestor nucleotide such that the number of bits needed for encoding the column is minimized. It can even outperform the columnwise approach in this case, see the 5 species dataset. Figure 5.7 studies R_c and R_n for two equally large species subsets having a different common ancestor. The *human, tenrec* subset has a significantly younger common ancestor than the *human, lizard* subset as can be seen from the phylogenetic tree in Figure 5.3. The closer is the common ancestor, the slower is the degradation of the compression rate with increasing θ.

The presented results promise substantial compression gains for a high degree of conservation (low values of θ) and many species in the dataset. Homologous sequences are alignable only if they have not diverged to much. Thus, low average values of θ can be expected for columns of MSA blocks in multiple genome alignment datasets. The proposed evolutionary compression scheme also nicely fits the fact that the number of species in the datasets is rapidly increasing, see Figure 5.1. More species lead to more homologous nucleotides per column and thus to a decreasing average compression rate per nucleotide.

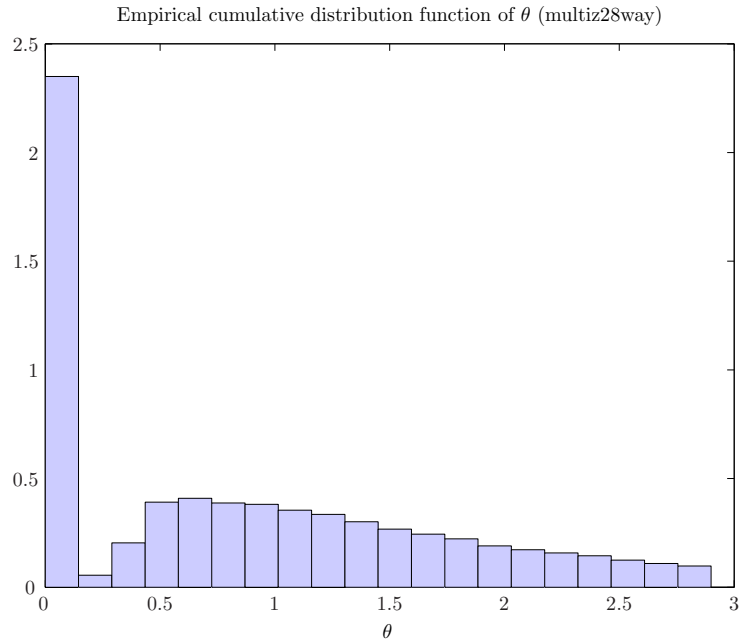


Figure 5.8: The empirical cumulative distribution function of θ values in the multiz28way dataset estimated using maximum likelihood.

Compression of the Rate Heterogeneity Parameter

The empirical cumulative distribution function of the estimated θ values for the multiz28way dataset is plotted in Figure 5.8. The values were estimated using a maximum likelihood estimator [DHL+08]. Based on the prevailing low values of θ, it can be concluded that the use of the evolutionary substitution model for compression promises high compression gains for the nucleotide portion of the MSA blocks. However, the rate heterogeneity parameter θ_j needs to be stored for each column j. In order to be able to efficiently compress the θ values they have to be quantized. By using quantized values of θ the evolutionary prediction used to losslessly compress the nucleotides becomes coarser. The less quantization levels are used, the better compressible are the θ values, but the more bits are needed to compress the nucleotides. The objective function of the quantization is the minimization of the overall compression rate comprising the compression of the θ values and the nucleotides and not the accurate reconstruction of θ. Therefore, instead of the Lloyd-Max quantizer, non-linear programming optimization [GMW81] has to be used to determine the optimal quantized values of θ for different numbers of quantization levels. Table 5.2 shows the overall compression rate per nucleotide for MSA blocks in the chromosome 21 (chr21) MAF file using different numbers of optimized quantization levels. The factual compression rate achieved was used as optimization function. As described in Section 3.7.4 strong Markov type correlations exist between neighbouring sites with respect to rate heterogeneity. Therefore the concatenated sequence of quantized θ values is compressed jointly using the Context Tree Weighting compression algorithm detailed in Section 4.2.2.

Note that the compression rate per nucleotide differs from the compression rate per MSA symbol, since only a portion of an MSA are nucleotides. Surprisingly, using a single

Optimal Quant. Levels	Comp. Rate
0.77	**1.16**
0.24, 1.98	1.15

Table 5.2: Minimum achievable compression rates for the nucleotide portion of MSA blocks in bits/nucleotide (multiz28way-chr21). The compression rates account for both the encoding of the quantized rate heterogeneities and the nucleotides.

quantization level for θ turns out to be sufficient for the considered multiz28way dataset. Obviously, the θ sequence is deterministic in this case and does not have to be stored. Increasing the number of quantization steps further only marginally improves the compression rate. It can be concluded that for the purpose of compressing the multiz28way dataset the space time process of rate variation can be omitted from the computation by scaling the tree branch lengths τ in the evolutionary model by the factor 0.77.

Algorithm Summary and Implementation Aspects

Algorithm 5.1 summarizes the final nucleotide compression algorithm. It relies on the nucleotidewise compression scheme. Note that in an actual implementation the algorithm can be considerably speeded up by pre-computing the channel matrices $\boldsymbol{P}_{X_{\ell_i}|X_\mathfrak{r}}$ for the set of all possible subtree leaves ℓ_i which equals to the leaves of the full phylogenetic tree $\mathcal{T}$ and for the set of all possible root nodes $\mathfrak{r}$ which equals to all the non-leave nodes of $\mathcal{T}$. The fact that column realizations $\boldsymbol{x}_\ell^j$ sometimes repeat in an MSA block can be used to further increase the encoding speed.

Algorithm 5.1 Compress nucleotides

Require: alignment block $\boldsymbol{A}$, evolutionary model parameters $\{\boldsymbol{R}, \mathcal{T}, \tau\}$

for all alignment columns **do**
  Determine set of species leaf nodes ℓ corresponding to nucleotides in column
  Encode ℓ
  Compute $\boldsymbol{x}_\ell$ (remove gaps from column)
  Set $\mathfrak{r}$ to the common ancestor node of the subtree of $\mathcal{T}$ corresponding to ℓ
  for all leaf nodes ℓ_i **do**
    Compute channel matrix $\boldsymbol{P}_{X_{\ell_i}|X_\mathfrak{r}} = e^{\boldsymbol{R} \cdot t_{\mathfrak{r} \to \ell_i}}$
  end for
  Estimate common ancestor nucleotide $\hat{x}_\mathfrak{r} = \arg\max_{x_\mathfrak{r}} \left(\prod_{i=1}^{N} p(x_{\ell_i}|x_\mathfrak{r}) \right)$
  Encode $\hat{x}_\mathfrak{r}$
  for all species leaf nodes ℓ_i **do**
    Encode $p(x_{\ell_i}|\hat{x}_\mathfrak{r})$
  end for
end for

5.3.2 Compression of Gaps

In Figure 5.3 (A) the background color of the symbols (white for nucleotides and grey for gaps) represents the binary image of size of the original MSA that has to be compressed in order to encode the positions of gaps. It turns out that a template driven predictor is capable of thoroughly describing the dependencies in the puncturing pattern, allowing for its efficient compression. The predicted probabilities are encoded using arithmetic entropy coding. While, at the encoder it is unimportant, whether the nucleotides or gaps are encoded first, at the decoder the puncturing pattern has to be recovered before decompressing the nucleotides.

Small scale evolutionary insertion and deletion events typically involve multiple neighbouring nucleotides. During the course of evolution, events that have occurred in a common ancestor propagate to the derived subspecies resulting in rectangular blocks of gap symbols in MSA. Thus, the binary MSA gap puncturing matrix shows strong horizontal and vertical correlations that can be used for compression. However, these regularities have different nature from those found in x-ray images or facsimile data. State-of-the-art lossless compression algorithms for these types of binary image data, like Joint Bi-level Image Experts Group (JBIG) [HAC+92] have been found to have difficulties in delivering satisfactory compression rates for gap puncturing matrices. For this reason a template driven prediction based compression algorithm is proposed to compress the puncturing matrices.

Puncturing Matrix Compression

In case of strong local correlations in image data, the neighbouring pixels carry a lot of information about the pixel of interest and can thus be used to reliably predict its value. If the purpose of the prediction is compression, only pixels that have already been encoded can be used for the prediction context C. In case of encoding the pixels row by row from left to right, only pixels in the rows above and left of the encoded pixel x_0 in the current row can be used as prediction context $\boldsymbol{x}_c = (x_{-1}, x_{-2}, \ldots, x_{-D})$ where D is the size of the context in pixels. The conditional prediction PMF $P_{X_0|\boldsymbol{X}_C}$ can be trained on sample puncturing matrices or learned adaptively as described later. The conditional probability $p(x_0|\boldsymbol{x}_c)$ is computed for each pixel in the puncturing matrix and encoded using arithmetic coding. The empirical conditional entropy $H(X_0|\boldsymbol{X}_c)$ derived from all MSA gap puncturing matrices in the chr21 MAF file can be used as an estimate for the expected compression rate. Permuting different contexts of increasing size it was found that the prediction context depicted in Figure 5.9 presents the best context of size 4 and that increasing the context size D any further improves the expected compression rate $H(X_0|\boldsymbol{X}_c) \approx 0.17$ bits/symbol only marginally and thus represents a suitable trade-off between complexity and compression gain. In order to be able to use this context the first column and the first two rows have to be encoded separately. The contexts used to encode these edges are also depicted in Figure 5.9. The pixel in the upper left edge of each MSA is saved uncoded. The procedure is summarized in Algorithm 5.2. Note that the separation between the main and the edge contexts has been omitted for simplification.

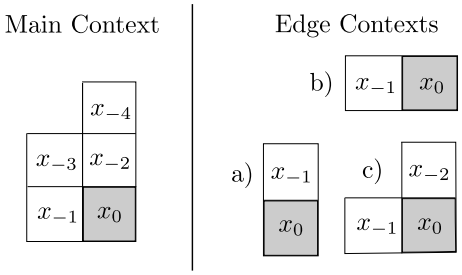


Figure 5.9: Left: Best Main Context of depth $D = 4$, Right: Edge Contexts a) first column, b) first row, c) second row.

Algorithm 5.2 Compress gap positions

Require: alignment block $\boldsymbol{A}$, prediction context C, prediction PMF $P_{X_0|\boldsymbol{X}_C}$

1: Encode the size of $\boldsymbol{A}$
2: Convert $\boldsymbol{A}$ to binary gap puncturing matrix
3: **for all** pixels **do**
4: Set x_0 to the current pixel
5: Set $\boldsymbol{x}_C$ to the neighbouring pixels according to the prediction context C
6: Encode $p(x_0|\boldsymbol{x}_C)$
7: **end for**

In order to make the puncturing matrix compression algorithm universal the conditional prediction PMF $P_{X_0|\boldsymbol{X}_C}$ has to be learned in course of the compression. Using a Context Tree Weighting model [WST95] seems particularly suitable for this task, since the CTW algorithm shows excellent convergence behaviour in terms of learning quickly the statistics of finite size context sources from finite length training data, see Section 4.2.2. The context tree structure implies an order of the context symbols. However, the pixels in the prediction context $\boldsymbol{x}_c$ are not inherently ordered. For best results the order should be chosen such that $H(X_0|X_i) > H(X_0|X_{i-1}), \forall i$. The indexing of the prediction contexts in Figure 5.9 already reflects this order.

Comparison to other Lossless Compressors

The proposed puncturing matrix compressor (PMc) has been compared to other state-of-the-art approaches for lossless compression including JBIG and JBIG2 [HAC+92] (specifically developed for source coding of medical images and fax data) and LZ (traditionally used for file compression on PCs). The obtained compression rates for the puncturing matrices of each MSA block in the chr21 MAF file were used to plot an approximate function of the compression rate as a function of the MSA block size in symbols. The result is shown in Figure 5.10. Also depicted is a histogram of the block lengths. It can be seen that for all algorithms the compression rate improves with increasing size of the puncturing matrix. This is especially the case for the JBIG compressor. Its bad compression rate for small and medium size puncturing matrices implies wrong initial assumptions about image statistics.

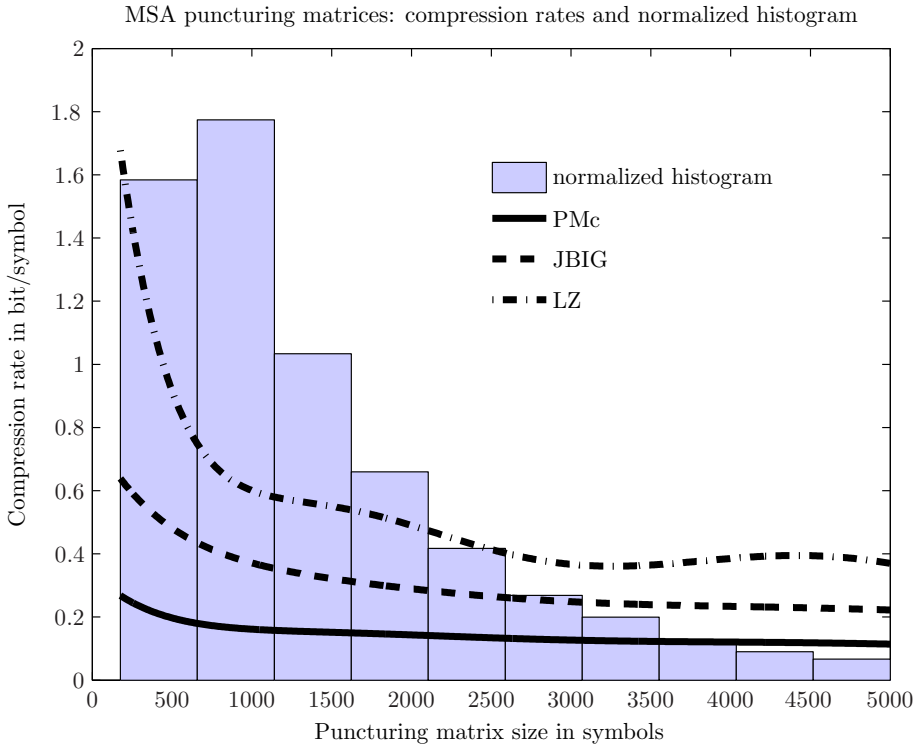


Figure 5.10: Compression rates for puncturing matrices of MSA blocks (chromosome 21 from the multiz28way dataset).

Table 5.3 shows the average compression rates achieved for all MSA puncturing matrices of chr21. The proposed PMc algorithm clearly outperforms its competitors.

PMc	JBIG	JBIG 2	LZ
0.17	0.30	0.48	0.53

Table 5.3: Average compression rate for gap puncturing matrices in bits/symbol (mulitz28way-chr21).

5.3.3 Multiple Sequence Alignment Compressor (MSAc)

Finally, the complete multiple sequence alignment compressor (MSAc) shall be described and its performance evaluated. MSAc compresses the alignment files MSA block by block. The nucleotides are compressed using the nucleotidewise compression scheme MSAc based on the probabilistic evolutionary model for substitutions described in Algorithm 5.1. The gap positions are compressed using the compression scheme presented in Algorithm 5.2 relying on image compression techniques. A two-dimensional (2-D) context driven predictor is used to compress the binary gap puncturing matrix. The static evolutionary parameters comprised of the rate matrix $\boldsymbol{R}$, the full phylogenetic tree $\mathcal{T}$ and the corresponding branch lengths τ are compressed once for the whole dataset.

Algorithm 5.3 Multiple genome sequence alignment compressor (MSAc)

Require: set of alignment blocks $\{\boldsymbol{A}_1 \dots \boldsymbol{A}_N\}$, prediction context C, evolutionary model parameters $\{\boldsymbol{R}, \mathcal{T}, \tau\}$

1: Encode $\{\boldsymbol{R}, \mathcal{T}, \tau\}$
2: Train the prediction PMF $P_{X_0|\boldsymbol{x}_C}$ (required by gap positions compression)
3: Encode PMF $P_{X_0|\boldsymbol{x}_C}$
4: **for all** alignment blocks $\boldsymbol{A}_n$ **do**
5: Compress nucleotides in $\boldsymbol{A}_n$ (Algorithm 5.1)
6: Compress gap positions in $\boldsymbol{A}_n$ (Algorithm 5.2)
7: **end for**

5.3.4 Performance and Comparison

Table 5.4 shows the overall achievable compression rates for different alignment files in the multiz28way dataset. For comparison purposes also the compression rate for the optimal columnwise compressor MSAc (col) and LZ is provided. The standard MSAc based on the nucleotidewise scheme has been abbreviated as MSAc (nuc) in Table 5.4. Note that the columnwise encoder MSAc (col) is not feasible for datasets containing more than a few species since the complete PMF of all possible column realizations (of the order $\mathcal{O}(|\mathcal{A}|^{28})$ for the multiz28way dataset), required by the arithmetic encoder, cannot be computed. Nonetheless, the compression rate that would be achieved using the columnwise compression scheme can be estimated since the column probabilities $p(\boldsymbol{x}_\ell^j), \forall j$ are computable. For MSAc the overall achievable compression rates are calculated as follows. For example, the MSA blocks in chr21 MAF file contain on average 74.7% nucleotides, the remaining symbols are gaps. The nucleotides are compressed at a rate of R_n=1.16 bits/nucleotide, see Table 5.2. The compression rate for gap positions per MSA symbol is 0.17 bits/symbol, see Table 5.3. Thus, the overall compression rate for chr21 is

$$1.04 \text{ bits/symbol} = 74.7\% \text{ nucleotide/symbol} \times 1.16 \text{ bits/nucleotide} + 0.17 \text{ bits/symbol}.$$

The currently used LZ source coding is outperformed by a factor of approximately 1.6. Note that the gzip compressor with the best compression setting was applied to a row by row concatenation of all MSA blocks in the chr21 MAF file in order to compute the compression rates for LZ. The results for the regular chromosomes chr01-chr22 making up almost all of the dataset are very similar. The corresponding alignment files all contain around 74% nucleotides. The compression rates for the sex chromosome chrY are higher, since it has a higher nucleotide portion of 89%. This is due to the fact that it is under stronger evolutionary pressure, since it only occurs in males in a single copy. The other sex chromosome chrX behaves similarly to regular chromosomes. The proposed nucleotidewise compressor MSAc (nuc) is only 5% worse than the infeasible but optimal columnwise compressor MSAc (col). With respect to complexity MSAc operates asymmetrically with significantly simpler decompression. The decompression can be performed in reasonable time on an ordinary workstation. With newly included species into the whole genome alignment datasets the compression rate of MSAc can be expected to further improve.

Data	MSAc		LZ	size in %
	(nuc)	(col)		
chr1	1.01	0.94	1.62	8.25%
chr2	1.00	0.94	1.63	8.84%
chr3	1.00	0.94	1.62	7.38%
chr4	1.01	0.96	1.64	6.38%
chr5	1.00	0.94	1.63	6.46%
chr6	1.00	0.95	1.63	6.10%
chr7	1.01	0.96	1.63	5.31%
chr8	1.02	0.96	1.64	4.99%
chr9	1.01	0.95	1.62	4.21%
chr10	1.02	0.96	1.64	4.76%
chr11	1.02	0.96	1.63	4.75%
chr12	1.00	0.94	1.62	4.60%
chr13	1.01	0.96	1.64	3.38%
chr14	1.00	0.94	1.62	3.24%
chr15	1.01	0.95	1.62	2.90%
chr16	1.03	0.97	1.63	2.77%
chr17	1.02	0.95	1.60	2.83%
chr18	1.01	0.96	1.64	2.72%
chr19	1.08	1.02	1.60	1.40%
chr20	1.03	0.96	1.64	2.18%
chr21	1.04	0.99	1.66	1.10%
chr22	1.07	1.01	1.64	1.07%
chrX	0.99	0.93	1.60	4.12%
chrY	1.19	1.18	1.68	0.26%
multiz28way	**1.01**	0.96	**1.63**	**100.00**%

Table 5.4: Overall compression rate for MSA in bits/symbol using different algorithms (uncompressed 2.32 bits/symbol required). The proposed nucleotidewise scheme MSAc (nuc) performs only 5% worse than the optimal but infeasible columnwise scheme (col). Compared to LZ, the compression rate is improved by a factor of 1.6.

A great advantage of MSAc compared to LZ is that with small adjustments it allows to decompress any single MSA block without having to decompress the entire dataset. This can be achieved by replacing the universal adaptive CTW prediction algorithm in the compression of the common ancestor and of the puncturing matrix by their static counterparts using the empirical distribution derived from the dataset for prediction. This adjustment increases the encoder complexity since it requires a two pass approach. The empirical distributions are collected in the first pass and subsequently used for compression in the second pass. Even though the empirical distributions now have to be encoded separately, the compression rate remains effectively unchanged. The complexity at the decoder is decreased, since it knows the prediction statistics a-priori and does not have to

learn it adaptively from already decoded data. Knowing the prediction statistics a-priori allows to decompress any single MSA block without the needed of decompressing the preceding blocks. The shift in complexity towards the encoder fits the envisaged distribution scenario. Additionally, the ability to decode any MSA block separately opens MSAc completely new application possibilities. LZ can only reasonably compress the dataset if compressing all MSA blocks jointly and is thus primarily suited for the distribution of the entire dataset at once. However, MSAc could also be used to decrease the storage requirement of database servers. Using MSAc, the dataset can be stored compressed on the server and the queried MSA blocks decompressed on the fly.

5.4 Multiple Alignment Format Files Compression

Currently, the whole genome alignment datasets provided by UCSC are distributed in form of multiple alignment format files. The MAF files can be downloaded LZ compressed from the UCSC genome browser website [KBD+03]. There exists one MAF file for each human chromosome comprising all MSA blocks containing homologs on that human chromosome. In the MAF file each MSA block carries supplementary information about the position of the aligned sequences in the reference genomes, see Table 5.5. This supplementary information makes up about 20% of the original file size, the remaining 80% are occupied by the MSA blocks. With MSAc, proposed in Section 5.3, an efficient compression algorithm for the MSA blocks has been derived. In the following possibilities of efficiently compressing the supplementary information shall be explored. An efficient multiple alignment format file compressor (MAFc) using MSAc to encode the MSA blocks is proposed and its overall performance evaluated.

5.4.1 Multiple Alignment Format (MAF)

Each MSA block is stored in a MAF file using the format shown in Table 5.5. The first line in each block is designated for the alignment score (**score**) returned by the alignment algorithm. Each row in the alignment contains position information about the aligned homologous region. The position information refers to the placement in the reference genomes used to construct the alignment. It comprises the species and chromosome (**spec.chr**), the start position in the chromosome (**start**), the length of the respective homolog in nucleotides (|**seq**|) and the DNA strand (**strand**). Additionally, the respective chromosome length is provided (|**spec.chr**|). Note that the name of the reference genome assembly is used to denote the species, e.g. "hg18" refers to the human genome assembly released by the UCSC genome browser in 2006 [KKZ+06]. Accordingly, the chromosomes correspond to the respective assembly. The reference strand is denoted by a "+" symbol.

5.4.2 Compression of MAF Supplementary Data

In order to compress the supplementary information it is important to understand its statistical regularities with respect to the whole dataset. In general, the different types

	spec.chr	start	|seq|	strand	|spec.chr|	MSA
			⋮			
a	**score**=34150.000000					
s	hg18.chr21	9882599	133	+	46944323	GAGGCTTTCG...
s	eriEur1.scaffold_266948	48	125	-	8534	GAAG--TCCG...
s	panTro2.chr15	81691913	133	-	100063422	GAAG--TTCG...
s	dasNov1.scaffold_971	14781	117	+	130999	GAA--TTTCG...
a	**score**=41484.000000					
s	hg18.chr21	9882732	233	+	46944323	CACATAA--G...
s	panTro2.chr15	81692046	233	-	100063422	CACATAA--C...
s	dasNov1.scaffold_971	14898	236	+	130999	CTCATGAGAC...
			⋮			

Table 5.5: Two MSA blocks in a multiple alignment format (MAF) file.

of position information show different statistical regularities and can thus be compressed separately. While the more subtle dependencies are accounted for in a pre-processing step, the remaining data is compressed as a sequence using CTW described in Section 4.2.2. Being a higher order statistical prediction based coder, CTW is capable of learning and using the Markov type dependencies left in the sequence. In the following, the exploited statistical regularities are discussed for each type of position information.

- (**score** - alignment score) Each MSA block is associated with an alignment score. This score is computed by the alignment algorithm. The scores are variable integer values mostly 4 to 6 digits long. The alignment scores are compressed as a sequence of digits separated by a space symbol using CTW. Although the digits are equally likely, the space symbol is more frequent and never followed by a space symbol. The CTW algorithm is capable to learn and benefit from these regularities.

- (**spec** - species) Homologous sequences are typically only found in a smaller subset of all species in the dataset. Thus, for each MSA it is necessary to encode the corresponding species subset. The ordering of the species in the subset is always the same and corresponds to the ordering in the phylogenetic tree, depicted in Figure 5.4. Since human is used as reference in the multiz28way dataset, every MSA block starts with the human species. Only homologous sequences that have not diverged too much can be identified by the alignment algorithm. Therefore, species distant to human occur very rarely in the dataset. Additionally, the phylogenetic dependencies described by the tree influence the species present in a subset. For example, if a homolog to human has been found in the mouse it is very likely that a homolog is also present in the rat since mouse and rat are very closely related compared to their evolutionary distance to human. In the proposed algorithm the species are encoded as a sequence from a 28 symbol alphabet using CTW. The actual names

of the reference genomes are stored in a separate table. Note that since every MSA block starts with the human species, the number of species in each MSA block can be recovered from the decoded species sequence and does not have to be stored separately.

- (**chr** - chromosome) The regular chromosome names (chr1, chr2 ... chrM) are encoded as a sequence from a higher order alphabet using CTW. The compression benefits from the fact that the standard naming scheme always assigns the name to the chromosomes in a species in decreasing order of the chromosome length. The longest chromosome is given the name chr1. The longer chromosomes are likely to contain more homologs due to their size and are thus also more frequent in the whole genome alignment dataset. Apart from the standard chromosome names, there exist names like scaffold_971 etc. in the multiz28way dataset, see Table 5.5. These originate from the limitations of modern high-throughput DNA sequencing methods, i.e. during "shotgun"-sequencing the DNA is split into many overlapping chunks of several hundred base pairs length that are subsequently sequenced and computationally re-assembled to chromosomes. Unfortunately, not all chunks can be unambiguously positioned in the actual chromosomes and have to be referenced using the scaffold_number scheme, where the chunk number is highly variable. The "unpositioned" chromosome chunks thus cannot be encoded together with the regular chromosomes and have to be encoded separately.

- (|**spec.chr**| - chromosome length) Each chromosome in a species has a certain length. For compression purposes it is sufficient to encode the length once for each spec.chr combination. Lengths of repeating spec.chr combinations are skipped in a pre-processing step. The remaining lengths are encoded as a sequence of digits separated by a space symbol using CTW. This is analogous to the encoding of the alignment scores. The proposed algorithm encodes the lengths in the order of appearance of the unique spec.chr combinations in the MAF file. This represents an efficient way of encoding also the association between each length and the respective spec.chr combination.

- (**start** - position in chromosome) In order to efficiently compress the start positions in the chromosomes it is important to realize that contiguous homologous sequences often get split over several consecutive MSA blocks due to large scale deletions and translocations. For example, the homologs of *hg18*, *panTro2*, *dasNov1* are contiguous in the two MSA blocks shown in Table 5.5, i.e. $\text{MSA}_2(\text{start})=\text{MSA}_1(\text{start})+\text{MSA}_1(|\text{seq}|)$. The alignment had to be split into two MSA blocks due to a shorter homolog in *eriEur1* resulting from a large scale deletion. Therefore, instead of encoding the absolute value of the start position, it is advisable to encode the offset to the preceding homolog segment if a homolog for the same species and on the same chromosome is present in the previous MSA block, i.e. instead of $\text{MSA}_x(\text{start})$ encode $\text{MSA}_x(\text{start})\text{-}\text{MSA}_{x-1}(\text{start})\text{-}\text{MSA}_{x-1}(|\text{seq}|)$ in such case. The replacement is done in the pre-processing step. The resulting integer values are as previously encoded as a sequence of digits separated by a space symbol using CTW.

- (|**seq**| - sequence length) The length of the homologous region in each species is already encoded in the MSA block and does not have to be encoded separately. It is the number of nucleotides in the corresponding MSA row.
- (**strand** - DNA strand) Homologous sequences are equally likely found on the reference "+" and the complementary strand "-". The binary strand information might thus seem incompressible at first. However, following the argument about contiguous homologous sequences split onto several consecutive MSA blocks, used to encode the start position, the strand information does not have to be encoded for sequence segments in an MSA block contiguous with the segments in the previous block. These are skipped in a pre-processing step. The remaining binary sequence is encoded using CTW.
- (**MSA**) An efficient compression scheme for MSA blocks has been introduced and analysed in Section 5.3 under the name MSAc.

The compressibility of the different types of position information using the dependencies described above is summarized in Table 5.6. Most of the statistical dependencies involve correlations between subsequent MSA blocks. Note that using such dependencies makes it impossible to decode the position information for any single MSA block without having to decode the position information for all the preceding blocks in the MAF file.

Type	Description	Compressibility
(**score**)	MSA alignment score	low
(**spec**)	species	very high
(**chr**)	chromosome	very high
(|**spec.chr**|)	chromosome length	high
(**start**)	homolog start position	high
(**strand**)	homolog DNA strand	average

Table 5.6: Qualitative summary of the compressibility of the supplementary position information in MAF files. The position information occupies around 20% of the file size.

5.4.3 Performance and Comparison

In the following the overall compression performance of the proposed lossless MAFc file compressor shall be evaluated. The original MAF files are stored using ASCII coding. Every character including the MSA symbols is stored using 1 byte or 8 bits. Note that the use of ASCII coding introduces a significant overhead. For example, for MSA symbols being from a 5 symbol alphabet only 2.32 bits/symbol would be needed in uncompressed state. The ASCII coded original MAF files are thus very large and highly compressible. The MAFc compressor encodes the MSA blocks using the MSAc algorithm proposed in Section 5.3, the position information is compressed as described in the previous section. Figure 5.11 depicts exemplary the composition of the chr21.maf file before (left) and

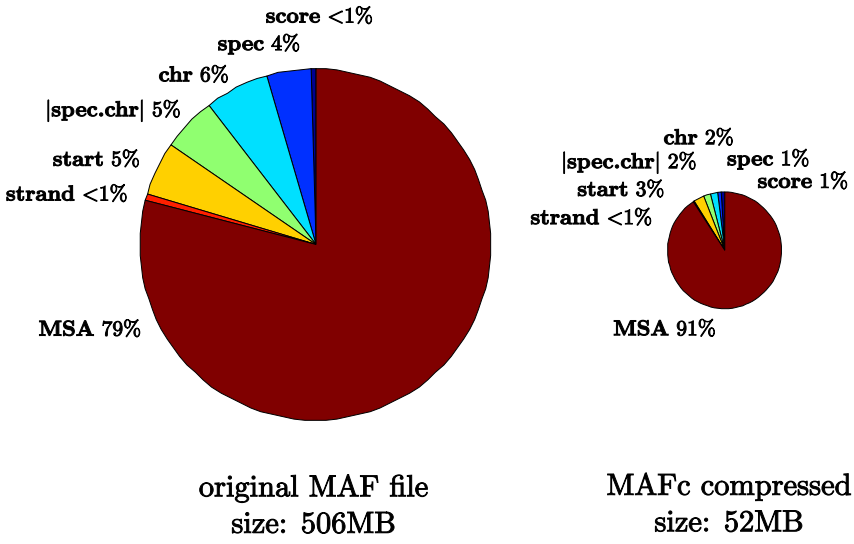


Figure 5.11: Composition of the chr21.maf file before (left) and after compression using MAFc (right). The overall file size is reduced almost tenfold.

after compression using MAFc (right). In case of the original file (left) the proportion refers to the percentage of ASCII symbols occupied by the given type of data. For the compressed file the percentage values refer to the relative amount of bits spent on encoding the respective feature. The overall file size is reduced almost tenfold. The decrease is reflected by the area reduction of the pie chart. The MSA blocks occupy 91% of the compressed file as opposed to 79% of the original file. In other words, the position information is compressed more efficiently than the MSA blocks. The individual comparison of the relative space requirement of the different kinds of position information before and after compression implicates that the integer values **score**, |**spec.chr**|, **start** are more difficult to compress than the **spec**, **chr**, **strand**. The latter group benefits from the fact that it is a coarser type of position information, i.e. there are only 28 different species, < 30 different chromosomes and 2 DNA strands. Consequently, the several ASCII symbols long species and chromosome names can be efficiently encoded as symbols from a higher order alphabet. As for the DNA strand information, solely by using binary instead of ASCII coding an 8 fold reduction is achieved.

However, the overhead introduced by using ASCII coding in the original MAF files can also be utilized by traditional universal statistical and dictionary based compression algorithms like CTW and LZ described in detail in Chapter 4. Therefore, in order to assess the benefit of using the source specific compressor MAFc it has to be compared also to these algorithms. Table 5.7 provides a detailed comparison. The original file size of the different MAF files in the multiz28way dataset and the relative size after compression using MAFc, CTW and LZ is shown. The size of the individual MAF files largely correlates with the size of the respective human chromosomes and also gradually decreases with the chromosome number. The only significant exception is chr20 that contains a higher proportion of conserved regions [DMA+01]. Both CTW and LZ manage to reduce the size of the MAF files around fivefold. Thereby, the CTW algorithm performs around 7% better than LZ. The proposed MAFc compressor reduces the file size almost tenfold to 10.4% of the

File	original size	relative compressed size MAFc	CTW	LZ
chr1.maf	4.0 GB	10.3%	19.4%	21.1%
chr2.maf	4.3 GB	10.4%	19.6%	21.1%
chr3.maf	3.6 GB	10.3%	19.6%	21.1%
chr4.maf	3.1 GB	10.5%	19.7%	21.3%
chr5.maf	3.1 GB	10.3%	19.5%	21.2%
chr6.maf	3.0 GB	10.3%	19.5%	21.2%
chr7.maf	2.6 GB	10.4%	19.6%	21.2%
chr8.maf	2.4 GB	10.7%	19.6%	21.4%
chr9.maf	2.0 GB	10.4%	19.6%	21.2%
chr10.maf	2.3 GB	10.4%	19.5%	21.2%
chr11.maf	2.3 GB	10.4%	19.5%	21.2%
chr12.maf	2.2 GB	10.3%	19.5%	21.1%
chr13.maf	1.6 GB	10.8%	19.6%	21.3%
chr14.maf	1.6 GB	10.2%	19.5%	21.0%
chr15.maf	1.4 GB	10.3%	19.5%	21.1%
chr16.maf	1.3 GB	10.4%	19.5%	21.3%
chr17.maf	1.4 GB	10.7%	19.2%	21.0%
chr18.maf	1.3 GB	10.7%	19.6%	21.4%
chr19.maf	0.7 GB	11.8%	19.4%	21.2%
chr20.maf	1.1 GB	10.9%	19.6%	21.4%
chr21.maf	0.5 GB	11.6%	19.6%	21.5%
chr22.maf	0.5 GB	11.8%	19.6%	21.6%
chrX.maf	2.0 GB	10.5%	20.0%	21.0%
chrY.maf	0.1 GB	14.7%	22.4%	23.2%
multiz28way	**48.5 GB**	**10.4%**	**19.7%**	**21.3%**

Table 5.7: Relative file size in % of compressed MAF files using different compression algorithms. The original uncompressed file size in Gigabyte (GB) is provided in the second column.

original file size on average and performs twice as good as the competitors. While CTW and LZ perform comparably for all regular chromosomes, MAFc suffers a slight penalty when compressing small MAF files, see chr19,21,22. This relates to the slightly worse compressibility of the respective MSA blocks, see also Table 5.4. Particularly difficult to compress for all algorithms seems chromosome Y. This relates to the fact that it is a haploid chromosome occurring only in the male lineage. For this reason it is subject to stronger evolutionary pressure and in general shows a higher degree of conservation than other chromosomes resulting in MSA blocks with fewer gaps that are harder to compress.

A beneficial capability of MSAc is that any individual MSA block can be decompressed without having to decompress all the preceding blocks. As discussed in Section 5.3.3 this opens up the possibility of reducing the storage requirement of database servers. This

advantageous property has been preserved in MAFc. It is only the supplementary position information that has to be decompressed for the entire MAF file at once. This represents a great advantage opposed to the competitors. Currently, when setting up a local database the uncompressed MAF files have to be used. With MAFc the compressed files could be used as the database back end reducing the overall storage requirement tenfold.

5.5 Summary

Recent advances in high-throughput DNA sequencing technology have led to the completion of evermore whole genome sequencing projects. The amount of sequence data becoming available is growing exponentially raising questions about efficient storage and distribution. This chapter has focused on source coding techniques for the voluminous sequence datasets used in genetics.

First, an overview of compression algorithms specifically developed for DNA sequence compression has been provided. The compressibility of genomic DNA sequence data seems rather limited. Consequently, DNA specific compression algorithms have so far not been broadly adopted for storage purposes. The key component to achieving any compression at all has turned out to be appropriate modelling of genomic sequence evolution.

The availability of multiple sequenced genomes has given rise to the field of comparative genomics. However, evolutionary mutations complicate genome comparisons. Large scale deletions, duplications and translocations result in rearrangements in the genome sequences. Additionally, the genomes are subject to small scale mutations comprising insertions, deletions and substitutions affecting only several neighbouring nucleotides. Thus, in order to compare genomes, the homologous regions sharing common ancestry have to be identified first. Subsequently, these have to be aligned using an extra gap symbol to compensate for the small scale insertions and deletions. A whole genome alignment is thus a set of many aligned blocks of homologous regions. Computing whole genome alignments requires huge computational power. Therefore, they are typically precomputed by several labs and curated by large public databases. In this thesis the first highly efficient lossless compression scheme for the alignment blocks has been proposed. The nucleotide portion of the alignment blocks is compressed using well established statistical models of evolutionary substitutions. Unfortunately, no elaborate statistical models exist for insertions and deletions. However, the gap positions were found to be well compressible using techniques from lossless binary image compression. The alignment blocks are distributed in form of multiple alignment format files containing also position information about the aligned regions. It is possible to compress this supplementary information efficiently by exploiting subtle dependencies in the position information of consecutive alignment blocks. The herein proposed compression MAFc algorithm for multiple alignment files reduces the file size tenfold and is twice as efficient as the universal compression algorithms like the dictionary based Lempel-Ziv or the statistical prediction based Context Tree Weighting. A great advantage of the MAFc algorithm against its competitors is that it allows to decompress individual alignment blocks without having to decompress the entire dataset. This property opens up completely new possibilities. For example, MAFc could also be

used as back end of database servers. Such servers are often queried for by researchers for individual alignments.

The proposed innovative algorithm for the compression of whole genome alignments has received the 2009 Capocelli Prize, which is the Best Paper Award of the Data Compression Conference [HDCH09]. Its extension and theoretical analysis was presented in the IEEE Transactions on Information Theory [HDCH10].

Part II

Marker Synchronization in Genetics

6

Marker Synchronization in Engineering

This chapter will concentrate on the traditional marker synchronization problem as formulated in [Bar53]. Particular positions in the data stream are marked by inserting short sequence patterns, also called syncwords, before the transmission over a noisy channel. At the receiver the synchronizer tries to identify the insertion positions in the received data. In communications engineering markers are typically used to distinguish the borders of data frames. The engineering aspects of marker synchronization shall be studied in detail in this chapter. After defining the synchronization model in Section 6.1 suitable likelihood functions for marker detection will be derived in Section 6.2. In Section 6.3 techniques to study the synchronization performance of different markers of the same length will be developed. The sequence specific binding taking place on the molecular level is comparable to marker synchronization. The analysis presented in this chapter will account for the specific aspects of molecular marker synchronization, such that the derived models can be applied to study actual binding sites in Chapter 7.

6.1 Synchronization Model

The original data stream with embedded syncwords is transmitted over a noisy channel. At the receiver the synchronizer searches for the positions in the received data stream, where the syncwords have originally been embedded. It is assumed that symbolwise synchronization has already been established and that the syncwords have a fixed length L. Using the received data stream $\boldsymbol{y}$, the syncword detector evaluates a suitable likelihood function $L(\mu)$ for each position μ in $\boldsymbol{y}$ describing the likelihood that μ corresponds to

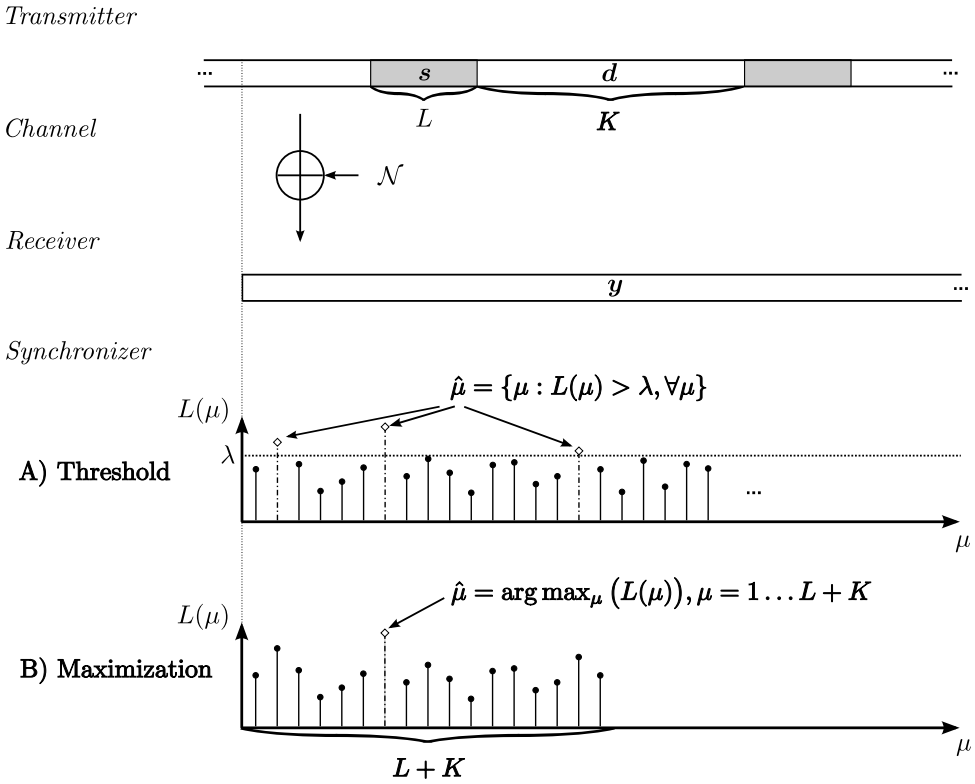


Figure 6.1: Synchronization model: (top) transmission model, (bottom) detection strategies: A) threshold synchronizer, B) maximization synchronizer

the insertion point of a syncword in the original transmitted sequence. In general, two basic types of marker synchronizers are used depending on whether information about the distribution of syncword insertion positions in the data stream is available at the receiver or not. The two types are described in the following and depicted in Figure 6.1.

6.1.1 Threshold Synchronizer

If the distribution of syncword insertion positions in the data stream is unknown, the synchronizer has to rely on threshold based detection. A position μ is declared to correspond to a syncword insertion point if $L(\mu) \geq \lambda$, see Figure 6.1 A). The choice of a suitable threshold value λ will be discussed later in the context of optimal likelihood functions derived in Section 6.2. In Chapter 7, it will be shown that marker synchronization at the molecular level in the cell closely resembles threshold based synchronization.

6.1.2 Maximization Synchronizer

If the syncwords are inserted periodically into the data stream, which is often the case in applications from communications engineering, the success rate of the synchronizer can be significantly improved using a maximization based detection strategy. Assuming that syncwords of length L are inserted every K data symbols, the synchronizer knows that the syncword has been inserted at exactly one position in a received window of length

$L+K$. Therefore, it evaluates the likelihood function for all possible insertion positions $\mu = 1 \dots L+K$ in a received window and declares $\hat{\mu} = \arg\max_{\mu}(L(\mu))$ maximizing the likelihood function to be the most likely syncword insertion position, see Figure 6.1 B). In its simplest version the synchronizer does this for a single block of length $L+K$. Further improvement can be achieved by combining predictions from several consecutive blocks. Another possibility is to pass a list of likely candidates to higher processing stages like the channel decoder as suggested in [Rob95]. In this manner, a priori knowledge about the channel encoded data frames can be used to improve the synchronization performance. A maximization synchronizer can also be used if the syncwords were inserted aperiodically according to a known distribution and if the position of the preceding syncword has already been successfully established [KB04].

6.2 Likelihood Functions for Marker Synchronization

The optimal likelihood function and suitable approximations thereof for the standard model assuming binary phase shift keying (BPSK) modulation, AWGN channel and symbolwise IUD data frames, were first introduced by Massey in [Mas72]. Herein a general optimal LLR based likelihood function suitable for both threshold and maximization based synchronization will be derived. It will be shown that using a single fixed syncword is optimal and that in this case the LLR is equivalent to the mutual self-information between the received sequence at a position and the syncword. The derived LLR will be adapted to the standard model and shown to be proportional and thus equivalent to Massey's proposal. Suitable approximations of the self-information based likelihood function will be proposed for low and high signal to noise ratio (SNR) and compared to the approximations proposed by Massey. The proposed low SNR approximation will be shown to be superior. In order, to be able to account for the particularities of the synchronization mechanism observed in genetics the derived LLR will be adapted to the discrete memoryless case and the evolutionary substitution channel models.

6.2.1 Optimal General Log Likelihood Ratio Function

Given the received sequence $\boldsymbol{y}$, the optimal estimation function for the starting position μ of the syncword is the a-posteriori probability $p(\mu|\boldsymbol{y})$ as used by the MAP estimator. However, it simplifies to the ML estimation function $p(\boldsymbol{y}|\mu)$, since each position μ is assumed to be equally likely

$$L_{MAP}(\mu) = p(\mu|\boldsymbol{y}) \cong p(\boldsymbol{y}|\mu)P(\mu) \propto L_{ML}(\mu) = p(\boldsymbol{y}|\mu). \tag{6.1}$$

In the following, the optimal likelihood function $L_{ML}(\mu) = p(\boldsymbol{y}|\mu)$ shall be simplified.

Assume a received window $\boldsymbol{y} = (y_1 \dots y_{L+K})$ of size $L+K$ with one syncword insertion of a marker pattern of length L, see Figure 6.1. For a time-invariant memoryless channel and IID data symbols

$$L_{ML}(\mu) = \sum_{\boldsymbol{s}} \prod_{i=1}^{L} p(y_{\mu+i-1}|s_i)P(\boldsymbol{s}) \cdot \prod_{i=L+1}^{L+K} \sum_{d} p(y_{\mu+i-1}|d)P(d), \tag{6.2}$$

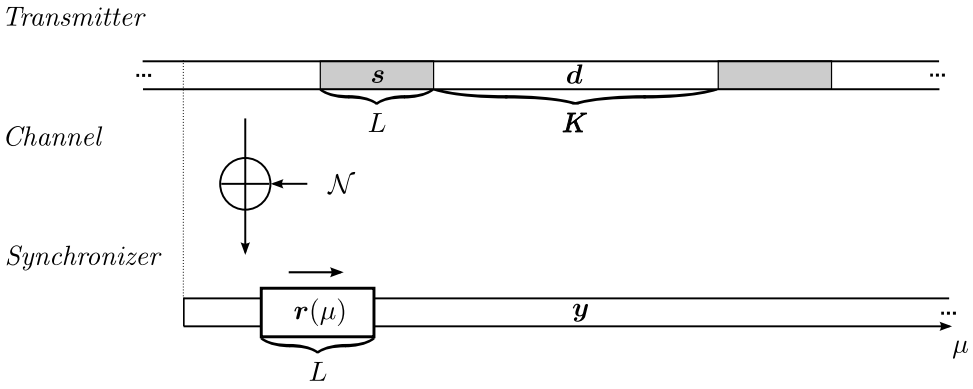


Figure 6.2: Optimal Synchronizer for IID data.

where $\boldsymbol{s}$ corresponds to a possible marker pattern (syncword), $P(\boldsymbol{s})$ is the probability that the marker $\boldsymbol{s}$ is used at the transmitter, d is a data symbol and $P(d)$ is the data symbol distribution. Note that for different positions μ in the received window $\boldsymbol{y}$ the likelihood function is evaluated in a tailbiting fashion $y_i = y_{i \bmod (L+K)}, \forall i$. Thus, in order to greatly simplify the expression we can divide by the μ independent constant term

$$\prod_{i=1}^{L+K} \sum_{\forall d} p(y_i|d)P(d). \tag{6.3}$$

The logarithm of the resulting expression is the following LLR

$$L_{LLR}(\mu) = \log \frac{\sum_{\boldsymbol{s}} \prod_{i=1}^{L} p(y_{\mu+i-1}|s_i)P(\boldsymbol{s})}{\prod_{i=1}^{L} \sum_{d} p(y_{\mu+i-1}|d)P(d)}. \tag{6.4}$$

It is dependent only on $(y_\mu \dots y_{\mu+L-1}) = \boldsymbol{r}(\mu)$, the portion of the received sequence $\boldsymbol{y}$ tested against the assumption to correspond to the originally embedded syncword sequence. Thus, the optimal detector can compute (6.4) using only a window of the size of the syncword L containing $\boldsymbol{r}(\mu)$ as depicted in Figure 6.2. For a communications system this means that the computation of the likelihood function can start in a sliding window fashion already after receiving the first L symbols. For the molecular synchronization system this means that a detector molecule stretching only around the syncword can operate optimally. In order to simplify the notation, in the remaining part of this section, the dependency of the likelihood function from the position μ will be implicitly assumed but notationally omitted, i.e. $\boldsymbol{r}(\mu) \rightsquigarrow \boldsymbol{r}$ and $L_{LLR}(\mu) \rightsquigarrow L_{LLR}$. The derived likelihood function L_{LLR} is in fact a LLR of the null hypothesis $\mathcal{H}_0$ that the observation $\boldsymbol{r}$ originates from the transmission of a syncword and the alternative hypothesis $\mathcal{H}_1$ that it originates from a transmission of the data portion of the frames

$$\boxed{L_{LLR} = \log \frac{\sum_{\boldsymbol{s}} p(\boldsymbol{r}|\boldsymbol{s})P(\boldsymbol{s})}{\prod_{i=1}^{L} \sum_{d} p(r_i|d)P(d)} = \log \frac{p(\boldsymbol{r}|\mathcal{H}_0)}{p(\boldsymbol{r}|\mathcal{H}_1)}.} \tag{6.5}$$

The likelihood function L_{LLR} is optimal for both threshold and maximization synchronizer under the assumption that the source of data frames is IID, the channel time-invariant and memoryless. For the threshold based synchronizer the threshold λ is chosen according to the Neyman-Pearson criterion in (2.17) by fixing the probability of falsely rejecting the null hypothesis $\mathcal{H}_0$ that the observation $\boldsymbol{r}$ originates from a syncword insertion. L_{LLR} can be interpreted as the weight of evidence for $\mathcal{H}_0$ over $\mathcal{H}_1$.

Thus, the Kullback-Leibler distance

$$D(P(\boldsymbol{R}|\mathcal{H}_0)||P(\boldsymbol{R}|\mathcal{H}_1)) = \int \sum_{\boldsymbol{s}} p(\boldsymbol{r}|\boldsymbol{s})P(\boldsymbol{s}) \log \frac{\sum_{\boldsymbol{s}} p(\boldsymbol{r}|\boldsymbol{s})P(\boldsymbol{s})}{\prod_{i=1}^{L} \sum_{d} p(r_i|d)P(d)} d\boldsymbol{r}, \tag{6.6}$$

where $\boldsymbol{R}$ denotes the random variable corresponding to the received sequence $\boldsymbol{r}$, can be used to measure the mean information per sample for discriminating in favour of hypothesis $\mathcal{H}_0$ that $\boldsymbol{r}$ originates from a syncword transmission against the hypothesis $\mathcal{H}_1$ that it originates from the data transmission. Since the conditional channel PDF and the data symbol distribution would typically be predetermined in the synchronization system to be designed, the only free system parameter is the syncword use distribution $P(\boldsymbol{s})$ at the transmitter. Optimal choice for $P(\boldsymbol{s})$ is one that maximizes the discrimination information $D(P(\boldsymbol{R}|\mathcal{H}_0)||P(\boldsymbol{R}|\mathcal{H}_1))$. In fact, for IUD data the discrimination information is maximized when any single syncword is permanently used at the transmitter. In case of IID data, the least likely single syncword according to the data symbol distribution achieves the maximum. When using a single syncword, the optimal likelihood function becomes:

$$\begin{aligned} L_{LLR} &= \log \frac{\prod_{i=1}^{L} p(r_i|s_i)}{\prod_{i=1}^{L} \sum_{d} p(r_i|d)P(d)} = \sum_{i=1}^{L} \log \frac{p(r_i|s_i)}{\sum_{d} p(r_i|d)P(d)} \\ &= \sum_{i=1}^{L} \log \left(\frac{p(r_i|s_i)}{\sum_{d} p(r_i|d)P(d)} \cdot \frac{P_D(s_i)}{P_D(s_i)} \right) = \sum_{i=1}^{L} \log \frac{p(r_i, s_i)}{p_D(r_i)P_D(s_i)} \\ &= \sum_{i=1}^{L} I(s_i; r_i) = I(\boldsymbol{s}; \boldsymbol{r}), \end{aligned} \tag{6.7}$$

where $I(\boldsymbol{s}; \boldsymbol{r})$ is the mutual self-information between the syncword $\boldsymbol{s}$ and the received sequence $\boldsymbol{r}$, whereas $\boldsymbol{r}$ is assumed to be a received realization of the IID data process, see also Section 2.2.1.

6.2.2 Likelihood Function for AWGN Channel

The standard model used in the literature in the context of marker synchronization assumes the use of a fixed single syncword and IUD distributed data frames. The transmission follows over an AWGN channel $y = x + n$ using BPSK modulation $x \in \{\pm 1\}$. Thus, the channel distribution function is Gaussian $p_{Y|X}(y|x) = \mathcal{N}(x, \sigma^2)$ with $\sigma^2 = \frac{N_0}{2E_s}$, where y is the normalized output of the matched filter, E_s the average transmission power per symbol, and $\frac{N_0}{2}$ the two sided power spectral density of the white noise.

Inserting the Gaussian distribution function from (2.10) into (6.7) and simplifying the expression results in:

$$L_{LLR} = \sum_{i=1}^{L} \log \frac{e^{r_i s_i/\sigma^2}}{\sum_d e^{r_i d/\sigma^2} P(d)}. \tag{6.8}$$

Assuming that the data symbols are IUD and using $\cosh(x) = (e^x + e^{-x})/2$ leads to

$$L_{LLR} = \frac{1}{\sigma^2} \sum_{i=1}^{L} r_i s_i - \sum_{i=1}^{L} \ln \cosh \left(\frac{r_i}{\sigma^2} \right). \tag{6.9}$$

Massey's Rule

Note that the L_{LLR} in (6.9) derived for the standard model is proportional and thus equivalent to the optimal likelihood function L_M proposed by Massey [Mas72]:

$$\boxed{L_M = \sum_{i=1}^{L} r_i s_i - \sigma^2 \sum_{i=1}^{L} \ln \cosh \left(\frac{r_i}{\sigma^2} \right) = L_{LLR} \sigma^2,} \tag{6.10}$$

where the first term corresponds to the so called soft-correlation rule L_{sC}. Alternatively to the soft-correlation, the so called hard-correlation rule L_{hC} can be used

$$L_{sC} = \sum_{i=1}^{L} r_i s_i, \qquad L_{hC} = \sum_{i=1}^{L} \operatorname{sign}(r_i)\, s_i. \tag{6.11}$$

Approximations of Massey's Rule

In order to simplify the optimal rule, Massey has proposed to approximate $\cosh(x) \approx e^{|x|}/2 : x \gg 1$ for large SNR (E_s/N_0) and $\ln(\cosh(x)) \approx x^2/2 : x \ll 1$ for low SNR. This results in

$$\tilde{L}_M^{high} = \sum_{i=1}^{L} r_i s_i - \sum_{i=1}^{L} |r_i| \qquad \text{for high SNR,} \tag{6.12}$$

$$\tilde{L}_M^{low} = \sum_{i=1}^{L} r_i s_i - \frac{1}{2\sigma^2} \sum_{i=1}^{L} r_i^2 \qquad \text{for low SNR.} \tag{6.13}$$

Conditional Self-Information

In Section 6.2.1 it has been argued that the use of a single fixed syncword for marker synchronization is optimal. In (6.7) the optimal likelihood function for this case has been shown to be equivalent to the mutual self-information $I(\boldsymbol{s};\boldsymbol{r})$ between the fixed syncword $\boldsymbol{s}$ and the received sequence $\boldsymbol{r}$. Since, $I(\boldsymbol{s};\boldsymbol{r}) = H(\boldsymbol{s}) - H(\boldsymbol{s}|\boldsymbol{r})$ and $H(\boldsymbol{s})$ is independent of μ, the likelihood function L_{LLR} can be simplified using only the conditional self-information

$$\boxed{L_H = -H(\boldsymbol{s}|\boldsymbol{r}) = -\sum_{i=1}^{L} H(s_i|r_i).} \tag{6.14}$$

The value of L_H can be interpreted as the negative uncertainty (in bits if log = ld) about the syncword $\boldsymbol{s}$ given the received sequence $\boldsymbol{r}$ at position μ in the received data stream. In the context of EXIT chart analysis it has been shown [TH02, Hag04] that for BPSK modulated transmission over an AWGN channel, as used in the standard model, the conditional entropy can be computed in a simple way, namely as

$$H(X|Y) = E\{\log(1 + e^{-xL_{X|Y}(x|y)})\}, \tag{6.15}$$

where

$$L_{X|Y}(x|y) = \ln\left(\frac{p(y|x=+1)}{p(y|x=-1)}\right) + \underbrace{\ln\left(\frac{P(x=+1)}{P(x=-1)}\right)}_{L_X} = \frac{2}{\sigma^2}y + L_X. \tag{6.16}$$

Thus, (6.14) becomes:

$$L_H = -\sum_{i=1}^{L} \log(1 + e^{-s_i(2r_i/\sigma^2 - L_D)}), \tag{6.17}$$

where L_D is the log-likelihood of the IID data symbols. The likelihood function L_H is simpler to compute than the one introduced by Massey (6.10) and it is not limited to IUD data sources. However, assuming IUD data further simplification is possible:

$$\boxed{L_H = -\sum_{i=1}^{L} \log(1 + e^{-2r_i s_i/\sigma^2}).} \tag{6.18}$$

Approximations of Conditional Self-Information

It is possible to approximate (6.18) in several different ways. For high SNR the max-log approximation $\log(x+y) = \log(\max(x,y)) : ||x-y|| \gg 0$ is suitable leading to

$$\tilde{L}_H^{high} = -\sum_{i=1}^{L} (\max(0, -r_i s_i)) = \tilde{L}_M^{high}. \tag{6.19}$$

This approximation is identical to Massey's high SNR rule in (6.12). Analogously, only terms where sign $(r_i) \neq$ sign (s_i) contribute to the computation. A suitable approximation

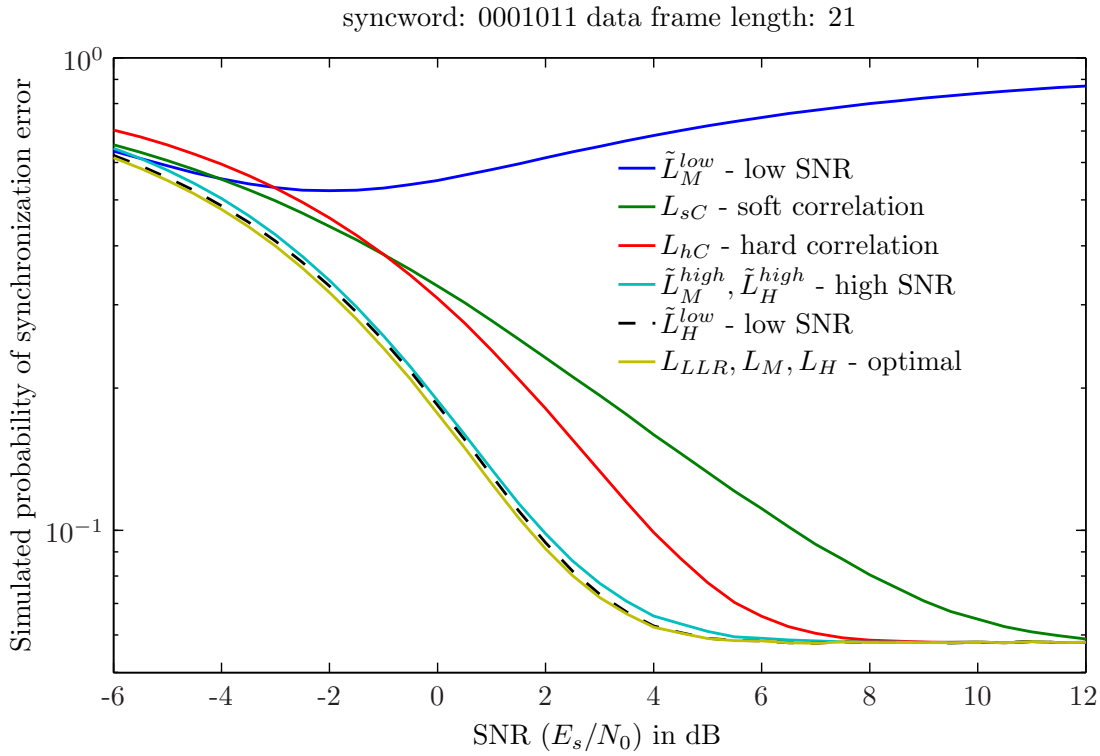


Figure 6.3: Synchronization performance of different likelihood functions. Detector: maximization synchronizer, Syncword: 0001011, Data frame length $K = 21$, Iterations: $3 \cdot 10^6$

for low SNR is achieved using the series expansion $\ln(1+x) \approx x : -1 < x < 1$

$$\boxed{\tilde{L}_H^{low} = -\sum_{i=1}^{L} e^{-2r_i s_i/\sigma^2},} \tag{6.20}$$

At this point, it can be asked whether this approximation is better than the low SNR approximation by Massey in (6.13). Indeed, this new approximation should be superior, since it can be transformed into an approximation proportional to (6.13) by further Taylor expansion $e^x \approx (1 + x + x^2/2)$

$$\tilde{L}_H^{low} = \sum_{i=1}^{L} (2r_i s_i/\sigma^2 - (2r_i s_i/\sigma^2)^2) \propto \tilde{L}_M^{low}. \tag{6.21}$$

In order to assess the performance of the derived likelihood functions simulations for the standard model have been performed. Figure 6.3 depicts the synchronization performance for the maximization synchronizer. Monte Carlo simulations were performed in order to determine the synchronization error probability. The sequence 0001011 of length $L = 7$ was chosen as syncword. The data frames length was set to $K = 21$ and the simulation was performed using $3 \cdot 10^6$ iterations.

It can be seen that the self-information based approximation of the optimum likelihood function $\tilde{L}_H^{low}$ performs close to the optimum and outperforms all other approximations for the chosen SNR range, even for high SNR values. The low SNR approximation rule $\tilde{L}_M^{low}$

cannot be recommended since its performance degrades rapidly with increasing SNR. The plot also signifies the great improvement provided by Massey's optimal likelihood function L_M as opposed to the soft-correlation rule L_{sC} used prior to Massey's proposal. It also shows that for high SNR values the hard-correlation rule L_{hC} outperforms the soft-correlation.

6.2.3 Likelihood Functions for Discrete Memoryless Channels

A DMC channel is fully characterized by a transition probability matrix, see Section 2.2.3. Thus, the L_{LLR} function in (6.7) can be evaluated as

$$\boxed{L_{DMC} = \sum_{i=1}^{L} \log \frac{P(r_i|s_i)}{\sum_d P(r_i|d)P(d)}.} \tag{6.22}$$

In case of IUD data the denominator term can be neglected. A special case of the DMC is the BSC. The use of hard decisions in the standard model is actually equivalent to the transmission over a binary symmetric channel with error probability $p = \frac{1}{2}\operatorname{erfc}(\sqrt{E_s/N_0})$. Thus,

$$P(r_i|s_i) = \begin{cases} 1-p & \text{if } r_i = s_i \\ p & \text{if } r_i \neq s_i \end{cases}. \tag{6.23}$$

Assuming IUD data symbols and adding $L\log(p)$ to (6.22) we get the likelihood function:

$$L_{BSC} = \sum_{\substack{\forall i \in \{1,\ldots,L\} \setminus \\ i: r_i = s_i}} \log \frac{1-p}{p}, \tag{6.24}$$

which is equivalent to:

$$\boxed{L_{BSC} = \begin{cases} -d_h(\boldsymbol{s},\boldsymbol{r}) & \text{if } 0 < p \leq 0.5 \\ d_h(\boldsymbol{s},\boldsymbol{r}) & \text{if } 0.5 \leq p < 1 \end{cases}.} \tag{6.25}$$

The term $d_h(\boldsymbol{s},\boldsymbol{r})$ is the Hamming distance (number of mismatches) between the syncword $\boldsymbol{s}$ and the received sequence $\boldsymbol{r}$. This is equivalent to the hard-correlation rule L_{hC} in (6.11) for the usual transmission error probability range $0 < p \leq 0.5$.

6.2.4 Likelihood Functions for Substitution Channel Models

In the following the optimal likelihood function shall be adapted to the evolutionary substitution channel models studied in Section 3.7.1. The substitution channel models are an instance of DMC channels. The channel transition probability matrices are computed according to (3.3) using the corresponding rate matrices. The substitution process is a continuous time Markov process with the stationary distribution $\boldsymbol{\pi}$ corresponding to the background distribution of the genome $\boldsymbol{\pi} = (\pi_{\mathtt{A}}, \pi_{\mathtt{C}}, \pi_{\mathtt{G}}, \pi_{\mathtt{T}})$. In the context of marker synchronization $\boldsymbol{\pi}$ corresponds to the IID data symbol distribution P_D. The stationary

behaviour of the data symbol distribution with respect to the channel allows to further simplify the optimal likelihood function for DMC in (6.22). Thus, for the general substitution channel model GTR, see Section 3.7.1

$$\boxed{L_{GTR} = \sum_{i=1}^{L} \log \frac{P(r_i|s_i)}{\pi(r_i)}.} \tag{6.26}$$

For the symmetric K2P and JC channels that presume IUD stationary distribution further simplification is possible

$$\boxed{L_{K2P} = \sum_{i=1}^{L} \log P(r_i|s_i).} \tag{6.27}$$

In case of the JC channel the mutation probability is identical for all possible substitutions. Let p denote the probability that a mutation has occurred, then

$$P(r_i|s_i) = \begin{cases} 1-p & \text{if } r_i = s_i \text{ no mutation} \\ p/3 & \text{if } r_i \neq s_i \text{ substitution} \end{cases}. \tag{6.28}$$

Following an analogous argumentation as for the BSC channel we get

$$L'_{JC} = \begin{cases} -d_h(\boldsymbol{s}, \boldsymbol{r}) & \text{if } 0 < p \leq 3/4 \\ d_h(\boldsymbol{s}, \boldsymbol{r}) & \text{if } 3/4 \leq p < 1 \end{cases}, \tag{6.29}$$

where the term $d_h(\boldsymbol{s}, \boldsymbol{r})$ is the Hamming distance. Note that for the JC channel model the probability that no mutation occurs is always higher than the probability of a substitution. Therefore, $0 < p \leq 3/4$ and we can simplify to

$$\boxed{L_{JC} = -d_h(\boldsymbol{s}, \boldsymbol{r}).} \tag{6.30}$$

6.3 Marker Performance and Syncword Choice

In the following, the question of choosing a good syncword shall be investigated. As shown in Section 6.2.1 with respect to the optimal likelihood function any single syncword pattern performs equally well for IUD data. However, even for noiseless transmission and IUD data different patterns of the same length are not equally likely to be spuriously emulated in a received window of finite size.

Example: Consider noiseless transmission of the all zero syncword 0000000 of length $L = 7$ in binary IUD data. The marker pattern can be spuriously emulated at any position in the received window apart from the insertion point. However, the pattern 0001011 cannot be emulated at positions overlapping with the syncword. Apparently, the self-overlap property of the syncword plays an important role. ∗

In the following a method for the quantitative analysis of the synchronization properties of different syncword patterns shall be derived. In the context of quantitative syncword performance analysis it is important to consider the synchronization scheme and distinguish

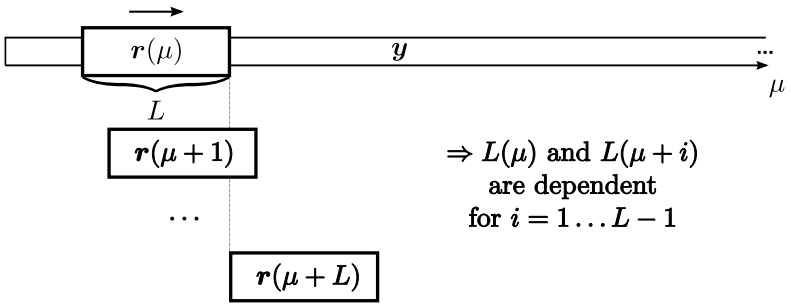


Figure 6.4: The likelihood function $L(\mu)$ exhibits dependencies for neighbouring positions μ due to overlap of received sequences $\boldsymbol{r}$ in detector window for neighbouring μ.

between threshold and maximization based synchronization. The syncword optimization problem will be defined as finding the syncword $\hat{\boldsymbol{s}}$ for which the probability $p_{sync}(\boldsymbol{s})$ of correctly recognizing the insertion position of the syncword in the received data stream is maximized. Note that from now on, in order to simplify further notation, the dependency of $p_{sync}(\boldsymbol{s})$ from the syncword $\boldsymbol{s}$ will be always assumed, but not explicitly stated unless necessary $p_{sync}(\boldsymbol{s}) \rightsquigarrow p_{sync}$. The probability p_{sync} is the probability that the position $\hat{\mu}$ declared to be a syncword by the synchronizer corresponds to an actual syncword insertion position μ_{SW}, see also Figure 6.1

$$\hat{\boldsymbol{s}} = \arg\max_{\boldsymbol{s}} \left(p_{sync}\right) = \arg\max_{\boldsymbol{s}} \left(p(\hat{\mu} = \mu_{SW})\right), \tag{6.31}$$

Computing the exact value of p_{sync} for given synchronization parameters and model is a non-trivial task due to the dependencies of the value of $L(\mu)$ for neighbouring positions μ. The likelihood value $L(\mu)$ shows dependencies on its $L-1$ neighbours. The reason is that the received values $\boldsymbol{r} = (y_\mu \dots y_{\mu+L-1})$, used by the detector for the computation of $L(\mu)$ as shown in (6.4), are partially also used for computing $L(\mu+i)$ for $i = 1 \dots L-1$, see Figure 6.4. Surprisingly, these dependencies are commonly neglected in the literature.

In the following, the focus will be placed on threshold synchronization in the context of transmission over discrete memoryless channels, since this scenario is the most relevant one for applications in molecular biology. Herein a recursive expression for exact computation of p_{sync} will be derived for this scenario and compared to $\tilde{p}_{sync}$ computed under the assumption of independent $L(\mu)$ and commonly used in the literature. It will be shown that neglecting the dependencies can lead to quite inaccurate estimates of the synchronization performance. For scenarios, where the computation of exact p_{sync} is not feasible, a qualitative answer to the question of optimal syncword selection will be attempted. The maximization based synchronization will be treated by providing a literature overview on different commonly used syncword selection criteria.

6.3.1 Threshold Based Synchronization

The threshold based synchronizer evaluates the likelihood function $L(\mu)$ over the received data stream using a sliding window of size L. It declares the first position where $L(\mu) \geq \lambda$ to be the syncword insertion position. Let k denote the starting offset of the the search from the actual syncword insertion position μ_{SW}, see Figure 6.5, and $p_{sync}(k)$ denote the

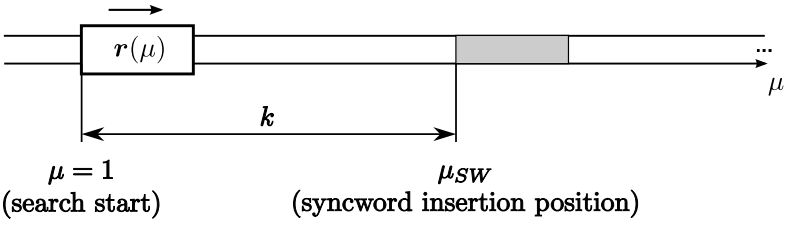


Figure 6.5: Threshold based syncword search. The search is started k steps ahead of the actual syncword insertion point μ_{SW}.

probability of successful synchronization when starting the search at offset k

$$p_{sync}(k) = p(L(\mu_{SW} - i) < \lambda, i = 1 \ldots k \wedge L(\mu_{SW}) \geq \lambda). \tag{6.32}$$

The overall synchronization success probability p_{sync} is the average synchronization probability of the synchronizer for the different starting offsets k. Thus, it depends on the probability distribution $p(k)$ of the starting offsets k

$$\boxed{p_{sync} = \sum_{k} p_{sync}(k)p(k).} \tag{6.33}$$

In the following, a novel recursive formula for the computation of the exact synchronization probability $p_{sync}(k)$ for noiseless and noisy transmission will be derived and used to identify optimal syncwords. The results will be compared to the classical approach assuming independence of the observations $L(\mu)$. Under this assumption an inexact synchronization probability $\tilde{p}_{sync}(k)$ is computed as

$$\tilde{p}_{sync}(k) = \prod_{i=1}^{k} p\left(L(\mu_{SW} - i) < \lambda\right) \cdot p\left(L(\mu_{SW}) \geq \lambda\right) = \prod_{i=1}^{k} \left(1 - p_{hit}(i)\right) \cdot p_{hit}(0) \tag{6.34}$$

where $p_{hit}(i) = p(L(\mu_{SW} - i) \geq \lambda)$ is the probability that position $\mu_{SW} - i$ is considered a syncword insertion position by the threshold synchronizer (a *hit* is claimed by the synchronizer at $\mu_{SW} - i$).

Noiseless Transmission

For noiseless transmission the synchronizer would only declare synchronization at μ if the pattern observed in the detector window $\boldsymbol{r}(\mu)$ is identical to the syncword $\boldsymbol{s}$.

In [Nie73a] it has been proven that the expected distance d in sliding window steps to the first occurrence of pattern $\boldsymbol{s}$ in a semi-infinite IUD distributed random data stream from alphabet $\mathcal{A}$ depends on the pattern length L and its self-overlap structure

$$E\{d\} = \sum_{i=0}^{L} h^{(i)} |\mathcal{A}|^i - L, \tag{6.35}$$

where $h^{(i)}, 0 \leq i \leq L$ are the so-called bifix[1] indicators of the pattern describing its

[1]The term bifix introduced was originally coined by J. Massey, see [Nie73a]

self-overlap structure

$$h^{(i)} = \begin{cases} 1 & \text{if } i = 0 \\ 1 & \text{if } (s_1 \dots s_i) = (s_{L-i+1} \dots s_L) \\ 0 & \text{if } (s_1 \dots s_i) \neq (s_{L-i+1} \dots s_L) \end{cases} \qquad \forall i = 0 \dots L \quad . \tag{6.36}$$

Per definition $h^{(L)} = h^{(0)} = 1$. In order to simplify the notation, let h denote the decimal representation of the pattern specific bifices $(h^{(L-1)} \dots h^{(1)})$. For sequences without any self-overlap (also referred to as bifix-free) $h = 0$. In general, $0 \leq h \leq 2^{L-1} - 1$. Loosely speaking, h is an indicator of the degree of the syncword's self-overlap.

Example:

$$\begin{aligned} \boldsymbol{s} = 00101 \rightarrow (h^{(L-1)} \dots h^{(1)}) = (0,0,0,0) &\rightarrow h = 0 \\ \boldsymbol{s} = 00010 \rightarrow (h^{(L-1)} \dots h^{(1)}) = (0,0,0,1) &\rightarrow h = 1 \\ \boldsymbol{s} = 00000 \rightarrow (h^{(L-1)} \dots h^{(1)}) = (1,1,1,1) &\rightarrow h = 15 \end{aligned}$$

In [BD95] it was found that the probability density function $p(d)$ of the distance to the first occurrence of the pattern $\boldsymbol{s}$ can be computed recursively as

$$p(d) = \sum_{l=1}^{\min(L,d)} \left(h^{(L+1-l)} r^{(l-1)} - h^{(L-l)} r^{(l)} \right) p(d-l), \qquad p(0) = r^{(L)}, \tag{6.37}$$

where $r^{(l)} = p(s_1 \dots s_l)$ is the probability of occurrence of the pattern prefix $(s_1 \dots s_l)$ in the data and $r^{(0)} = 1$ per definition. Obviously, for IUD symbols $r^{(l)} = 1/|\mathcal{A}|^l, \forall \boldsymbol{s}$. The generalization to IID distributed data streams was introduced in [BSV05][2].

Example: Assuming binary IID data where $p(0) = q$ and $p(1) = 1 - q = p$

$$\begin{aligned} \boldsymbol{s} = 010 \rightarrow (r^{(0)}, r^{(1)}, r^{(2)}, r^{(3)}) &= (1, q, pq, pq^2) \\ \boldsymbol{s} = 100 \rightarrow (r^{(0)}, r^{(1)}, r^{(2)}, r^{(3)}) &= (1, p, pq, pq^2) \end{aligned}$$

Since, $p(d)$ is a PMF $\sum_{d=0}^{\infty} p(d) = 1$. Additionally, from the recursive definition follows that $p(d)$ is a monotonically decreasing function, i.e. $p(d-1) \geq p(d)$.

The probability $p_{sync}(k)$ of first observing the syncword after sliding k steps is very similar to the search for the syncword pattern in random data. The difference is that the syncword has been embedded into the random data stream at the position with a fixed probability of one. Thus, the recursive definition in (6.37) can be used to recursively compute $p_{sync}(k)$ by adjusting the initial probability of the recursion

$$\boxed{p_{sync}(k) = \sum_{l=1}^{\min(L,k)} \left(h^{(L+1-l)} r^{(l-1)} - h^{(L-l)} r^{(l)} \right) p_{sync}(k-l), \qquad p_{sync}(0) = 1.} \tag{6.38}$$

This simply corresponds to a scaling of the PMF in (6.37) by a factor of $1/r^{(L)}$. The reader is referred to Appendix B.2 for a derivation and intuitive explanation of Equation (6.38).

[2]Note that $r^{(l)}$ was mistakenly defined as the probability of the suffix tail $(s_{L-l+1} \dots s_L)$ in [BSV05]

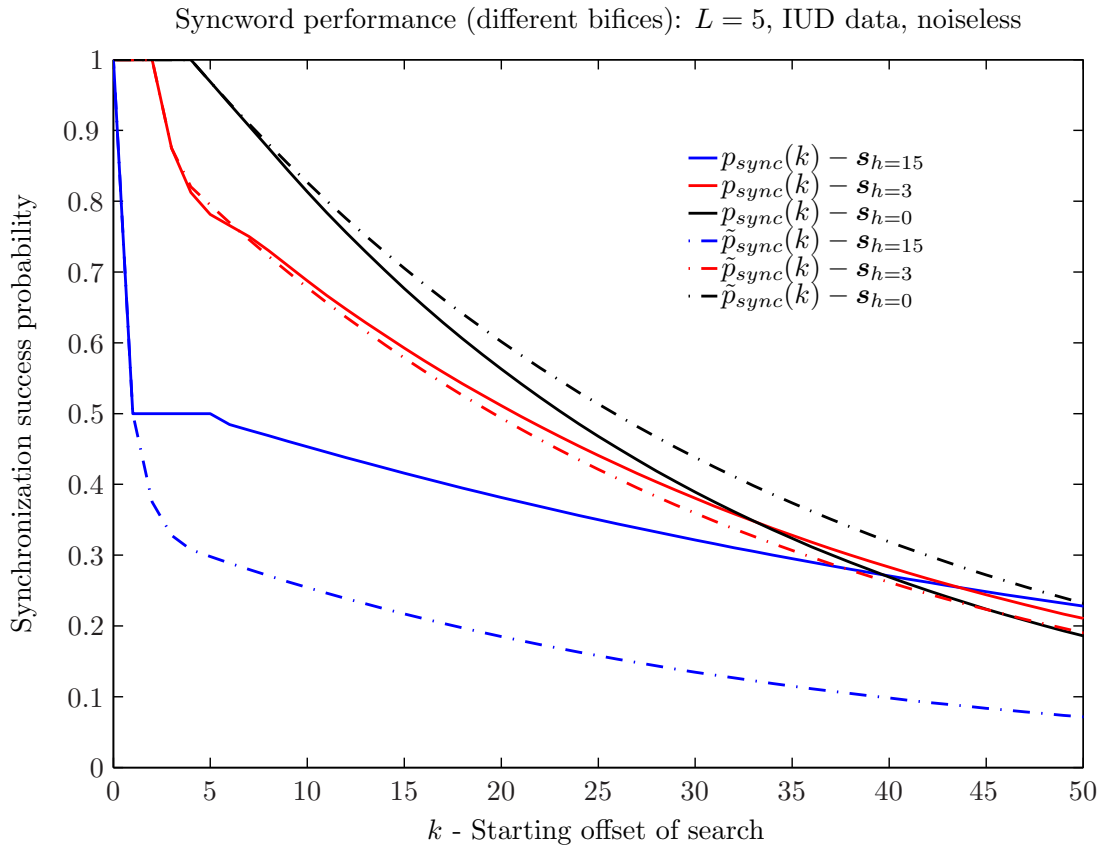


Figure 6.6: Exact $p_{sync}(k)$ and approximated $\tilde{p}_{sync}(k)$ probability of successful synchronization if starting k symbols ahead of the syncword insertion point (syncword length $L = 5$, IUD data, noiseless channel).

Figure 6.6 depicts $p_{sync}(k)$ for syncwords of length $L = 5$ with different bifix patterns h. The data symbols are assumed to be IUD. Thus, all syncwords with the same bifix pattern perform equally well. The dotted lines represent the approximation $\tilde{p}_{sync}(k)$ in (6.34) assuming independent $L(\mu)$. Note that $p_{hit}(i)$ in (6.34) equals to

$$p_{hit}(i) = \begin{cases} h^{(L-i)} r^{(i)} & \text{if } i < L \\ r^{(L)} & \text{else} \end{cases} . \tag{6.39}$$

The results imply that the approximation $\tilde{p}_{sync}(k)$ overestimates the actual synchronization performance $p_{sync}(k)$ for sequences with a small degree of self-overlap h and underestimates the performance for sequences with a large self-overlap. For small values of k bifix-free sequences have a higher probability to be correctly detected. For high values of k self-overlapping sequences perform better.

In the following, the overall synchronization success probability p_{sync} shall be assessed for different syncwords. As shown in (6.33), p_{sync} depends on the distribution $p(k)$ of the search starting offsets k. Assume a fixed data frame length K and that each syncword search is started randomly at any $0 \leq k \leq K$. Thus, $p(k)$ is equally distributed $p(k) = 1/(K+1), 0 \leq k \leq K$. The overall synchronization success probability in (6.33) becomes

$$p_{sync} = \frac{1}{K+1} \sum_{k=0}^{K} p_{sync}(k), \tag{6.40}$$

and the best syncword is one that maximizes

$$\hat{\boldsymbol{s}} = \arg\max_{\boldsymbol{s}} (p_{sync}) = \arg\max_{\boldsymbol{s}} \sum_{k=0}^{K} p_{sync}(k). \tag{6.41}$$

In the following it shall be proven that for IUD data any bifix-free syncword $h = 0$ is the best and maximizes (6.41). Equation (6.35) implies that the probability of not observing a pattern after sliding for K steps in IUD data is the lowest for bifix-free patterns and increases with h. In other words, $1 - \sum_{d=K+1}^{\infty} p(d|\boldsymbol{s}_{h=0}) \leq 1 - \sum_{d=K+1}^{\infty} p(d|\boldsymbol{s}_{h>0})$. Thus, $\sum_{d=0}^{K} p(d|\boldsymbol{s}_{h=0}) \geq \sum_{d=0}^{K} p(d|\boldsymbol{s}_{h>0})$ and because $p(d)$ is proportional to $p_{sync}(k)$

$$\sum_{k=0}^{K} p_{sync}(k|\boldsymbol{s}_{h=0}) \geq \sum_{k=0}^{K} p_{sync}(k|\boldsymbol{s}_{h>0}). \tag{6.42}$$

Together with (6.41), this proves that for noiseless transmission of IUD data and equally likely $0 \leq k \leq K$ the bifix-free syncwords are optimal for threshold based synchronization independently of how large K is, even though the advantage diminishes for large values of K. Since $p_{sync}(k)$ is monotonically decreasing this result can be extended to all monotonically decreasing search start distributions $p(k)$. If the data is IID distributed, one would expect the bifix-free syncword which is least likely to be emulated in the data to be the best syncword choice. For noiseless transmission this would be a pattern of the form $BA \ldots A$, where A is the least and B is the second least likely symbol.

Figure 6.7 depicts p_{sync} for syncwords of length $L = 5$ with different bifix patterns h. It confirms that bifix-free syncwords $\boldsymbol{s}_{h=0}$ perform best independently of K and that the performance decreases with increasing bifix pattern indicator h. The plot demonstrates the importance of choosing a good syncword, compare the performance of bifix-free syncwords $\boldsymbol{s}_{h=0}$ to the maximum overlapping syncwords $\boldsymbol{s}_{h=15}$. The approximation $\tilde{p}_{sync}$ assuming independent $L(\mu)$ overestimates the true p_{sync} for syncwords with a small degree of self-overlap and greatly underestimates the true performance in case of a high degree of self-overlap.

Discrete Noisy Channels

In case of transmission over discrete noisy channels the threshold synchronizer recognizes all patterns $\mathcal{S}' = \{\boldsymbol{s}'_1 \ldots \boldsymbol{s}'_N\}$ for which the likelihood function $L(\boldsymbol{s}') \geq \lambda$.

In [BSV05] the PDF of the search for the first occurrence of a single pattern (6.37) in IID data has been generalized to the search for the first occurrence of any pattern from a set of patterns

$$p_{\mathcal{S}'}(d) = \sum_{i=1}^{N} p_{\mathcal{S}'}(d, \boldsymbol{s}'_i), \tag{6.43}$$

where $p_{\mathcal{S}'}(d, \boldsymbol{s}'_i)$ refers to the probability that pattern $\boldsymbol{s}'_i$ is the first pattern from $\mathcal{S}'$ found at position d in the semi-infinite IID data stream. This probability can be expressed

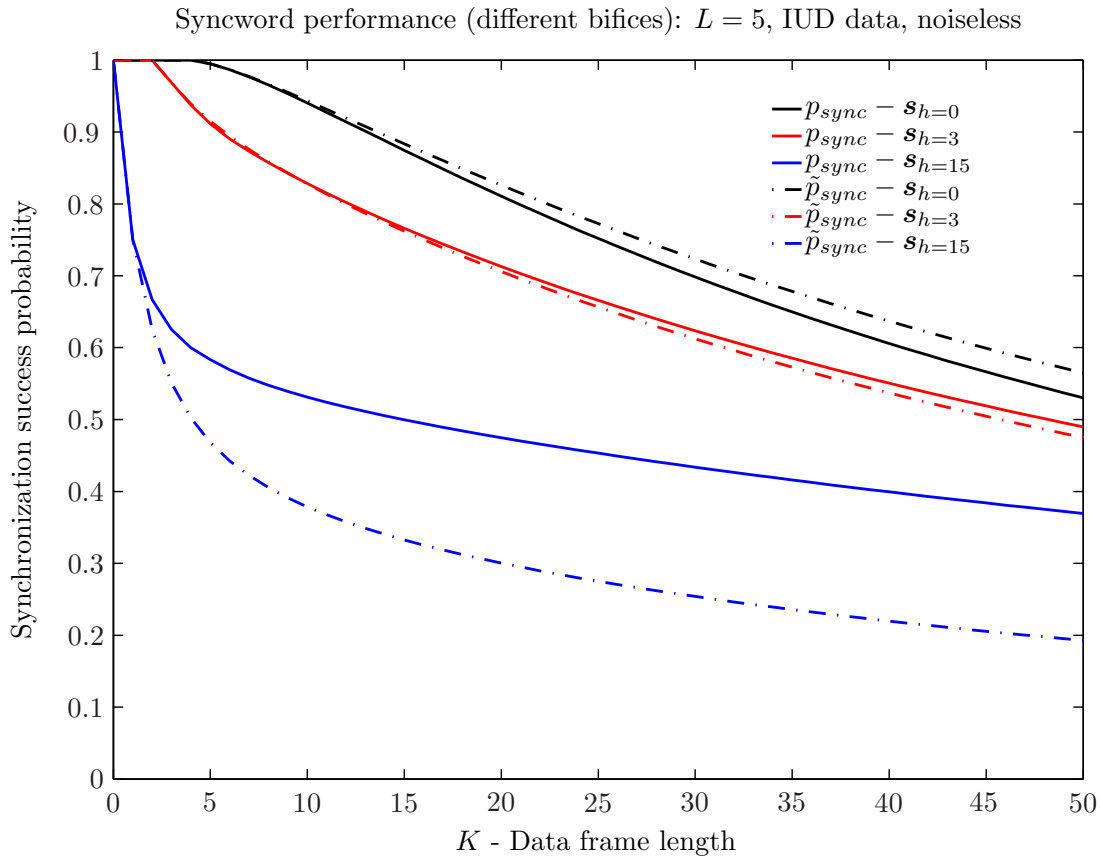


Figure 6.7: Overall exact p_{sync} and approximated $\tilde{p}_{sync}$ probability of successful synchronization as a function of data frame length K assuming that the actual starting offsets $k < K$ are equally likely (syncword length $L = 5$, IUD data, noiseless channel).

recursively

$$p_{\mathcal{S}'}(d, \boldsymbol{s}'_i) = \sum_{j=1}^{N} \sum_{l=1}^{\min(L,d)} \left(h_{ji}^{(L+1-l)} r_j^{(l-1)} - h_{ji}^{(L-l)} r_j^{(l)} \right) p_{\mathcal{S}'}(d-l, \boldsymbol{s}'_j) \qquad p_{\mathcal{S}'}(0, \boldsymbol{s}'_j) = r_j^{(L)}, \forall j, \tag{6.44}$$

where $r_j^{(l)}$ is the probability that the suffix of length l of the pattern $\boldsymbol{s}'_j$ appears in the data frame as introduced in (6.37). The factors $h_{ji}^{(l)}$ with $j, i \in \{1 \dots L\}$ are the so-called cross-bifix indicators indicating whether the suffix of $\boldsymbol{s}'_j$ and the prefix of $\boldsymbol{s}'_i$, both of length l, are identical. By definition $h_{ji}^{(0)} = 1, \forall j, i$.

Example: Let $\mathcal{S}' = \{\boldsymbol{s}'_1 = 010, \boldsymbol{s}'_2 = 100\}$, then

$$\boldsymbol{h} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} : \qquad \boldsymbol{h}^{(3)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \boldsymbol{h}^{(2)} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \boldsymbol{h}^{(1)} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, \boldsymbol{h}^{(0)} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \tag{6.45}$$

*

In the following a modified version of this recursive formula shall be used in the computation of $p_{sync}(k)$ for the threshold based synchronization model over discrete channels. Let $\boldsymbol{s}$ denote the originally used syncword and $\mathcal{S}' = \{\boldsymbol{s}'_1 \dots \boldsymbol{s}'_N\}$ be the set of all patterns of length L believed to originate from $\boldsymbol{s}$ by the threshold synchronizer $\boldsymbol{s}' \in \mathcal{S}'$ iff $L(\boldsymbol{s}') \geq \lambda$.

Given the transition probability matrix of the DMC, the probabilities $p(\boldsymbol{s}'_i|\boldsymbol{s})$ that the original syncword $\boldsymbol{s}$ is received as $\boldsymbol{s}'_i$ can be directly calculated. Thus, the synchronization success probability $p_{sync}(k)$ can be expressed as

$$\boxed{p_{sync}(k) = \sum_{\boldsymbol{s}'_i \in \mathcal{S}'} p_{\mathcal{S}'}(k, \boldsymbol{s}'_i) p(\boldsymbol{s}'_i|\boldsymbol{s}),} \tag{6.46}$$

where $p_{\mathcal{S}'}(k, \boldsymbol{s}'_i)$ is the probability that the first pattern from $\mathcal{S}'$ closest to the actual syncword insertion point is $\boldsymbol{s}'_i$ and is situated k positions away from the actual insertion point. In analogy to the approach used in the previous section for the noiseless case, the recursion in (6.44) can be adjusted for the recursive computation of the probability $p_{\mathcal{S}'}(k, \boldsymbol{s}'_i)$ by setting the initial probability to one $p_{\mathcal{S}'}(0, \boldsymbol{s}'_j) = 1, \forall \boldsymbol{s}'_j \in \mathcal{S}'$

$$\boxed{p_{\mathcal{S}'}(k, \boldsymbol{s}'_i) = \sum_{j=1}^{N} \sum_{l=1}^{\min(L,k)} \left(h_{ji}^{(L+1-l)} r_j^{(l-1)} - h_{ji}^{(L-l)} r_j^{(l)} \right) p_{\mathcal{S}'}(k-l, \boldsymbol{s}'_j) \qquad p_{\mathcal{S}'}(0, \boldsymbol{s}'_j) = 1, \forall j.} \tag{6.47}$$

The suffix probabilities $r_j^{(l)}$ must be computed according to the PMF of the IID data after transmission.

Again, the exact synchronization performance p_{sync} shall be compared to the approximation $\tilde{p}_{sync}$ computed under the assumption of independence of $L(\mu)$. For the noisy case the probability $p_{hit}(l)$ in (6.34) becomes

$$p_{hit}(l) = \sum_{i=1}^{N} p_{hit}(l|\boldsymbol{s}'_i) p(\boldsymbol{s}'_i|\boldsymbol{s}), \tag{6.48}$$

where $p_{hit}(l|\boldsymbol{s}'_i)$ is the probability of observing any pattern from $\mathcal{S}'$ exactly l symbols ahead of the syncword insertion point, given that the originally embedded syncword was received as $\boldsymbol{s}'_i$

$$p_{hit}(l|\boldsymbol{s}'_i) = \begin{cases} \sum_{j=1}^{N} h_{ji}^{(L-l)} r_j^{(l)} & \text{if } l < L \\ \sum_{j=1}^{N} r_j^{(L)} & \text{else} \end{cases}. \tag{6.49}$$

In the following, simulation results for the standard model assuming IUD data symbols, a BSC channel and equally distributed $k < K$, see (6.40), shall be presented and discussed. For the BSC the likelihood function is the negative Hamming distance between the received pattern and the syncword, see (6.25). The threshold of detection $\lambda = -d_h^{max}$ is thus equivalent to minus the maximum tolerable Hamming distance. Thus, $\mathcal{S}' = \{\boldsymbol{s}' : d_h(\boldsymbol{s}', \boldsymbol{s}) \leq d_h^{max}\}$ and the probability $p(\boldsymbol{s}'_i|\boldsymbol{s})$ can be computed as

$$p(\boldsymbol{s}'_i|\boldsymbol{s}) = p^{d_h(\boldsymbol{s}'_i, \boldsymbol{s})} \cdot (1-p)^{L - d_h(\boldsymbol{s}'_i, \boldsymbol{s})}, \tag{6.50}$$

where p is the symbol error probability of the BSC.

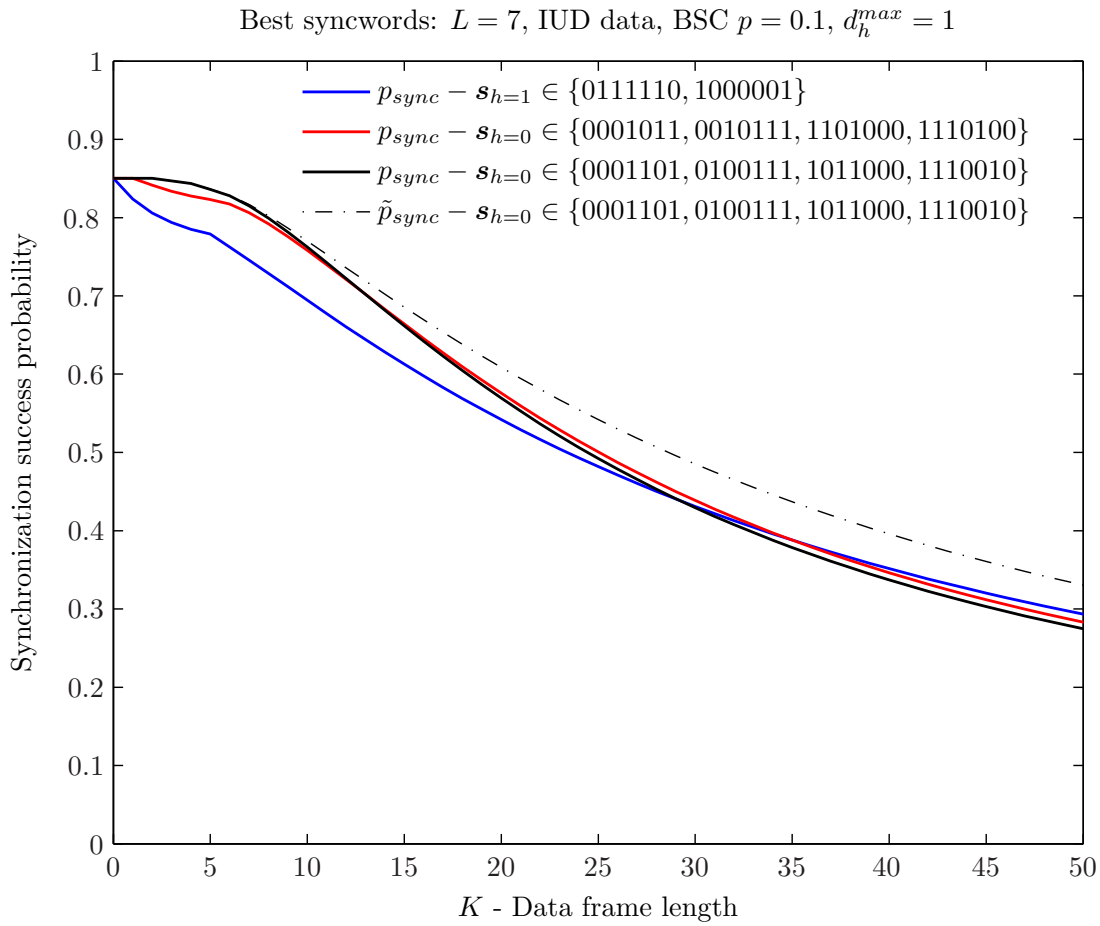


Figure 6.8: Performance comparison of the best syncwords according the exact p_{sync} and approximated $\tilde{p}_{sync}$ synchronization probability (syncword length $L = 7$, IUD data, BSC $p = 0.1$, $d_h^{max} = 1$). The approximation $\tilde{p}_{sync}$ fails to recognize the best syncwords.

Figure 6.8 shows the overall synchronization success probability for syncwords $\boldsymbol{s}$ that perform best for at least one K. IUD data and transmission over a BSC with error probability $p = 0.1$ was assumed. The syncword length was set to $L = 7$ and the detection threshold was set according to the maximum tolerable Hamming distance $d_h^{max} = 1$. The full lines represent the performance of the best syncwords according to the exact p_{sync} synchronization success probability and the dotted line represents the best syncwords according to the approximation $\tilde{p}_{sync}$ assuming independent $L(\mu)$. It can be seen that the optimal syncword depends on K and is not necessarily bifix-free. The plot also demonstrates that $\tilde{p}_{sync}$ is not optimal for choosing the best syncword. According to $\tilde{p}_{sync}$ the syncwords $\boldsymbol{s}_{(h=0)} \in \{0001101, 0100111, 1011000, 1110010\}$ would erroneously be considered optimal for any K in the plotted range. From now on, we will restrain to using the exact formula for p_{sync} in order to determine optimal syncwords.

In Figure 6.8 the maximum Hamming distance was fixed. However, the choice of proper d_h^{max} also depends on the parameter K. For example, for $d_h^{max} = 0$ the set $\mathcal{S}' = \{\boldsymbol{s}'_1 = \boldsymbol{s}\}$ contains only the original syncword. The probability that the transmitted syncword $\boldsymbol{s}$ remains unchanged is relatively low causing bad performance for small K compared to $d_h^{max} > 0$. However, the probability of emulation of a pattern recognized by the

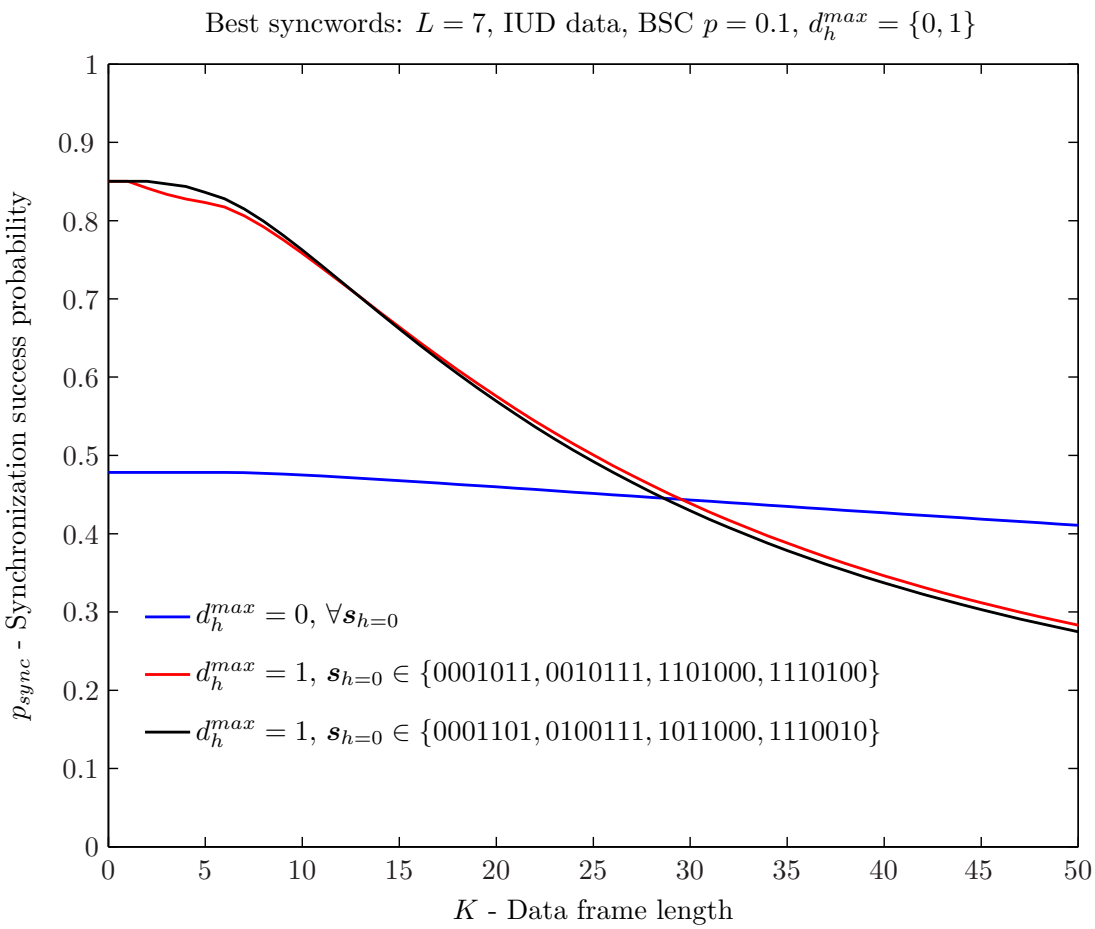


Figure 6.9: Performance comparison of the best syncwords obtained using different thresholds $d_h^{max} = \{0, 1\}$ (syncword length $L = 7$, IUD data, BSC $p = 0.1$).

synchronizer in the received data is higher for $d_h^{max} > 0$. As a result $d_h^{max} = 0$ outperforms $d_h^{max} > 0$ for larger K. Thus, the choice of best d_h^{max} as well as the syncword $\boldsymbol{s}$ depends on K. Figure 6.9 depicts the synchronization success probability p_{sync} for the best syncwords of length $L = 7$ and for $d_h^{max} = \{0, 1\}$. In comparison to Figure 6.8 the combination of ($d_h^{max} = 1$,$\boldsymbol{s}_{(h=1)} \in \{0111110, 1000001\}$) is outperformed by ($d_h^{max} = 0$,$\forall \boldsymbol{s}_{(h=0)}$) for $K > 30$. For $13 < K \leq 30$ ($d_h^{max} = 1$,$\boldsymbol{s}_{h=0} \in \{0001011, 0010111, 1101000, 1110100\}$) should be used and for $K \leq 13$ ($d_h^{max} = 1$,$\boldsymbol{s}_{h=0} \in \{0001101, 0100111, 1011000, 1110010\}$). However, $K \leq 13$ is unlikely to be used in a real system due to the amount of bits used for the syncword vs. for the data.

Note that the syncwords claimed to perform equally well are actually flipped realizations of each other in terms of flipping the bits and flipping the patterns from left to right. Consequently, all have the same bifix pattern h. For BSC and IUD data such flipped realizations are expected to perform equally well. This notion can be used to reduce the set of all $|A|^L$ syncwords that need to be compared in order to find the best performer.

6.3.2 Maximization Based Synchronization

The maximization based synchronizer evaluates the likelihood function $L(\mu)$ for each position in the received window $(y_1 \dots y_{L+K})$ of length $L+K$.Positions $\hat{\boldsymbol{\mu}} = \arg\max_{\mu} L(\mu)$ are potential syncwords. If several candidates are found, which is often the case for discrete channels, the synchronizer chooses randomly one of the candidates as $\hat{\mu}$

$$p_{sync}(\boldsymbol{s}) = \sum_{\forall \hat{\boldsymbol{\mu}}: \mu_{SW} \in \hat{\boldsymbol{\mu}}} \frac{1}{|\hat{\boldsymbol{\mu}}|} p(\hat{\boldsymbol{\mu}}), \qquad \hat{\boldsymbol{\mu}} = \arg\max_{\mu} L(\mu), \forall \mu = 1 \dots L+K, \tag{6.51}$$

where $|\hat{\boldsymbol{\mu}}|$ refers to the cardinality of the set $\hat{\boldsymbol{\mu}}$. In order to compute $p_{sync}(\boldsymbol{s})$ the probabilities of all possible sets $\hat{\boldsymbol{\mu}}$ for which $\mu_{SW} \in \hat{\boldsymbol{\mu}}$ have to be determined, which can be done for the noiseless case, but turns out to be infeasible for noisy channels.

Noiseless Transmission

For maximization based synchronization and noiseless transmission, the syncword optimization problem simplifies to finding syncwords $\boldsymbol{s}$ of length L that are least likely to appear at any position in the transmitted frames. This is obviously the case for the so called bifix-free syncwords whose prefix and suffix of any length do not overlap. Bifix-freedom of syncwords guarantees that all emulated syncword realizations in the data are non-overlapping with each other and thus independent. Additionally, the emulated copies do not overlap with the actual syncword. Thus, no more than $|\hat{\boldsymbol{\mu}}| \leq \lfloor K/L \rfloor$ copies of the syncword can be found in the data portion of the frame. These emulated copies are the sole source of error. The synchronization success probability for bifix-free syncwords $\boldsymbol{s}_{h=0}$ can be calculated. For IUD data this has been accomplished using combinatorics [Nie73b]

$$p_{sync}(\boldsymbol{s}_{h=0}) = 1 - \sum_{i=1}^{\lfloor K/L \rfloor} \frac{(-1)^{i+1}}{i+1} \binom{K-(L-1)i}{i} |\mathcal{A}|^{-Li}, \tag{6.52}$$

where $|\mathcal{A}|$ refers to the cardinality of the used alphabet. In [Rob95] it has been pointed out that increasing L has a large positive impact on performance, while increasing K degrades the performance only slightly for moderate and large K. In case the syncwords are self-overlapping $\boldsymbol{s}_{h>0}$ (not bifix-free) the exact synchronization success probability $p_{sync}(\boldsymbol{s}_{h>0})$ cannot be easily computed due to the dependencies between possible overlapping syncword instances. However, since in addition to the equally likely non-overlapping syncword emulation, self-overlapping emulation can occur for not bifix-free sequences, they will qualitatively perform worse $p_{sync}(\boldsymbol{s}_{h>0}) > p_{sync}(\boldsymbol{s}_{h=0})$. For IUD data all bifix-free syncwords perform equally good. The result can be extended to IID data. Again, the bifix-free pattern that is least likely to occur in the IID data can be expected to be the best syncword choice. It is going to be of the form $A \dots AB$, where A is the least and B the second least likely symbol.

Having proven that for noiseless transmission bifix-free syncwords are optimal for threshold and maximization based synchronization, one could ask how many bifix-free syncwords

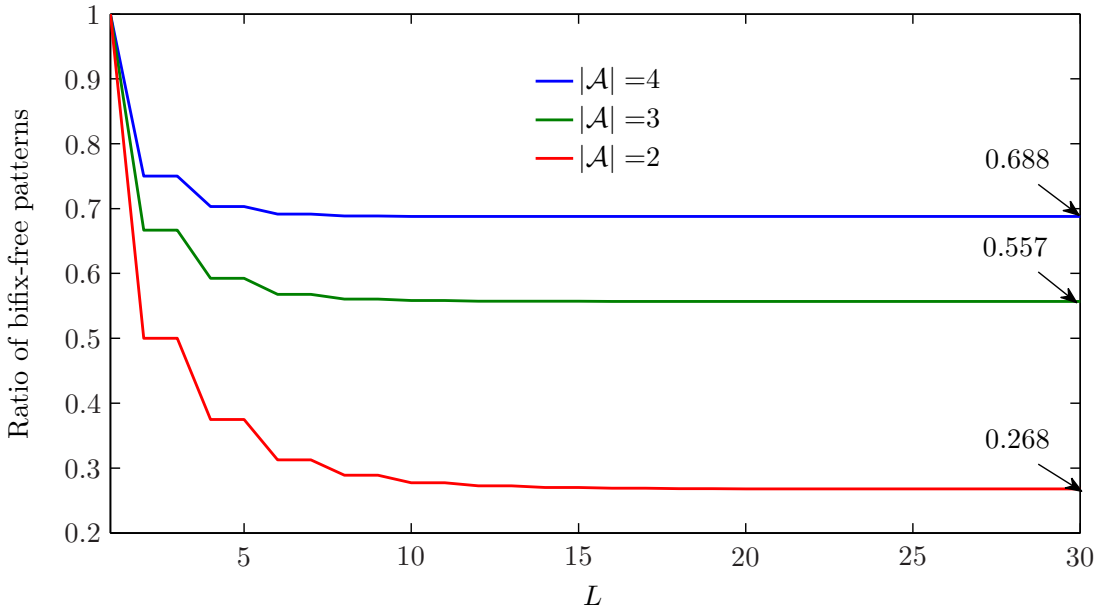


Figure 6.10: Ratio of bifix-free sequences of length L from Alphabet $\mathcal{A}$.

of certain length $N(L)$ actually exist for a given alphabet $\mathcal{A}$. In [GMS02] a recursive formula was derived for the binary alphabet. The extension to higher order alphabets is straight forward and leads to

$$N(L) = \begin{cases} |\mathcal{A}| & \text{if } L = 1 \\ |\mathcal{A}| \cdot N(L-1) - N(L/2) & \text{if } L \text{ even} \\ |\mathcal{A}| \cdot N(L-1) & \text{else} \end{cases} . \tag{6.53}$$

The ratio of bifix free sequences converges quickly against a fixed value as shown in Figure 6.10.

Noisy Channel

For noisy channels computing the synchronization success probability $p_{sync}(\boldsymbol{s})$ precisely becomes intractable for the maximization based synchronization. This is also the case for discrete channels including the simple BSC channel. In the literature the question of syncword choice for maximization based synchronization is typically addressed under the assumption of independent $L(\mu)$, which allows to reduce the problem to finding the syncword maximizing some function related to the synchronization success probability in the portion where data and syncword overlap. Usually, the side lobes of the autocorrelation function (ACF) $R_{\boldsymbol{s}}$ are combined with a min-max or min-average method to select the best candidate. Barker has proposed the use of syncwords with autocorrelation side lobes of magnitude smaller than one $|R_{\boldsymbol{s}}(\mu)| \leq 1, \forall \mu : 1 \leq \mu \leq L-1$ in his pioneering work from 1953 [Bar53]. They exist only for lengths $L = 2, 3, 4, 5, 7, 11, 13$ and are particularly suitable for noisy channels with phase ambiguities, which can occur when using modulation schemes such as BPSK. The term phase ambiguity refers to the fact that the receiver cannot differentiate between the possible mappings of the original symbols to the received sequence (e.g the receiver cannot distinguish between 00101 and 11010). Barker's findings were extended to sequences of other lengths using a min-average rule on the side lobes of

the ACF $|R_{\boldsymbol{s}}|$ in [Wil62, MS64]. A min-max rule was used in [Boe67]. In 1971, Neuman and Hoffman noticed that not only the amplitudes of the sidelobes of the ACF play an important role, but also their exact position [NH71]. It can be shown, that this position dependency is actually related to the degree of the bifices. In [Sch80] the approximate synchronization success probability $\tilde{p}_{sync}(\boldsymbol{s})$ of threshold based synchronization under the assumption of independent $L(\mu)$ was used to search for good syncwords for maximization synchronization[3]. In [LT87] the upper bound on $\tilde{p}_{sync}(\boldsymbol{s})$ for Gaussian channels assuming independent $L(\mu)$ was introduced. In [Rob95] the result was generalized to arbitrary phase shift keying (PSK) modulation schemes and used to search for optimal syncwords according to the bound. Possibly, as an extension of this work, the results presented in [Sch80, LT87, Rob95] can be revised using the exact formula for $p_{sync}(\boldsymbol{s})$ introduced in Section 6.3.

6.4 Summary

Marker synchronization in engineering has been studied in detail in this chapter. The main motivation was that sequence specific binding on the molecular level shows strong parallels to marker synchronization as will be detailed in Chapter 7. The derivations and analysis were conducted such that the results are easily applicable to the biological scenario.

In engineering marker synchronization refers to the recovery of insertion positions of short sequence markers artificially introduced into the transmitted data stream to tag certain positions, e.g. the borders of transmitted data frames. At the receiver the synchronizer evaluates for each position in the received data stream a likelihood function conveying the likelihood that the position is an actual marker insertion point. In this thesis a general optimal likelihood function was derived. Under the assumption of a memoryless, time-invariant channel and IID data the optimal likelihood function only depends on the observation in a detection window of the size of the syncword and corresponds to the LLR between the hypothesis that the observation originates from a syncword insertion and the hypothesis that the observation was generated by the data. It has been shown that using a single fixed marker as syncword at the transmitter is optimal and that the LLR corresponds to the mutual self-information between the received sequence in the detection window and the syncword.

An optimal likelihood function and approximations thereof for the standard model assuming an AWGN channel, BPSK modulation and IUD data have been originally proposed by Massey in [Mas72]. Equivalence between the derived LLR and Massey's proposal has been established. Using the self-information a new approximation to the optimal likelihood function superior to the approximations proposed by Massey was derived. The LLR was adapted to discrete channel models including the evolutionary substitution channel models required for the analysis of the biological scenario.

[3]Please note that [Sch80] uses combinatorics based notation for $\tilde{p}_{sync}(\boldsymbol{s})$ as opposed to the bifix notation used throughout this work. In addition, $\tilde{p}_{sync}(\boldsymbol{s})$ is referred to as first-pass acquisition probability.

Subsequently the question of assessing the synchronization performance of different marker patterns has been addressed. Focus was placed on threshold based marker synchronization over discrete memoryless channels since this setting closely resembles the biological scenario. The threshold synchronizer declares synchronization for positions where the likelihood function attains values above a certain threshold. The synchronization performance was found to depend on the self-overlap pattern of the syncword with itself. It was proven that for noiseless transmission and threshold based detection the overlap-free syncwords are always optimal. For noisy transmission the synchronization success probability depends in addition to the self-overlap also on the cross-overlap to all other patterns recognized by the synchronizer. For the noisy case the choice of optimal syncword cannot be answered in general, but depends on the SNR, the frame length and the chosen threshold value.

The exact computation of the synchronization success probability is a difficult issue since the values of the likelihood function are not independent for neighbouring positions. Herein, a recursive formula for the exact computation of the synchronization success probability has been derived and compared with the approximation assuming position-wise independence of the likelihood values commonly used in the literature. It was found that the approximation tends to greatly overestimate the synchronization performance of syncwords with a small degree of self-overlap and underestimate the performance of syncwords with a high degree of self-overlap. Additionally, the approximation was demonstrated to be incapable to correctly determine the best syncword.

7

Marker Synchronization in Genetics

In this chapter parallels between marker synchronization in engineering and genetics shall be investigated. As described in Chapter 3, the genetic information is distributed all over the genome. Often, sequence specific markers are used by the molecular machinery to distinguish information carrying regions. Marker synchronization is utilized across different layers of genetic information processing including transcription, translation and splicing. Specific sequence markers are also used by regulatory proteins that need to bind at specific binding sites in the genome. These sites are subject to evolutionary mutations and the molecular machinery allows for a certain degree of variation in the recognized sequence patterns. Binding site recognition strongly resembles threshold based marker synchronization over a noisy channel.

In Section 7.1 established quantitative models for binding sites are presented and analysed using an information theoretic framework. In particular, it is accounted for the limited sample size of experimentally verified binding sites. Section 7.2 addresses the in silico inference of novel putative binding sites from already known and verified ones. Parallels to threshold based synchronization are established and an extension to the current approach is proposed. In Section 7.3 current knowledge about the in vivo recognition of binding sites by the molecular machinery is presented. Selected markers utilized in prokaryotic transcription are studied in terms of their synchronization performance using the framework derived in Chapter 7.

7.1 Sequence Specific Binding

Sequence specific binding takes place in the DNA as well as the RNA domain. In general, for RNA binding sites the sequence specific binding follows the same principle as the complementary nucleotide base pairing, see Section 3.2.1. DNA binding sites are usually directly bound by proteins and the binding mechanism is more complex. In the following, the focus will be placed on protein DNA binding. However, the introduced quantitative models are also applicable to RNA binding sites. In Section 7.1.1 the basic aspects of protein DNA binding are briefly introduced. Established statistical binding site models are discussed in Section 7.1.2 from an information theoretic point of view. Particular focus is placed on the fact that the sample of actually experimentally verified sites is very small for most proteins. A commonly used assumption is that the nucleotides in a binding site contribute to the binding independently. In Section 7.1.3 mutual information is used to study positionwise dependencies in binding sites.

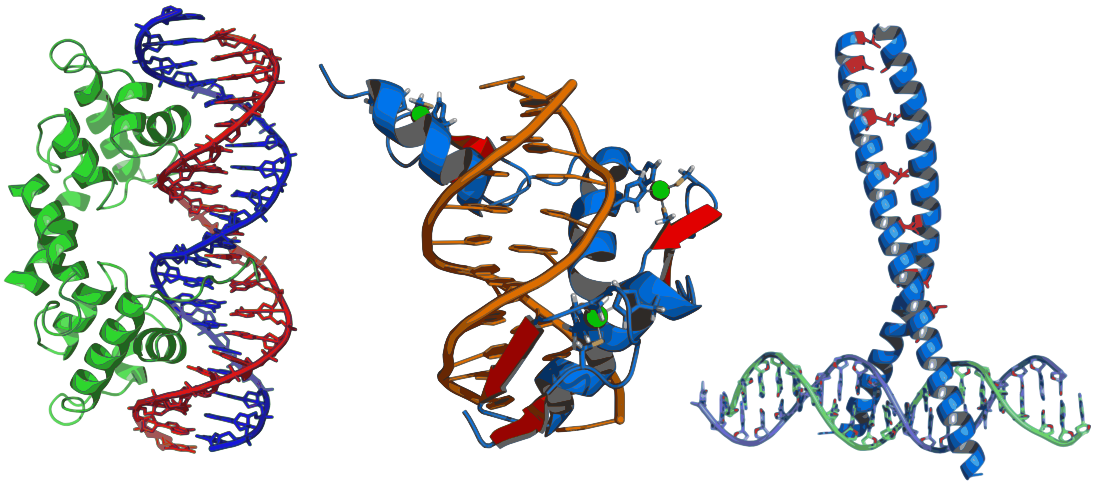
7.1.1 Protein DNA Binding

The DNA binding proteins include transcription factors which modulate the process of transcription, nucleases which cleave DNA molecules, and histones which are involved in DNA packaging in the cell nucleus. The amino acid residues of a DNA binding protein involved in the binding are commonly referred to as the binding domain. A protein's DNA binding domain is usually composed of a recognition and a stabilization region. The recognition of DNA by the protein takes place at two levels. The non-specific binding between the protein side-chains and the DNA sugar/phosphate backbone is responsible for attaching to the DNA double helix and is independent of the nucleotide composition. Sequence specific binding takes place between the protein side-chains and the nucleotide bases. Sequence specificity is generally stronger in the portion of the DNA binding domain attaching to the major groove of the DNA which relates to the fact that the base pairs are more exposed in this region, see Section 3.2.1.

The DNA binding domains of different proteins can be characterized according to the protein folding pattern they form. Three common DNA binding domains involved in transcription regulation are the helix-turn-helix, the zinc finger, and the leucine zipper domain depicted in Figure 7.1. It can be seen that the base specific interactions take place primarily in the DNA major groove and are usually realized by the protein α-helices, see also Section 3.2.3.

7.1.2 Binding Motifs

A set of aligned experimentally verified binding sites bound by the same protein in vivo is commonly referred to as the binding motif. In the following, established description models for binding motifs shall be presented and analysed.



helix-turn-helix zinc finger leucine zipper

Figure 7.1: Common DNA binding domains. Depicted are the helix-turn-helix, the zinc finger, and the leucine zipper domain. Modified from [Wik08].

Consensus Sequence

The simplest description of a binding motif is the consensus sequence. The consensus sequence assigns each column in the alignment a letter, which can be either the most frequent nucleotide (majority vote) or a representative description of the group of nucleotides observed, e.g. `N` for any nucleotide or `R` for purines. A consensus sequence allows for easy visual comparisons. However, it is only a qualitative description of a binding motif distorting the quantitative properties.

Position Frequency Matrix

A more accurate description of binding motifs is the position frequency matrix (PFM) corresponding to a nucleotide frequency count for each column in the binding motif. A normalized PFM approximates the positionwise PMF $P_{X_i}(x), x \in \mathcal{X}, i = 1 \dots L$ of the binding motif, where L is the length of the motif and $\mathcal{X} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$ is the nucleotide alphabet. This description presumes positionwise independence in the binding motif. In biological terms, the nucleotides are assumed to contribute to the binding independently.

Sequence Logo

An easily interpretable visualization of the PFMs are the sequence logos introduced in [SS90]. A sequence logo depicts the nucleotides occurring in each column vertically scaled and sorted according to their relative frequency of occurrence in the respective binding motif column. In other words, the nucleotides in a column are scaled and stacked in the order of importance with the most frequent nucleotides placed on top. Most frequent nucleotides appear on top of each column and are equivalent to the consensus sequence. The height of each column stack is scaled by its information content defined as

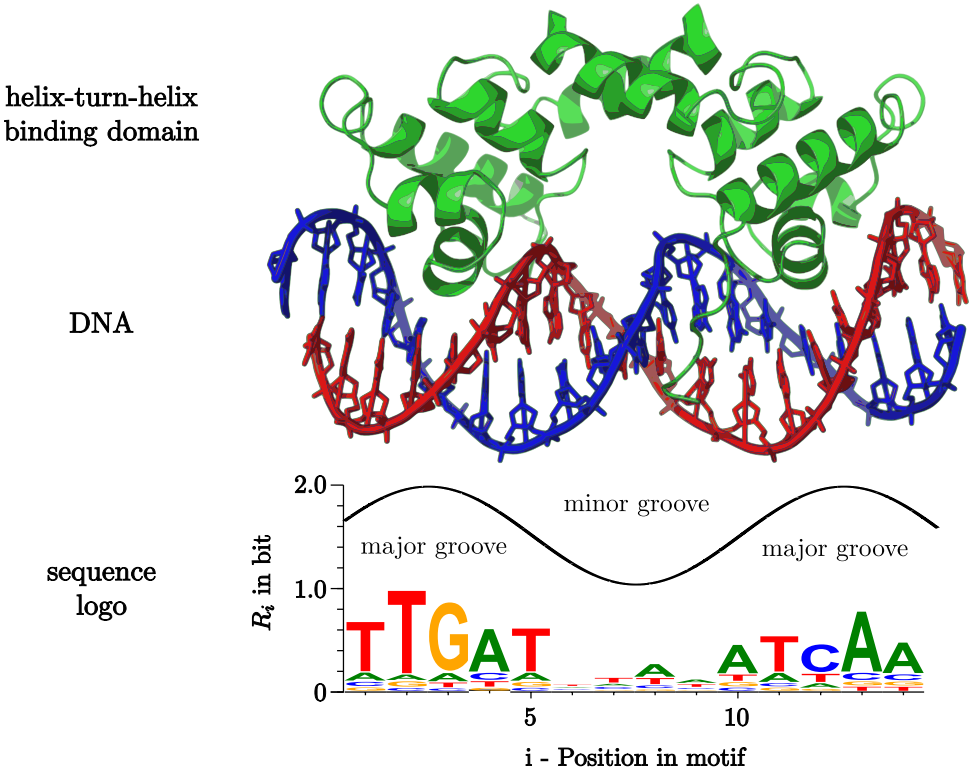


Figure 7.2: The DNA (red/blue) bound helix-turn-helix binding domain (green) of the transcription factor FNR is depicted together with the sequence logo, position frequency matrix and the consensus sequence of the corresponding binding motif derived from 74 annotated FNR binding sites in E. coli.

$$R_i = H_\pi - H(X_i), \tag{7.1}$$

where H_π is the background entropy of the genome and $H(X_i)$ is the entropy of the i-th column in the binding motif [SSGE86]. Under the assumption that the genome is IUD, the information content can be computed as

$$R_i = \mathrm{ld}\,|\mathcal{X}| - H(X_i) = 2 + \sum_{x \in \{\mathcal{X}\}} P_{X_i}(x)\,\mathrm{ld}\,P_{X_i}(x). \tag{7.2}$$

Consequently, R_i is in the range of $0 < R_i < 2$ bit. Loosely speaking, the column height measures the conservation of a column in bits, whereas $R = 2$ bits indicates absolute conservation and $R = 0$ bits no conservation.

Example: The sequence logo, the PFM and the consensus sequence of the binding motif of the transcription factor protein (FNR) is depicted in Figure 7.2. The bind-

ing motif was extracted from the 74 E. coli binding sites annotated in the Regulon database [GCJJPG+08]. In Section 3.2.1 it has been pointed out that due to the wound double helix structure of the DNA, the sequence specific binding takes place primarily in the major groove region where the bases are more accessible to the binding protein. As a consequence, the conservation pressure on each position in a consecutive binding motif is influenced by its accessibility. Roughly speaking the protein can distinguish between all 4 nucleotides in the center of the major groove, since it can reliably recognize the base pair type $\mathtt{A}-\mathtt{T}$ or $\mathtt{C}-\mathtt{G}$ and its orientation $\mathtt{A}-\mathtt{T}$ vs. $\mathtt{T}-\mathtt{A}$ and $\mathtt{C}-\mathtt{G}$ vs. $\mathtt{G}-\mathtt{C}$. However, in the minor groove distinguishing between the different orientations is difficult and the protein generally only distinguishes between the 2 base pair types [SRR76]. As a result, in the minor groove $0 < R_i < 1$ bit. [1] In Figure 7.2, this property is reflected by the sine like wave with a periodicity of 10.4 bp corresponding to the DNA double helix turn periodicity [SBS93]. The FNR protein has a helix-turn-helix binding domain. It is a transcriptional dual regulator [LG76]. ∗

Small Sample Size Correction

A binding motif typically comprises only a small amount of actually verified binding sites. The columnwise PMFs $P_{X_i}(x)$, thus have to be estimated from a limited sample size using frequency counts $\hat{P}_{X_i}(x) = f_i(x)/N$, where $f_i(x)$ is the count and N the number of binding sites in the binding motif. In [Mil55, Bas59] it has been shown that the frequency count based estimate of entropy $\hat{H}(X)$ is a biased, asymptotically normal estimate of the real entropy $H(X)$. Using Taylor series expansion of $\hat{H}(X)$ around P_X an approximation for the expectation of the sampled uncertainty can be derived:

$$E\{\hat{H}(X)\} = H(X) - \frac{|\mathcal{X}|-1}{2N\ln 2} + O\left(\frac{1}{N^2}\right), \tag{7.3}$$

where $|\mathcal{X}|$ is the cardinality of the alphabet. In [AK01] it has been proven that the frequency based entropy estimate always underestimates the true entropy $E\{\hat{H}(X)\} \leq H(X), \forall P_X$. Thus, for large enough $N >> |\mathcal{X}|$, a correction term can be used to improve the estimate of a columns entropy

$$H(X) \approx \hat{H}(X) + \underbrace{\frac{|\mathcal{X}|-1}{2N\ln 2}}_{e(N)}, \qquad N >> |\mathcal{X}|. \tag{7.4}$$

Taking the limited sample size into account, the information content in (7.1), used to scale each column in a sequence logo, should be approximated as

$$R_i = H_\pi - H(X_i) \approx H_\pi - \left(\hat{H}(X_i) + e(N)\right), \tag{7.5}$$

where $e(N)$ is the correction term from (7.4).

[1] This could be interpreted as fading on the channel or varying rate heterogeneity parameter.

7.1.3 Positionwise Independence of Binding Sites

The presented models for the description of binding sites assume that the nucleotides contribute to the binding independently. In order to test whether this assumption holds, pairwise dependencies between the binding motif columns shall be studied using mutual information. Restricting to pairwise dependencies is justified by the small sample size due to low number of experimentally verified binding sites. The true mutual information $I(X;Y)$ has to be determined from a frequency count based estimate $\hat{I}(X;Y)$. Using Taylor series expansion around the independence point , the following approximation is obtained for mutual information

$$I(X;Y) \approx \hat{I}(X;Y) - \frac{(|\mathcal{X}|-1)(|\mathcal{Y}|-1)}{2N\ln 2}, \qquad N >> |\mathcal{X}| \cdot |\mathcal{Y}|. \tag{7.6}$$

In [HDG+04, GDHM05] it has been shown that for independent random variables X, Y the mutual information $\hat{I}(X;Y)$ approximately follows a gamma distribution

$$\hat{I}(X;Y) \sim \Gamma\left(\frac{1}{2}(|\mathcal{X}|-1)(|\mathcal{Y}-1|), \frac{1}{N\ln 2}\right). \tag{7.7}$$

This observation can be used to test for statistical independence between X and Y. Based on the chosen significance level α (typically α=0.05), the significance of the observed value of $I(X;Y)$ can be assessed by comparison with the corresponding quantile

$$c = \Gamma_{1-\alpha}\left(\frac{1}{2}(|\mathcal{X}|-1)(|\mathcal{Y}-1|), \frac{1}{N\ln 2}\right), \tag{7.8}$$

where c stands for the critical value. If $I(X;Y) > c$ the random variables X and Y are dependent with a $1-\alpha$ probability[2]. The proposed mutual information based independence test was used to study dependencies between binding site positions of E. coli transcription factors annotated in the Regulon database [GCJJPG+08]. In accordance with [TO07] we found that some factors show evidence of dependencies while others do not. For example, Figure 7.3 shows the mutual information $I(X;Y)$ between different positions of the binding motif of the transcription factor FNR, whose sequence logo is depicted in Figure 7.2. The binding motif comprises $N = 74$ annotated binding sites. The diagonal corresponds to neighbouring positions in the binding motif. The distance to the diagonal is proportional to the distance of the columns in the motif. The significance level was set to $\alpha = 5\%$ which yields a critical value of $c = 0.16$ bits according to (7.8). The motif columns showing dependencies above the critical value are mostly neighbouring columns. This was also observed for other transcription factors showing any significant interposition dependence. The proximal dependencies are believed to relate directly to the 3-D structure of the DNA binding protein domain. A possible explanation for distant dependencies are conformational changes in the DNA structure. The presence of a particular base at a certain position might alter the accessibility of the base at the dependent position. Note that for some DNA binding proteins dependencies between neighbouring nucleotide positions have also been validated experimentally [MS01, BHCC01].

[2] In [DGH+06, SGD+07] we have successfully applied this significance test to gene mapping of complex diseases

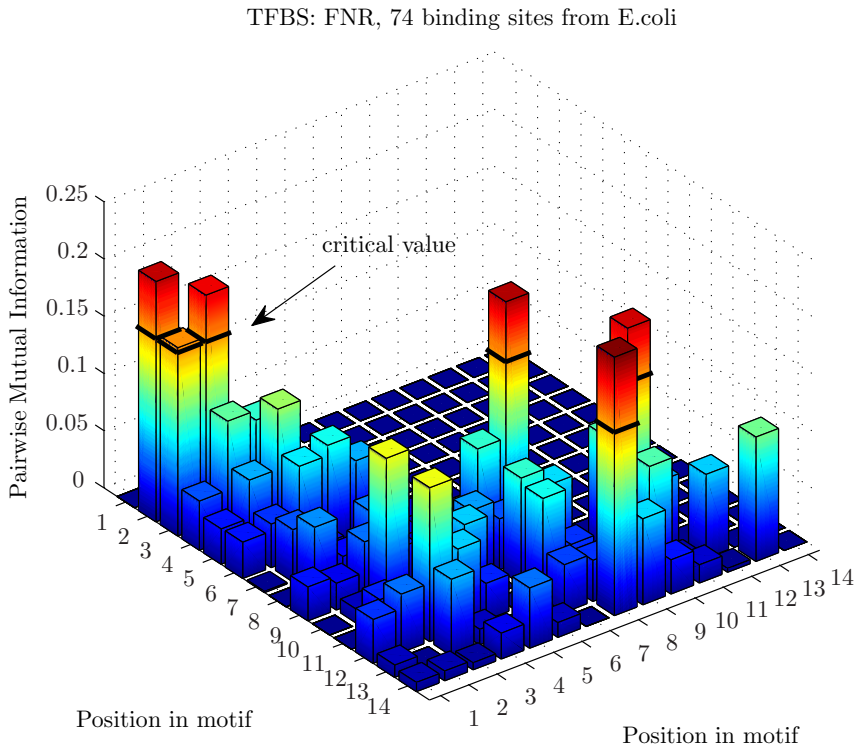


Figure 7.3: Mutual information between different positions of the FNR binding motif. The motif comprises 74 annotated binding sites. Positions above the critical value are dependent with a probability of 95%.

7.2 Binding Site Inference and Synchronization

An important field of bioinformatics research motivated by the lack of experimentally verified sites is in silico binding site inference of novel putative binding sites from already known and verified ones. Binding site inference uses a scheme very similar to threshold based marker detection described in Section 6.1. The genome is scanned for positions $\hat{\mu}$ for which the likelihood function $L(\mu)$ that the position μ is a binding site lies above a certain threshold $\hat{\mu} = \{\mu : L(\mu) > \lambda\}$, see Figure 7.4. The likelihood function used in binding site inference will be presented in Section 7.2.1. It is based on binding motif data and proportional to the binding energy. In Section 7.2.2 it is compared to the likelihood function used in synchronization and shown to correspond to a LLR of two competing hypotheses. This finding is used to evaluate the discrimination information contained in the binding motifs of different proteins. In Section 7.2.3 discrimination information is used to propose an extension to the current binding site inference scheme.

7.2.1 Likelihood Function for Binding Site Inference

Current binding site inference algorithms conduct the search for novel putative binding sites using a sliding window of size L corresponding to length of the binding motif. In analogy to Section 6.2 the pattern observed in the sliding window shall be denoted $\boldsymbol{r} = (r_1, r_2 \dots r_L)$. Although, it has been found in the previous section that some binding

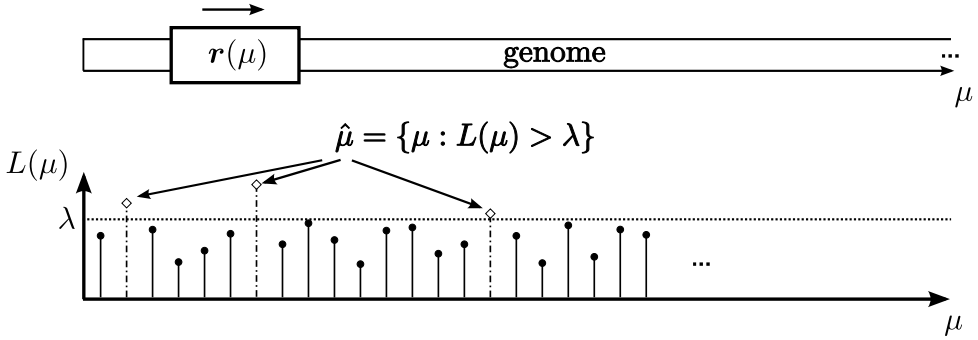


Figure 7.4: In silico inference of putative binding sites is analogous to threshold based syncword detection.

motifs show significant positionwise dependencies, additivity in protein DNA interactions will be assumed in the following. In [BBS02] it has been shown that this assumption is adequate in the context of binding site inference. In binding site inference the likelihood function $L_{BSI}(\boldsymbol{r})$ that the sequence $\boldsymbol{r}$ is a binding site is usually computed as

$$L_{BSI}(\boldsymbol{r}) = \sum_{i=1}^{L} \operatorname{ld} \frac{P_{X_i}(r_i)}{\pi(r_i)}. \tag{7.9}$$

where P_{X_i} is the PMF of the ith binding motif column and $\boldsymbol{\pi} = (\pi_{\mathtt{A}}, \pi_{\mathtt{C}}, \pi_{\mathtt{G}}, \pi_{\mathtt{T}})$ is the background distribution of the genome [WS04].

Relationship to Binding Energy

It has been shown in [SF98] that the maximum likelihood estimate for the binding energy of sequence $\boldsymbol{r}$ given only the binding motif is

$$E(\boldsymbol{r}) = -\sum_{i=1}^{L} \ln \frac{P_{X_i}(r_i)}{\pi(r_i)}, \tag{7.10}$$

and is proportional to the likelihood function L_{BSI}

$$L_{BSI}(\boldsymbol{r}) = -\ln 2 \cdot E(\boldsymbol{r}). \tag{7.11}$$

Pseudocounts

The true columnwise distributions P_{X_i} in the binding motif have to be estimated from positionwise frequency counts $f_i(x)$. Pseudocounts c are added to the frequency counts in order to compensate for the typically small amount of actually verified binding sites N comprising the binding motif.

$$P_{X_i}(x) \approx \hat{P}_{X_i}(x) = \frac{f_i(x) + c \cdot \pi(x)}{N + c}. \tag{7.12}$$

Using pseudocounts compensates for rarely occurring nucleotides in a binding motif column that might be absent from a small sample by chance. Additionally, the technical

issues related to taking a logarithm of zero when computing the weights are omitted. Although the use of pseudocounts has become ubiquitous, there is no standard way of choosing them. Often, the pseudocount is set to $c = \sqrt{N}$ as proposed in [LAB+93]. However, in a recent systematic study based on actual transcription factor binding motifs from the TRANSFAC database [WDKK96] it has been suggested that using $c = 0.8$ is preferable [NFN08]. Interestingly, this value is very close to the pseudocount used by the Krichevsky Trofimov probability distribution estimator presented earlier in Section 4.2.1 in the context of adaptive compression.

7.2.2 Comparison to the Synchronization Likelihood Function

The likelihood function L_{BSI} for binding site inference in (7.9) can similarly to the optimal likelihood function for marker synchronization in (6.5) be interpreted as a LLR

$$L_{BSI} = \sum_{i=1}^{L} \mathrm{ld}\, \frac{P_{X_i}(r_i)}{\pi(r_i)} = \sum_{i=1}^{L} \mathrm{ld}\, \frac{P(r_i|\mathcal{H}_0)}{P(r_i|\mathcal{H}_1)} = \mathrm{ld}\, \frac{P(\boldsymbol{r}|\mathcal{H}_0)}{P(\boldsymbol{r}|\mathcal{H}_1)}. \tag{7.13}$$

Assuming the general evolutionary substitution channel in (6.26), the alternative hypothesis $\mathcal{H}_1$ is the same as for synchronization, namely that the observed sequence $\boldsymbol{r}$ originates from the IID genome background process. However, care needs to be taken with respect to the underlying hypothesis $\mathcal{H}_0$. In the context of binding site inference, $p(\boldsymbol{r}|\mathcal{H}_0)$ denotes the likelihood that $\boldsymbol{r}$ is a sequence recognized by the binding protein. In the engineering terminology this would represent the likelihood that $\boldsymbol{r}$ is a sequence recognized by the synchronizer. However, the $\mathcal{H}_0$ hypothesis used in the optimal likelihood function for marker synchronization as derived in Chapter 6, see (6.5), is that $\boldsymbol{r}$ is the syncword transmitted over the channel. In other words, in engineering the synchronizer decision is based on the a-posteriori probability of the received syncword given the knowledge about the channel. In binding site inference the decision is based on the probability that the observed pattern is recognized by the binding protein given the binding motif.

Binding Motifs and Discrimination Information

Using the notion that the likelihood function L_{BSI} for binding site inference can be interpreted as a LLR of two hypothesis, the Kullback-Leibler divergence $D(P_{\boldsymbol{X}}||\boldsymbol{\pi})$ can be used to measure the discrimination information contained in the binding motif of a protein

$$D(P_{\boldsymbol{X}}||\boldsymbol{\pi}) = \sum_{i=1}^{L} D(P_{X_i}||\boldsymbol{\pi}), \tag{7.14}$$

and used to compare the binding specificity of different proteins. Note that $D(P_{X_i}||\boldsymbol{\pi})$ has to be estimated from frequency counts $\hat{P}_{X_i} = f_i(x_i)/N$. Using Taylor series expansion we arrive at the following approximation for large enough N

$$D(P_{X_i}||\boldsymbol{\pi}) \approx D(\hat{P}_{X_i}||\boldsymbol{\pi}) - \frac{|\mathcal{X}|-1}{2N \ln 2}, \qquad N >> |\mathcal{X}|. \tag{7.15}$$

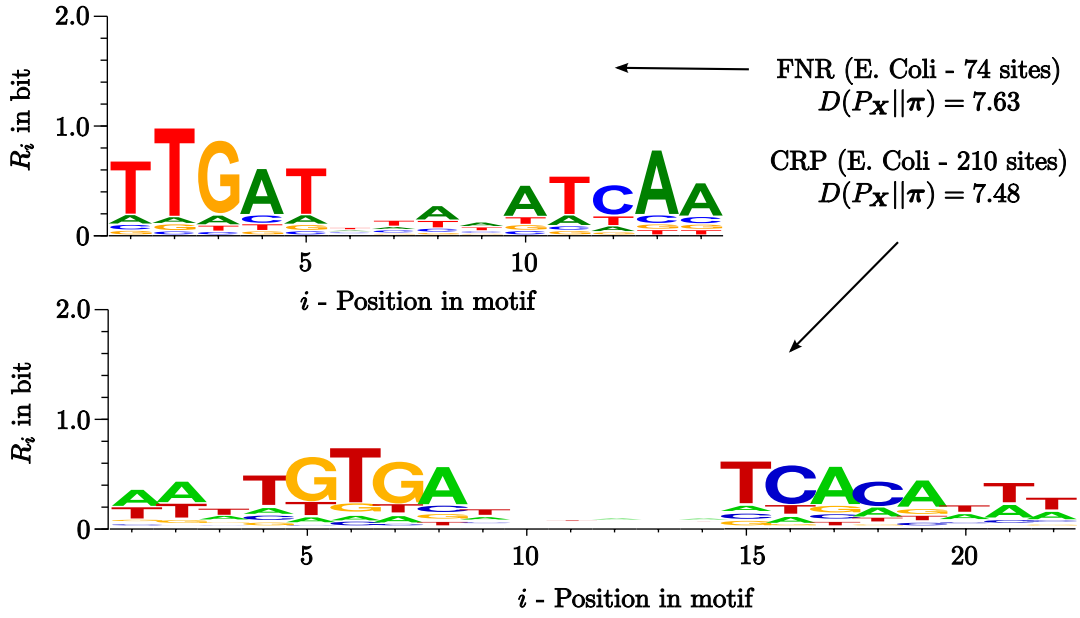


Figure 7.5: Sequence logos and discrimination information of the binding motifs of the E. coli transcription factors FNR and CRP. The discrimination information is lower for CRP although the motif is longer.

A shorter binding motif with high specificity can have a higher overall discrimination information than a highly variable longer motif. Note that for IUD genomes the discrimination information becomes identical to the information content in (7.2) used to scale the sequence logo columns. Thus, for IUD genomes the height of each column in a sequence logo corresponds to the specificity of the respective motif position.

Example: Figure 7.5 depicts the sequence logos and discrimination information of the binding motifs of the E. coli transcription factor protein (CRP) and (FNR). The motifs were obtained from the Regulon database [GCJJPG+08]. The genome distribution of E. coli is $\boldsymbol{\pi} = (\pi_{\mathtt{A}}, \pi_{\mathtt{C}}, \pi_{\mathtt{G}}, \pi_{\mathtt{T}}) = (24.6\%, 25.4\%, 25.4\%, 24.6\%)$, which is almost IUD. Note that there exist many species with a strongly biased nucleotide composition. Although the FNR motif is shorter, it shows a higher specificity. ∗

7.2.3 Vicinity Extended Binding Site Inference

Current binding site inference methods restrict the computation of L_{BSI} in (7.13) to the binding motif [Sto00]. This is only optimal under the assumption that the vicinity of binding sites is IID distributed according to the genome background distribution. However, this is often not true. For example, the vicinity of transcription start sites and splice sites, where most of the regulatory proteins bind, has been shown to experience a strong nucleotide composition bias [TAACT04]. The bias is believed to partially originate from transcription coupled DNA repair mechanisms discussed in Section 3.5.1. The nucleotide composition around E. coli transcription start sites annotated in the Regulon database [GCJJPG+08] is depicted in Figure 7.6. There is a nucleotide bias towards the weak `A` and `T` nucleotides. Note that the strong bias around -35 and -10 base pairs ahead

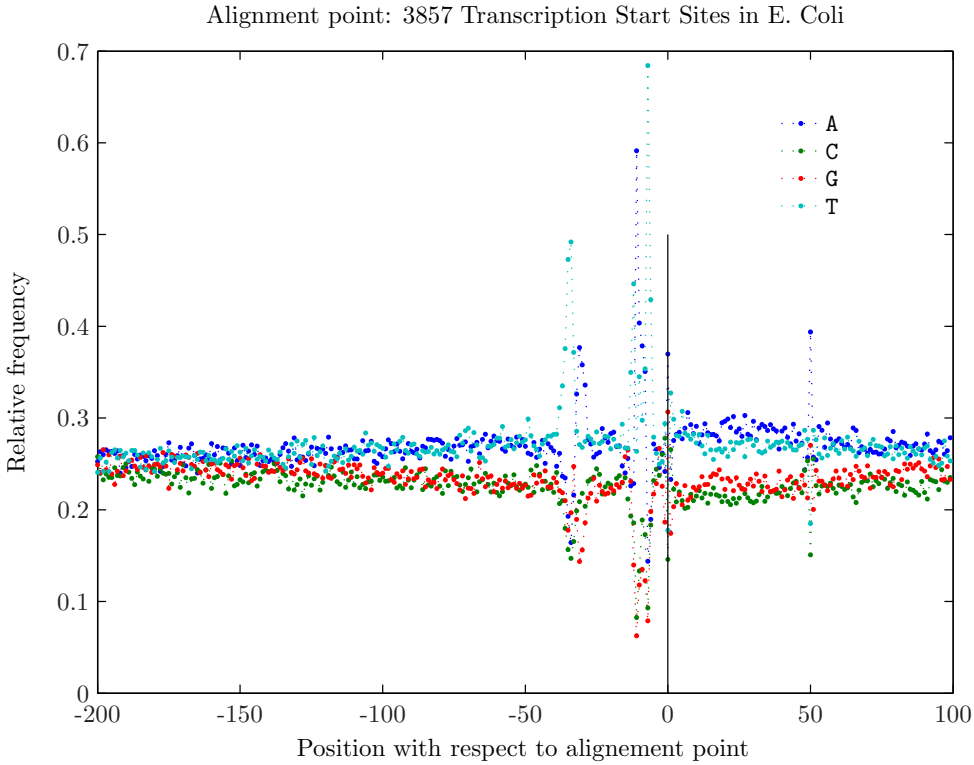


Figure 7.6: Nucleotide bias around transcription start sites in E. coli.

of the transcription start site is caused by the promoter binding sites situated at this location and discussed in detail in Section 7.3.4.

Recalling the derivation of the optimal LLR for synchronization in Section 6.2, it becomes obvious that for optimal performance positions with biased distribution around the binding site should be included in the computation of the likelihood in (7.13). In other words, the optimal likelihood function for binding site inference should not be restricted to the size of the binding site, but should also include the vicinity as long as its distribution is biased. This extension increases the overall discrimination information. The per position discrimination information $D(P_{X_i}||\boldsymbol{\pi})$ can be used to determine which positions should be included. Note that the Kullback-Leibler divergence follows a gamma distribution [Par06]

$$D(\hat{P}_{X_i}||\boldsymbol{\pi}) \sim \Gamma\left(\frac{1}{2}(|\mathcal{X}|-1), \frac{1}{N\ln 2}\right). \tag{7.16}$$

This notion can be used to test for identity and to compute a significance threshold, see Section 7.1.3. In Figure 7.7 the vicinity of the FNR binding sites has been included in the alignment. It can be seen that the motif is surrounded by sites which are divergent with a 95% probability (marked red). This fact is also reflected by the extended sequence logo on top of the divergence plot. By including these sites the false positive inference rate could be slightly reduced.

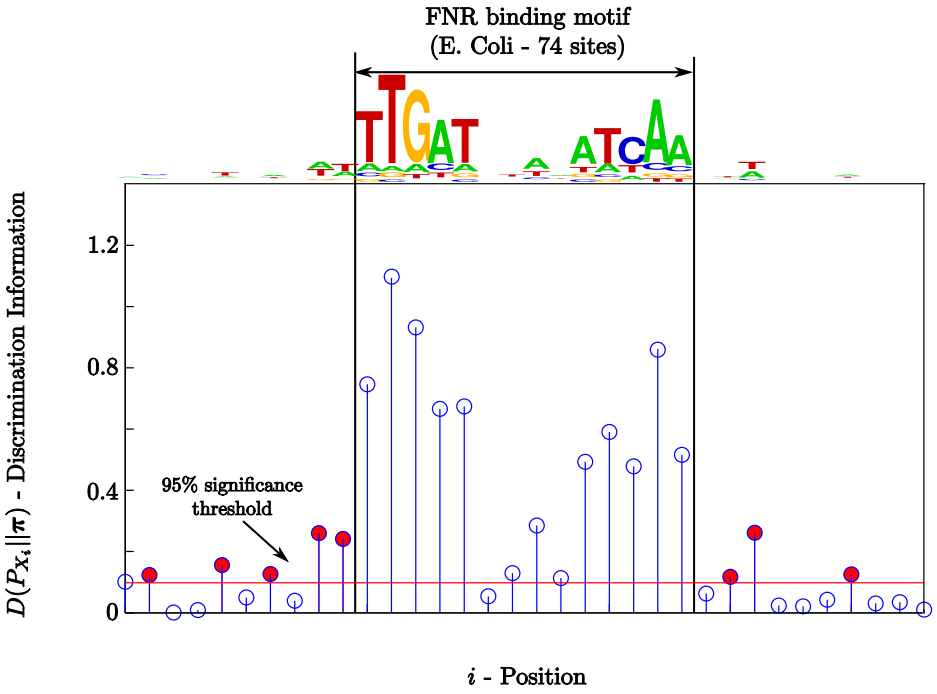


Figure 7.7: Positionwise divergence plot for the FNR binding motif and its vicinity. There exist divergent sites in the vicinity. These should be included in an optimal likelihood function for binding site inference.

7.2.4 Limitations of Binding Site Inference

It has been found that binding site inference methods relying solely on binding motif data greatly suffer from a high rate of false positives [WS04]. In other words, essentially almost all predicted binding sites that are generated with such methods will have no functional role[3]. Whether a potential site is an actual in vivo binding site depends apart from its sequence also from many other often epigenetic factors like the DNA methylation pattern and the limited accessibility of potential sites due to DNA packaging, see Section 3.2.1. Additionally, the content of adjoining sequences and proximity of other bound proteins has been found to play an important role [WS04]. However, datasets comprising the listed binding site sequence unrelated factors that could be used to improve the prediction accuracy are currently almost non-existent. Nonetheless, there exists phylogenetic foot printing data about candidate sites originating from multiple genome alignments, see Section 5.2. Under the assumption that mutations within regions having sequence-specific functionality accumulate slower due to evolutionary selection pressure, the identified binding site candidates can be filtered using evolutionary conservation profiles provided by comparative genomics [UVEB03]. In this fashion, the false positive rate can be reduced tenfold while retaining 70% of the experimentally validated sites [WS04].

[3]This has been referred to as the "futility" theorem in [WS04].

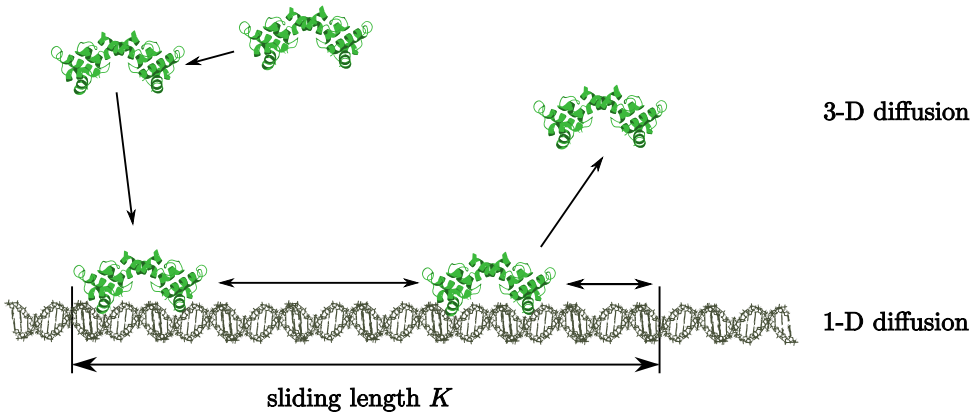


Figure 7.8: Binding site detection is a mixture of 3-D and 1-D diffusion. Sliding length K refers to the length of the DNA region scanned by the protein during the 1-D diffusion. Modified from [Wik08].

7.3 Molecular Synchronization

The in vivo binding site recognition shall be addressed in the following. In Section 7.3.1 the known biological aspects of binding site detection are presented. Section 7.3.2 establishes the parallels to threshold based synchronization. General properties of molecular markers are explained in Section 7.3.3. It is hypothesized that fundamental molecular processes are likely to use markers with good synchronization properties. This hypothesis is confirmed for the prokaryotic transcription initiation in Section 7.3.4 using the synchronization marker evaluation method derived in Chapter 6.

7.3.1 Binding Site Detection

The exact mechanism by which proteins detect binding sites in vivo remains an open question and is difficult to uncover experimentally [GG08]. Nonetheless, current experimental evidence supports the hypothesis that the detection is a mixture of three-dimensional (3-D) and one-dimensional (1-D) diffusion. The DNA binding protein floats in the cell nucleus looking for the DNA in a 3-D diffusion process. Once the double helix is found, it is bound non-specifically by the protein and searched for a marker in a 1-D diffusion sliding process, see Figure 7.8. Note that the protein can bind the linear DNA in two possible orientations. During the 1-D diffusion the protein undergoes randomly sliding steps to the left and to the right along the DNA. Eventually, if no marker is found, a dissociation event occurs. The characteristic distance explored between the association and dissociation events is referred to as the sliding length K. Due to the random nature of the 1-D diffusion, the same DNA sites are sampled repeatedly. The 3-D diffusion keeps the protein from spending too long 'oversampling' any particular region of the DNA contour by 1-D diffusion.

As explained in Section 3.2.1, most sequence specific protein DNA interactions take place in the major groove of the DNA. Since the DNA is a twisted double helix, this raises the question whether the protein rotates around the DNA helix during the 1-D diffusion.

Although rotation has not yet been directly observed, there exists indirect experimental evidence [SSS04, GG08]. This implies that the DNA is probed for the marker in single nucleotide sliding window steps. Binding affinity is used to probe for the marker. It can be concluded that the 1-D diffusion portion of the DNA binding site search corresponds to threshold based synchronization studied in Chapter 6.

7.3.2 Parallels to Threshold Synchronization

In Chapter 6 optimal marker synchronization techniques in engineering have been studied. It can be assumed that marker detection on the molecular level has evolved towards the use of an optimal binding site detection strategy. In Section 6.2.1, it has been argued that for synchronization purposes it is optimal to use a single syncword at the transmitter. In molecular biology this syncword is not known and has to be deduced from a set of binding sites corresponding to the received noisy realizations of it. Since the binding sites are still recognized by the binding protein, they can be assumed to have not diverged too much from the original syncword. Given the binding motif of a protein derived from its actual verified binding sites, the maximum likelihood estimate of the original syncword is the consensus sequence, see Section 7.1.2. Therefore, the molecular syncword of a protein shall be defined as the consensus sequence of its binding motif.

The threshold synchronizer evaluates a likelihood function for each position in a sliding window fashion. A threshold value is used to decide whether a position is claimed to be a syncword insertion point. Apart from the syncword also sequences that the syncword is likely to mutate to are recognized. Optimal likelihood functions for well established evolutionary substitution channel models have been derived in Section 6.2.4. In the context of molecular DNA binding site detection the equivalent of the likelihood function is the binding affinity of the binding protein to a particular DNA sequence, which can also be expected to have evolved towards optimality. Thus, assuming a substitution channel, it is valid to use the derived likelihood functions to study molecular synchronization. Neglecting insertions and deletions is justified by the fact that they are far less frequent and mostly have a detrimental effect on the functionality of a binding site. Opposed to the engineering case, the parameters of the substitution channel and the threshold value of the detector are not known. However, both can be estimated from the experimentally verified binding sites. The channel is estimated from the binding motif using maximum likelihood and the threshold value is chosen such that there is a good agreement between the annotated binding patterns and the ones recognized by the threshold detection, see Section 7.3.4.

7.3.3 General Properties of Molecular Markers

Unlike syncwords used in engineering, molecular markers can also be subject to other optimization criteria apart from maximizing the synchronization success probability, e.g. the marker might have to allow for easy unzipping of the two DNA strands and therefore has to be `AT`-rich (`A-T` bond is weaker).

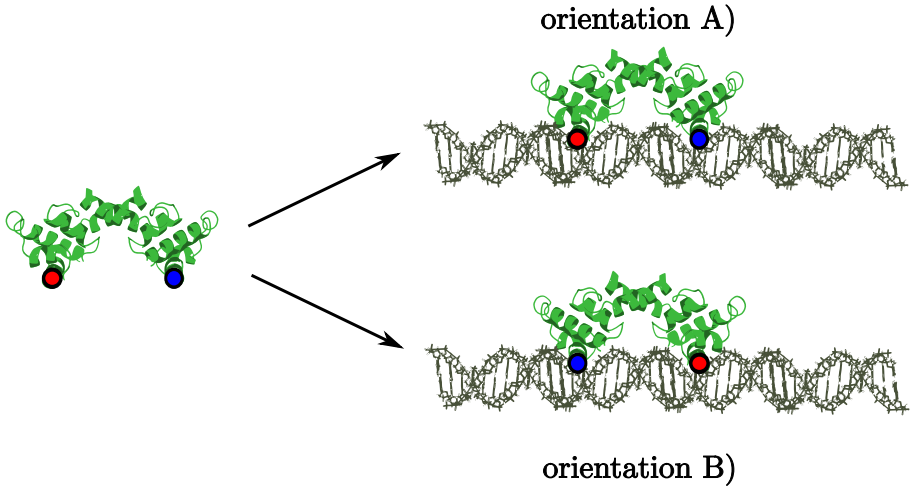


Figure 7.9: The two possible orientations A) and B) that a protein can bind to DNA. Modified from [Wik08].

The particularities of nucleotide accessibility in double stranded DNA also have implications on the structure of binding sites. In case of DNA protein binding, the binding site is often composed of two markers with a certain spacing. This relates to the twisted character of the double helix and the varying accessibility of the base pairs to the protein attaching from the side, see Section 7.1.1. For good synchronization performance the two markers should have a large Hamming distance and a possibly small cross-bifix in order to reduce the risk of shifted synchronization due to confusing one marker for the other, see also Section 6.3.

Additionally, for undirected binding proteins the marker should be self-reverse complementary. The self-reverse complementarity of the marker allows the binding site to be recognized in any of the two possible orientations that the protein can bind to the double stranded DNA, see Figure 7.9. Thereby, the overall recognition efficiency is effectively doubled. Note that undirected binding proteins are such that do not need to recognize the direction of the DNA. This is for instance the case for most transcription factors. Proteins involved in transcription initiation are an example of directed proteins, see Section 7.3.4.

Example: The consensus sequence of the FNR transcription factor comprises two 5 bp long syncwords separated by 4 arbitrary base pairs, see Figure 7.2. The Hamming distance of the two equals $d_h(\texttt{TTGAT}, \texttt{ATCAA}) = 3$. Note that no prefix of the first syncword is a suffix of the second, effectively reducing the risk of shifted synchronization. Additionally, the consensus sequence of the FNR protein is a spaced inverted repeat.

$$\overbrace{3' - \cdots \underrightarrow{\texttt{TTGAT}} \cdots \overrightarrow{\texttt{ATCAA}} \cdots - 5'}^{\text{1. orientation}}$$
$$\underbrace{5' - \cdots \underleftarrow{\texttt{AACTA}} \cdots \underleftarrow{\texttt{TAGTT}} \cdots - 3'}_{\text{2. orientation}}$$

This allows the FNR protein to recognize the binding site in any of the two possible orientations that it can bind to the DNA. ∗

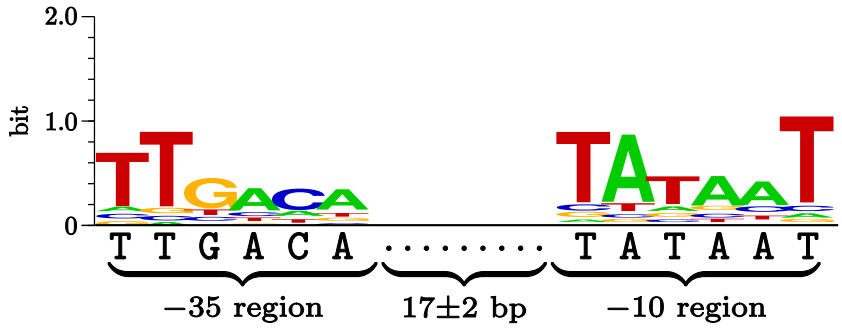


Figure 7.10: Depicted is the E. coli σ^{70} promoter structure. It comprises two distinct 6 bp long markers with the shown sequence logos (top) and consensus sequences (bottom).

Further constraints result from the simple fact that there exist binding sites for many different proteins. While some proteins bind to similar sites with the objective of mutually exclusive binding, many proteins use distinct sites that should not be bound by others. Markers for proteins requiring distinct sites should have a large Hamming distance and a small cross-bifix. Considering the amount of different binding proteins, good markers cannot be used by all binding proteins. However, fundamental processes like transcription are likely to have evolved towards the use of markers with good synchronization properties.

7.3.4 Synchronization Performance of Transcription Markers

In the following, the synchronization performance of prokaryotic transcription markers shall be studied using the framework derived in Section 6.3. Note that the procedure is equally applicable to other binding sites.

Marker Use in Prokaryotic Transcription

The step in transcription involving marker detection is transcription initiation. In the following transcription initiation will be explained for the Escherichia coli bacterium (E. coli) a common prokaryotic model organism. General background on the process of transcription is provided in Section 3.4.1. The transcription in prokaryotes is performed by the RNA polymerase (RNAp) protein complex. RNAp comprises a core enzyme and a detachable subunit called sigma factor (σ factor). The RNAp weakly attaches to the DNA and rapidly slides along the double helix until it disassociates again [AJW+08]. Once RNAp slides into a sequence specific region called a promoter, it binds tightly and initiates the transcription by opening up the double helix. The σ factor is responsible for the recognition of the promoter and makes specific contact with the bases exposed on the outside of the double helix. The main σ factor is called σ^{70} and is used for transcription under normal conditions[4]. The remaining σ factors are primarily used in reaction to environmental changes, e.g. during heat stress response.

Structure of the Promoter

In the following we will restrain to the use of σ^{70} for which there exist 672 documented E. coli promoters in the Regulon database [GCJJPG+08]. A σ^{70} promoter comprises two 6 bp long markers that lie 17 ± 2 bp apart. They are called the -35 and the -10 region respectively. Their names refer to their approximate midpoint distance to the transcription start site. The sequence logos and the consensus sequences of the two promoter markers are depicted in Figure 7.10. It can be seen that both sequences are well conserved.

The Promoter Synchronization Model

The synchronization performance of the two markers shall be studied using the method for threshold based synchronization over discrete memoryless channels derived in Section 6.3. In order to be able to apply the method, an appropriate channel model, its parameters and a good threshold value, have to be determined first.

The well established substitution channel models that have been presented and explained in detail in Section 3.7.1 can be used to study the synchronization properties of binding sites. In E. coli the genome background distribution is slightly biased $\boldsymbol{\pi} = (\pi_{\mathtt{A}}, \pi_{\mathtt{C}}, \pi_{\mathtt{G}}, \pi_{\mathtt{T}}) = (0.246, 0.254, 0.254, 0.246)$ [GACE+00]. Additionally, for bacteria the transition/transversion ratio is strongly biased $r_{Ts}/r_{Tv} = 2$ [Och03]. Thus, the continuous time Markov model of the substitution process is best described using the HKY rate matrix

$$\boldsymbol{R}_{HKY} = \begin{pmatrix} \star & \pi_{\mathtt{C}}\alpha & \pi_{\mathtt{G}}\beta & \pi_{\mathtt{T}}\alpha \\ \pi_{\mathtt{A}}\alpha & \star & \pi_{\mathtt{G}}\alpha & \pi_{\mathtt{T}}\beta \\ \pi_{\mathtt{A}}\beta & \pi_{\mathtt{C}}\alpha & \star & \pi_{\mathtt{T}}\alpha \\ \pi_{\mathtt{A}}\alpha & \pi_{\mathtt{C}}\beta & \pi_{\mathtt{G}}\alpha & \star \end{pmatrix}. \tag{7.17}$$

The transition probability matrix $\boldsymbol{P}$ characterizing the substitution channel is the matrix exponential of the rate matrix $\boldsymbol{P} = e^{t \cdot \boldsymbol{R}}$, where t is the evolutionary time and reflects the degree of conservation. Given the genome background distribution $\boldsymbol{\pi}$, the transition/transversion ratio $r_{Ts}/r_{Tv} = \frac{\beta}{2\alpha}$ and by setting $\alpha = 1$, the only remaining free parameter is the degree of conservation t. It can be estimated using maximum likelihood from actual binding sites under the assumption that the consensus sequence has been transmitted over the HKY channel. Note that all positions in the binding motif are assumed to be subject to the same channel. An estimate of t based on the 673 annotated σ^{70} promoters leads to the following channel matrix

$$\boldsymbol{P}_{\sigma^{70}} = \begin{pmatrix} 0.755 & 0.047 & 0.158 & 0.046 \\ 0.046 & 0.748 & 0.047 & 0.153 \\ 0.153 & 0.047 & 0.748 & 0.046 \\ 0.046 & 0.158 & 0.047 & 0.755 \end{pmatrix}, \tag{7.18}$$

It can be seen that the probability of a substitution occurring at any position in the promoter consensus sequence when transmitted over this channel is around 25%.

[4]The number 70 in σ^{70} refers to the molecular weight of the σ factor in kilo Dalton.

The optimal likelihood function for the substitution channel has been derived in Section 6.2.4. However, the actual likelihood function used by the σ-factor is the binding affinity. Even though, the exact binding affinity of the σ-factor to different nucleotides is not known, it is justified to assume that it has evolved towards optimality and should be proportional to the optimal likelihood function for the substitution channel derived in (6.26).

At last the threshold value of detection remains to be determined. With respect to the threshold value, it turns out that by fixing the false negative rate to $\alpha = 10\%$ in the Neyman-Pearson criterion (2.17) a threshold value is obtained that leads to a good agreement between the annotated promoter markers actually recognized by the σ-factor and the markers recognized by the detection. In engineering fixing $\alpha = 10\%$ means that at most 10% of the transmitted markers would not be recognizable by the threshold synchronizer at the receiver due to transmission errors.

Synchronization Performance of the -35 and -10 Promoter Region

The two markers of the σ_{70} promoter are subject to several different constraints. Since the transcription is directed, the promoter should not be a spaced inverted repeat. In fact, the Hamming distance to the reverse complement should be sufficiently large to prevent binding in the opposite orientation. This requirement seems to be fulfilled

$$d_h(\texttt{TTGACA}\cdots\texttt{TATAAT}, \texttt{ATTATA}\cdots\texttt{TGTCAA}) = 6.$$

Additionally, the use of two markers involves the risk of shifted synchronization, due to confusing one marker with the other. Therefore, the two markers should have a large Hamming distance and a small cross-bifix overlap, which is the case

$$d_h(\texttt{TTGACA}, \texttt{TATAAT}) = 4.$$

Note that the DNA has to be unzipped during transcription initiation in the -10 promoter region [AJW+08]. This imposes an additional constraint on the composition of the -10 region which is motivated biologically and not by the synchronization performance. It has to be AT-rich since the A-T bond is weaker.

The length K (sliding length) of the region scanned by the RNAp during a 1-D diffusion search actually corresponds to the data frame length K, see Section 6.3. For all search attempts where a binding site is present in the scanned region, the starting offsets k are equally distributed. The question to be answered is how do the actual σ^{70} promoter consensus markers perform compared to other markers of the same length. Could nature have chosen better markers? A relative comparison makes it possible to treat the two promoter markers separately. This greatly simplifies the computation, since it is no more necessary to compare 4^{12} marker patterns but only 4^6. The separate consideration is justified because the spacing between the two markers is large compared to their actual length and because the risk of shifted synchronization due to confusing one marker with the other is low.

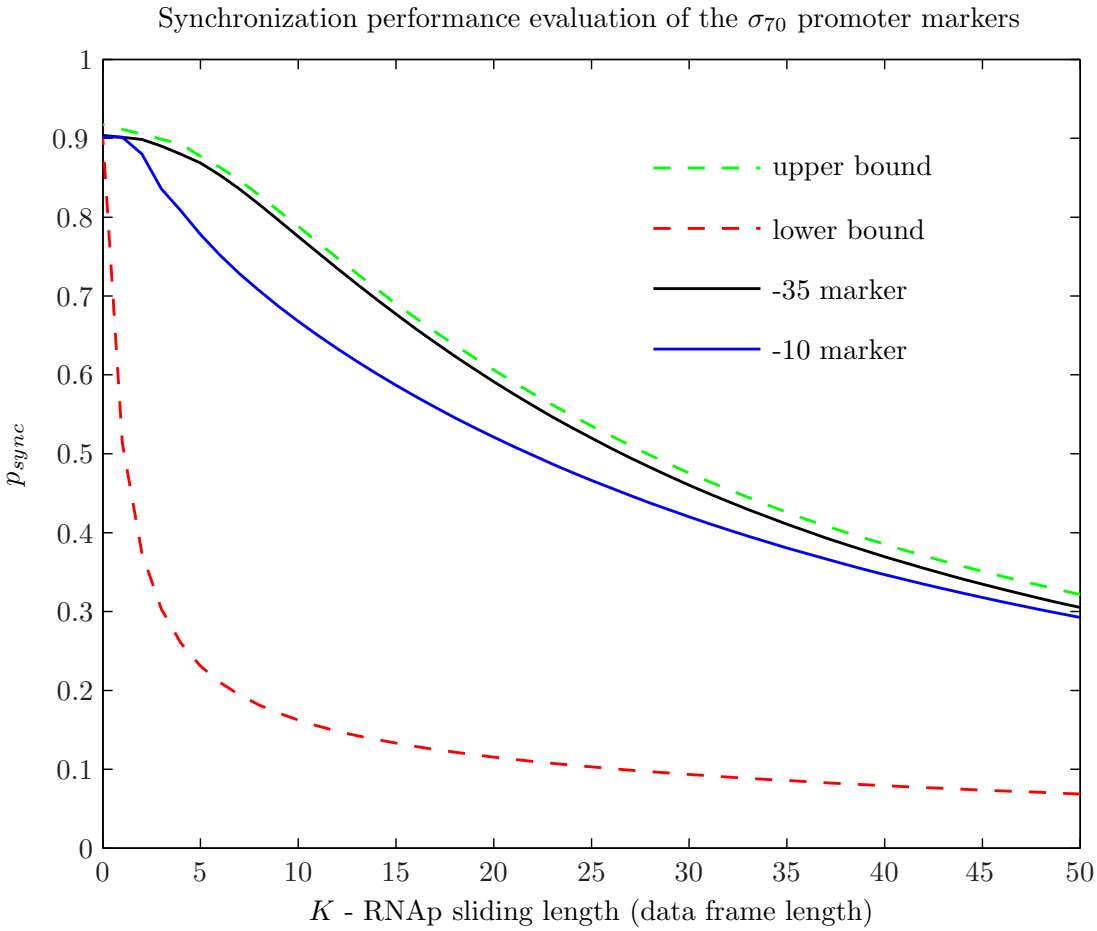


Figure 7.11: Relative synchronization performance of the σ_{70} promoter markers. The upper and lower bound correspond to the performance of the best and worst performing patterns for any given sliding length K (data frame length).

Figure 7.11 depicts the performance of the -35 and the -10 marker for different RNAp sliding lengths K (data frame lengths) under the assumption that a marker is present in the scanned region. Note that the derived $\boldsymbol{P}_{\sigma_{70}}$ channel and threshold corresponding to $\alpha = 10\%$ was used. Also depicted is the upper and lower bound. The bounds correspond to the performance of the best and worst markers for a particular sliding length K respectively. It can be seen that the -35 region is performing close to the upper bound for the entire plotted range of possible sliding lengths. The -10 promoter is slightly inferior but still a good performer. The limited performance of the -10 region can be attributed to the biologically imposed requirement of `AT`-richness. It can be concluded that the -35 marker of the σ_{70} promoter confirms the hypothesis that important molecular markers evolve towards superior synchronization performance.

7.4 Summary

In this Chapter the parallels between marker synchronization in engineering and sequence specific binding in genetics have been studied. Sequence markers are used by the molecular machinery across different stages of genetic information processing to distinguish

information carrying regions and to mark binding positions for regulatory proteins. The insights about marker synchronization gained in Chapter 6 were used to study in silico inference techniques of putative binding sites and the in vivo binding site detection on the molecular level.

First, quantitative description models of binding sites were analysed using an information theoretic framework. Particular attention was paid to the appropriate treatment of the small sample size resulting from a small amount of experimentally verified binding sites available in public databases. The focus was placed on DNA protein binding. Mutual information was used to test the independence of binding site positions.

An important field of bioinformatics research motivated by the lack of experimentally verified sites is the in silico binding site inference. Parallels were established between the likelihood function used in inference and in synchronization. As an extension to the current inference scheme it has been proposed to include the neighbouring positions of the binding sites into the inference algorithms to improve performance. Kullback-Leibler divergence was used to measure the per position discrimination information of binding sites and their vicinity and to decide which positions should additionally be included.

The in vivo binding site detection strongly resembles threshold based synchronization over noisy channels. The binding sites are subject to evolutionary mutations and the binding protein typically allows a certain variability in the recognized sequence patterns. The consensus sequence of binding sites of the same type has been defined as the molecular syncword used by the protein. The experimentally verified binding sites were considered to be noisy realizations of it and used to estimate the parameters of the underlying substitution channel. General properties of molecular markers have been studied. Transcription factors that do not need to distinguish the direction of the DNA were shown to have evolved towards the use of self-reverse complementary markers which effectively doubles their synchronization performance. Some molecular markers were shown to be subject to biologically motivated constraints that restrict their synchronization performance. The marker evaluation method derived for threshold based synchronization over discrete memoryless channels in Chapter 6 was used to assess the performance of the promoter marker in prokaryotic transcription initiation. The marker was found to perform close to the optimum.

The author has been awarded the 2006 Best Student Presentation Award of the International Society for Computational Biology (ISCB) Student Council for the work on the synchronization properties of molecular markers [HW06].

8

Conclusion

Modern communication systems are digital. Representing information digitally offers the possibility to code it in a way such that it can be losslessly compressed or transmitted error-free over a noisy channel [Sha48]. Interestingly, the genetic information of each organism is stored in the DNA as a quaternary digital signal. Therefore, the theory and methods developed to design and analyse communication systems can also be used to study the storage, processing and transmission of genetic information to progeny on the molecular level. Recent progress in DNA sequencing technology has caused an exponential growth of the available sequence data. The sequencing of the human genome [LLB$^+$01] and other vertebrate genomes has enabled comparative genomics. The genetic information was found to comprise more than protein coding genes. It has been revealed that the information stored in the DNA and its processing is far from understood [BPM$^+$04,Che07]. The increasing availability of sequence data and the newly raised questions have also ignited the interest of communication engineers and information theorists. The initial research results [HDG$^+$04, SGD$^+$07, DHHM05, DHL$^+$08, MV04, SRS05] confirm that the interdisciplinary approach can be highly beneficial. On the one hand, methods to store, distribute and analyse the large amounts of collected data are contributed. On the other hand, new insights are gained by modelling and studying molecular information processing from a communications theoretic perspective.

The results obtained during this thesis represent contributions in both areas. In Part I a highly efficient compression algorithm for multiple genome alignment datasets has been developed. In Part II parallels between marker synchronization and sequence specific binding have been established. Evidence that molecular markers evolve also under the constraint of good synchronization performance has been found. In the following, the main contributions shall be summarized.

Part I

In Part I the problem of efficient storage and distribution of multiple genome alignment datasets has been addressed. Such alignments are used for comparative studies and represent one of the largest sequence datasets used in molecular biology. Every few years a new dataset comprising roughly twice as many species is released. In order to compare genome sequences of different species, they have to be aligned to compensate for evolutionary mutations comprising insertions, deletions and substitutions. An extra "gap" symbol is used to denote the missing entries in the alignment. Large scale mutations cause genome rearrangements. Therefore, multiple genome alignment datasets comprise many locally aligned blocks of homologous regions that share common ancestry. Typically, the genome alignments are pre-computed by large labs possessing the necessary computational resources and centrally provided for download.

In this thesis the first highly efficient lossless compression scheme for genome alignments has been proposed. In order to design a good algorithm, different types of universal source coding techniques and DNA specific compression algorithms have been studied first. The key to DNA compression was found to be appropriate modelling of genome evolution. The statistical dependencies present in multiple genome alignments are determined by the mutational processes and the evolutionary relationship of the aligned species. Therefore, the nucleotide portion of the alignment is compressed using predictions based on well established statistical models of evolutionary substitutions in combination with arithmetic coding. Due to the lack of elaborate statistical models for insertions and deletions, the gap positions have to be compressed differently. Techniques from lossless binary image compression are used. The multiple genome alignment datasets are distributed in form of multiple alignment format files containing for each aligned block also the position information about the contained homologous regions. It is possible to efficiently compress this supplementary information by exploiting subtle dependencies in the position information of consecutive blocks. The proposed multiple alignment file compression algorithm (MAFc) reduces the file size tenfold and is twice as efficient as the universal compression algorithms like the dictionary based Lempel-Ziv or the statistical prediction based Context Tree Weighting. The complexity of compression is higher than that of decompression which fits the distribution scenario. An important feature of the MAFc algorithm is that it allows to decompress individual alignment blocks without having to decompress the entire dataset. Thus, MAFc is also suitable as back end of database servers that are typically queried by researchers for individual alignment blocks. The algorithm scales well in terms of newly included species in the dataset. The more species are included the better compression rate can be expected. The proposed algorithm has received the 2009 Capocelli Prize, which is the Best Paper Award of the Data Compression Conference [HDCH09].

The MAFc algorithm is currently optimized for the multiple genome alignment datasets provided by the UCSC Genome Browser database. In future work it could be adopted to datasets provided by the Ensembl database. Additionally, the possibilities of using the algorithm in the context of biological data analysis could be investigated. For example, it could be used as an alternative measure of evolutionary conservation.

Part II

In Part II of this thesis parallels between binding site detection and marker synchronization over noisy channels have been studied. Marker synchronization in engineering refers to the recovery of the insertion points of short sequence patterns artificially introduced into the transmitted data stream to mark certain positions. At the receiver a threshold synchronizer evaluates in a sliding window fashion for each position the likelihood that it is an actual marker insertion point. The scenario closely resembles binding site detection on the molecular level. In DNA protein binding the protein slides along the DNA using binding energy to find its sequence specific binding site.

Marker use in engineering was reviewed first. Thereby, novel results could be obtained, demonstrating that the interdisciplinary research can also benefit engineering. The optimal likelihood function for marker synchronization over Gaussian channels and approximations thereof were introduced by J. Massey in [Mas72]. Herein, it has been shown that the optimal likelihood function is equivalent to the mutual self-information between the syncword and the received pattern observed at a position. Based on this observation a new superior approximation has been derived for the Gaussian channel. The likelihood function was further adopted to discrete memoryless channels, in particular, to the evolutionary substitution channel models. Different syncword patterns of the same length are known to perform differently well with respect to the synchronization success probability. The exact computation of this probability is non-trivial due to the dependencies of the likelihood function values of neighbouring positions, which is generally neglected in the literature. Herein, an exact recursive formula is proposed for discrete memoryless channels and shown to deviate significantly from the approximation assuming independence. In future work, the formula could be adapted to continuous channel models.

The DNA binding sites of a particular protein can be regarded as noisy realizations of the molecular marker recognized by the protein transmitted over the evolutionary substitution channel. Currently, the amount of experimentally verified binding sites is rather limited. Established binding site models and in silico binding site inference methods were reviewed under this aspect from an information theoretic perspective and extensions were proposed. The in vivo binding site detection and the synchronization properties of selected molecular markers were studied next. Transcription factors were shown to have evolved towards the use of self-reverse complementary markers which effectively doubles their synchronization performance. The syncword evaluation method derived for threshold based synchronization over discrete memoryless channels was used to assess the performance of the promoter marker used in prokaryotic transcription initiation. The marker was found to perform close to the optimum. In future work the method could be applied to other binding proteins. The work on the synchronization properties of molecular markers has been awarded the 2006 Best Student Presentation Award of the International Society for Computational Biology (ISCB) Student Council [HW06].

A

Publications

Parts of this work were published in the following articles:

Journal Papers

[1] P. Hanus, J. Dingel, G. Chalkidis, and J. Hagenauer, "Compression of whole genome alignments," *IEEE Transactions on Information Theory*, vol. 56 no. 2, pp. 696–705, 2010.

[2] J. Weindl, Z. Dawy, P. Hanus, J. Zech, and J. C. Mueller, "Modeling promoter search by E. coli RNA polymerase: One-dimensional diffusion in a sequence-dependent energy landscape," *Journal of Theoretical Biology*, vol. 259, no. 3, pp. 628–634, 2009.

[3] J. Dingel, P. Hanus, N. Leonardi, J. Hagenauer, J. Zech, and J. Mueller, "Local conservation scores without a priori assumptions on neutral substitution rates," *BMC Bioinformatics*, vol. 9, no. 1, p. 190, 2008.

[4] P. Hanus, B. Goebel, J. Dingel, J. Weindl, J. Zech, Z. Dawy, J. Hagenauer, and J. C. Mueller, "Information and communication theory in molecular biology," *Electrical Engineering (Archiv fuer Elektrotechnik)*, vol. 90, no. 2, pp. 161–173, 2007.

[5] Z. Dawy, P. Hanus, J. Weindl, J. Dingel, and F. Morcos, "On genomic coding theory," *European Transactions on Telecommunications*, vol. 18, no. 8, pp. 873–879, 2007.

[6] J. Weindl, P. Hanus, Z. Dawy, J. Zech, J. Hagenauer, and J. C. Mueller, "Modeling DNA-binding of Escherichia coli sigma 70 exhibits a characteristic energy landscape around strong promoters," *Nucleic Acids Research*, vol. 35, no. 20, pp. 7003–7010, 2007.

[7] M. Sarkis, B. Goebel, Z. Dawy, J. Hagenauer, P. Hanus, and J. C. Mueller, "Gene mapping of complex diseases-A comparison of methods from statistics information theory, and signal processing," *IEEE Signal Processing Magazine*, vol. 24, no. 1, pp. 83–90, 2007.

[8] P. Hanus and J. Hagenauer, "Information theory helps historians," *IEEE Information Theory Society Newsletter*, vol. 55, no. 8, 2005.

Conference Proceedings

[9] P. Hanus, J. Dingel, G. Chalkidis, and J. Hagenauer, "Source coding scheme for multiple sequence alignments," in *Proceedings of the Data Compression Conference (DCC09)*, Mar. 2009, pp. 183–192. **(Best Paper Award)**

[10] V. Y. Kuryshev and P. Hanus, "Compression based classification of primate endogenous retrovirus sequences," in *Proceedings of the International Workshop on Computational Systems Biology (IWCSB08)*, Jun. 2008, pp. 81–84.

[11] P. Hanus, "Synchronization properties of protein binding sites," in *Proceedings of the International Conference on Intelligent Systems for Molecular Biology Student Council Symposium (ISMB07)*, Jul. 2007.

[12] P. Hanus, J. Dingel, J. Zech, J. Hagenauer, and J. C. Mueller, "Information theoretic distance measuers in phylogenomics," in *Proceedings of the International Workshop on Information Theory and Applications (ITA07)*, Jan. 2007, pp. 421–425.

[13] P. Hanus and J. Weindl, "Synchronization model of transcription initiation in prokaryotes," in *Proceedings of the International Conference on Intelligent Systems for Molecular Biology Student Council Symposium (ISMB06)*, Aug. 2006. **(Best Student Presentation Award)**

[14] P. Hanus, J. Dingel, J. Hagenauer, and J. Mueller, "An alternative method for detecting conserved regions in multiple species," in *Proceedings of the German Conference on Bioinformatics (GCB05)*, Oct. 2005, p. 64.

[15] P. Hanus, Z. Dawy, J. Hagenauer, and J. C. Mueller, "DNA classification using mutual information based compression distance measures," in *Proceedings of the International Conference of Medical Physics (ICMP05)*, Sep. 2005, pp. 1434–1435.

[16] B. Goebel, M. Sarkis, Z. Dawy, P. Hanus, J. Hagenauer, and J. C. Mueller, "Mutual information and independent component analysis in population-based gene mapping," in *Proceedings of the International Conference of Medical Physics (ICMP05)*, Sep. 2005, pp. 1452–1453.

[17] P. Hanus, J. Dingel, J. Hagenauer, and J. C. Mueller, "An alternative method for detecting conserved elements in multiple sequence alignments," in *Proceedings of the European Conference on Computational Biology Student Council (ECCB05)*, Sep. 2005, pp. 18–20.

[18] Z. Dawy, J. Hagenauer, P. Hanus, and J. C. Mueller, "Mutual information based distance measures for classification and content recognition with applications to genetics," in *Proceedings of the IEEE International Conference on Communications (ICC05)*, vol. 2, May 2005, pp. 820–824.

[19] J. Hagenauer, Z. Dawy, B. Gobel, P. Hanus, and J. Mueller, "Genomic analysis using methods from information theory," in *Proceedings of the IEEE Information Theory Workshop (ITW04)*, Oct. 2004, pp. 55–59.

B

Derivations

B.1 Matrix Exponential

The matrix exponential $e^{\boldsymbol{R}}$ is defined by its Taylor expansion and is easily evaluated if the singular value decomposition for $\boldsymbol{R}$ exists: Let $\boldsymbol{R}$ be a rate matrix on $\mathcal{A}$ and suppose $\boldsymbol{R}$ can be decomposed as

$$\boldsymbol{R} = \boldsymbol{U}\boldsymbol{\Phi}\boldsymbol{U}^{-1}, \tag{B.1}$$

where $\boldsymbol{U}$ denotes the matrix of eigenvectors and $\boldsymbol{\Phi}$ the matrix with the eigenvalues of $\boldsymbol{R}$ in the diagonal. Then it can be shown that [Nor97]:

$$\boldsymbol{P}(t) = e^{t\boldsymbol{R}} = \boldsymbol{U}e^{t\boldsymbol{\Phi}}\boldsymbol{U}^{-1},$$

with

$$e^{t\boldsymbol{\Phi}} = \begin{pmatrix} e^{t\phi_1} & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & e^{t\phi_{|\mathcal{A}|}} \end{pmatrix}.$$

For most types of rate matrices assumed in molecular evolution theory, the singular value decomposition of $\boldsymbol{R}$ always exists. In particular it always exists for rate matrices defining a reversible process [Kel79].

B.2 Synchronization Success Probability

In the following the recursive formula for the synchronization success probability $p_{sync}(k)$ in (6.38) shall be derived and intuitively explained. Note that the channel is assumed to be noiseless. In this case the synchronization success probability $p_{sync}(k)$ can also be

interpreted as the probability that no other occurrence of the syncword $\boldsymbol{s}$ is found when sliding back k steps from the pattern insertion position. First, assume $k > L$ such that the detector at position k only sees symbols originating from the data and there is no overlap between the inserted syncword and the observation. Let $p_{fail}(k)$ be the probability that the first emulated occurence of $\boldsymbol{s}$ is found after sliding back exactly k steps. Since bifix-free patterns do not overlap $p_{fail}(k)$ can be expressed $\forall \boldsymbol{s}_{(h=0)}$ as the probability that no emulation has taken place until the position $(k-L)$ multiplied by the probability $r^{(L)}$ of emulation of $\boldsymbol{s}$ in the data

$$p_{fail}(k) = r^{(L)} p_{sync}(k-L), \forall \boldsymbol{s}_{(h=0)}. \tag{B.2}$$

If the pattern is not bifix-free, then for $l = 1 \ldots L-1$ portions of the emulated $\boldsymbol{s}$ for which the bifix $h^{(L-l)}$ exists will lead to a failure already at $(k-l)$. Their contribution has to be subtracted from (B.2). Thus, for $k > L$

$$p_{fail}(k) = r^{(L)} p_{sync}(k-L) - \sum_{l=1}^{L-1} h^{(L-l)} r^{(l)} p_{fail}(k-l), \forall \boldsymbol{s}, \tag{B.3}$$

and using the fact that $p_{fail}(k) = p_{sync}(k-1) - p_{sync}(k)$ we obtain

$$p_{sync}(k) = p_{sync}(k-1) - \sum_{l=1}^{L-1} h^{(L-l)} r^{(l)} \big(p_{sync}(k-l-1) - p_{sync}(k-l)\big) - r^{(L)} p_{sync}(k-L). \tag{B.4}$$

By setting $r^{(0)} = h^{(0)} = h^{(L)} = 1$ per definition it can be further simplified to

$$p_{sync}(k) = \sum_{l=1}^{L} \big(h^{(L+1-l)} r^{(l-1)} - h^{(L-l)} r^{(l)}\big) p_{sync}(k-l), \forall k > L. \tag{B.5}$$

Finally, the special case of $0 < k \leq L$ has to be treated and the initial value of the recursion $p_{sync}(0)$ set arriving at Equation (6.38).

Nomenclature

List of Abbreviations

1-D	one-dimensional (p. 127)
2-D	two-dimensional (p. 78)
3-D	three-dimensional (p. 127)
AC	arithmetic coding (p. 46)
ACF	autocorrelation function (p. 111)
ASCII	American Standard Code for Information Interchange (p. 55)
AWGN	average white Gaussian noise channel (p. 14)
bp	base pairs (p. 19)
BPSK	binary phase shift keying (p. 93)
BSC	binary symmetric channel (p. 14)
CDF	cumulative density function (p. 6)
chr21	chromosome 21 (p. 74)
CRP	transcription factor protein (p. 124)
CTW	Context Tree Weighting (p. 51)
DMC	discrete memoryless channel (p. 14)
DNA	deoxyribonucleic acid (p. 18)
DNAc	DNAcompress (p. 62)
DNAx	DNA-eXpertModel (p. 62)
E. coli	Escherichia coli bacterium (p. 130)
FEL	Felsenstein evolutionary model (p. 34)

FNR	transcription factor protein (p. 118)
GB	Gigabyte (p. 86)
GTR	general time reversible evolutionary model (p. 33)
HKY	Hasegawa, Kishino, and Yano evolutionary model (p. 34)
IID	independent and identically distributed (p. 7)
InDel	insertion and deletion (p. 30)
ISCB	International Society for Computational Biology (p. 134)
IUD	independent and uniformly distributed (p. 7)
JBIG	Joint Bi-level Image Experts Group (p. 76)
JC	Jukes Cantor evolutionary model (p. 34)
K2P	Kimura 2 parameter evolutionary model (p. 34)
KT	Krichevsky Trofimov estimator (p. 50)
LLR	log likelihood ratio (p. 8)
LZ	Lempel and Ziv (p. 56)
LZMA	Lempel-Ziv-Markov chain algorithm (p. 57)
LZW	Lempel-Ziv-Welch algorithm (p. 57)
MAF	multiple alignment format (p. 68)
MAFc	multiple alignment format file compressor (p. 81)
MAP	maximum a-posteriori probability estimator (p. 8)
MDL	minimum description length (p. 55)
ML	maximum likelihood estimator (p. 8)
mRNA	messenger RNA (p. 20)
MSA	multiple sequence alignment (p. 64)
MSAc	multiple sequence alignment compressor (p. 78)
NCBI	National Center for Biotechnology Information (p. 59)
ncRNA	non-coding RNA (p. 20)
NP-hard	non-deterministic polynomial-time hard (p. 63)

PDF	probability density function (p. 6)
PFM	position frequency matrix (p. 117)
PMc	puncturing matrix compressor (p. 77)
PMF	probability mass function (p. 6)
PSK	phase shift keying (p. 112)
RNA	ribonucleic acid (p. 20)
RNAp	RNA polymerase (p. 130)
rRNA	ribosomal RNA (p. 20)
RV	random variable (p. 5)
SD	Shine-Dalgarno (p. 27)
SIMO	single input multiple output (p. 36)
SNR	signal to noise ratio (p. 93)
SW	Smith-Waterman algorithm (p. 65)
tRNA	transfer RNA (p. 20)
TSS	transcription start site (p. 24)
UCSC	University of California Santa Cruz (p. 59)
UTR	untranslated region (p. 27)

List of Notations

$f(\cdot)$	arbitrary function (p. 6)
$\tilde{f}(\cdot)$	approximation of the function $f(\cdot)$ (p. 96)
$\arg\max(\cdot)$	returns the maximizing argument (p. 8)
$\lceil\cdot\rceil$	ceiling function (p. 45)
$\cosh(\cdot)$	hyperbolic cosine (p. 96)
$D(\cdot\|\|\cdot)$	Kullback-Leibler divergence (relative entropy) (p. 11)
$\mathrm{E}\{\cdot\}$	expectation function (p. 6)
$\lfloor\cdot\rfloor$	floor function (p. 110)
$H(\cdot)$	entropy function (p. 10)

$\mathcal{H}(\cdot)$	entropy rate (p. 12)
$d_h(\cdot,\cdot)$	Hamming distance between two sequences (p. 99)
$I(\cdot;\cdot)$	mutual information (p. 11)
$\mathcal{I}(\cdot;\cdot)$	information rate (p. 13)
$\vartheta(\cdot)$	indicator function (p. 56)
$L(\cdot)$	log likelihood ratio function (p. 9)
$\log(\cdot)$	logarithm function (p. 10)
$\mathrm{ld}(\cdot)$	binary logarithm (p. 10)
$\ln(\cdot)$	natural logarithm (p. 96)
$\max(\cdot)$	maximum function (p. 8)
$(\cdot)_2$	binary representation of the value (p. 47)
$(\cdot)_2$	decimal representation of the value (p. 49)
$\mathrm{sign}\,(\cdot)$	sign function (p. 96)
$F_X(\cdot)$	cumulative distribution function (p. 6)
$\mathtt{N}$	typewriter capital letter - nucleotide type (p. 18)
$p_X(\cdot)$	lower case p - probability density function (PDF) of a continuous RV (p. 6)
$P_X(\cdot)$	capital P - probability mass function (PMF) of a discrete RV (p. 6)
$P_{XY}(\cdot,\cdot)$	joint distribution function (p. 6)
$\mathbb{N}$	blackboard bold letter - set of numbers (p. 7)
X	capital letter - random variable (RV) (p. 5)
$\{X_t\}_{t\geq 0}$	continuous time random process (p. 32)
$\boldsymbol{X}$	bold capital letter - matrix (p. 32)
$[\boldsymbol{X}]_{ij}$	element in ith row and jth column of a matrix (p. 32)
$\boldsymbol{X}^T$	transpose of a matrix (p. 33)
x	lower case letter - realization of random variable (p. 5)
$\lvert x\rvert$	absolute value (p. 6)
$\hat{x}$	symbol with a hat - estimate value (p. 8)

$\boldsymbol{x}$	bold lower case letter - vector or sequence (p. 8)
$[\boldsymbol{x}]_i$	ith element of a vector (p. 8)
$\lvert\boldsymbol{x}\rvert$	length of the sequence $\boldsymbol{x}$ (p. 44)
$\boldsymbol{x}^n$	sequence $(x_1 \dots x_n)$ of symbols (p. 45)
$\mathcal{X}$	calligraphic symbol - countable non-empty sample space (alphabet) (p. 5)
$\lvert\mathcal{X}\rvert$	cardinality of sample space (alphabet size) (p. 6)

List of Symbols

$\boldsymbol{A}$	sequence alignment matrix (p. 65)
$\mathcal{A}$	nucleotide alphabet (p. 32)
`A`	adenine nucleotide (p. 18)
C	channel capacity (p. 13)
c	pseudocount (p. 123)
$C(x)$	codeword corresponding to x (p. 44)
`C`	cytosine nucleotide (p. 18)
$\boldsymbol{c}$	context sequence (p. 53)
D	context size in symbols (p. 51)
d_h^{max}	maximum tolerable Hamming distance (p. 107)
$\delta(x)$	codeword redundancy (p. 45)
$D(P_{\boldsymbol{X}}\lvert\lvert\boldsymbol{\pi})$	discrimination information of a binding motif (p. 123)
E_s	average transmission power (p. 14)
$E(\boldsymbol{r})$	binding energy of nucleotide sequence $\boldsymbol{r}$ (p. 122)
E_s/N_0	signal to noise ratio (SNR) (p. 96)
$\Gamma(n, \lambda)$	Gamma distribution with n degrees of freedom and shape parameter λ (p. 7)
`G`	guanine nucleotide (p. 18)
$\mathcal{H}_0$	null hypothesis (p. 9)
$\mathcal{H}_1$	alternative hypothesis (p. 9)
h	decimal representation of the bifix pattern (p. 103)

$h^{(l)}$	bifix indicator of self-overlap between prefix and suffix of length l (p. 102)
$h_{ji}^{(l)}$	cross-bifix indicator between suffix of $\boldsymbol{s}'_j$ and prefix of $\boldsymbol{s}'_i$ of length l (p. 106)
H_π	background entropy of a genome (p. 118)
I_n	nth interval (p. 49)
$I(\boldsymbol{s};\boldsymbol{r})$	mutual self-information between syncword $\boldsymbol{s}$ and received sequence $\boldsymbol{r}$ (p. 95)
k	starting offset of syncword search to syncword insertion point (p. 103)
K	data frame length (p. 104)
ℓ	leaf of the phylogenetic tree (p. 37)
λ	threshold value (p. 9)
L	syncword length (p. 91)
L_M	optimal self-information based synchronization likelihood function (p. 97)
L_{LLR}	optimal LLR function for marker synchronization (p. 94)
L_M	optimal synchronization likelihood function by Massey [Mas72] (p. 96)
μ_{SW}	syncword insertion position (p. 102)
$\mathcal{N}(\mu,\sigma^2)$	Gaussian distribution with variance σ and mean μ (p. 7)
N	number of binding sites in a binding motif (p. 119)
$N_0/2$	two-sided power spectral density of AWGN noise (p. 14)
`N`	any nucleotide (p. 117)
$\boldsymbol{P}$	DMC channel matrix (p. 32)
p_{hit}	probability that threshold synchronizer claims synchronization (p. 102)
p_{sync}	synchronization success probability (p. 101)
$\boldsymbol{\pi}$	genome background distribution of nucleotides (p. 33)
$\boldsymbol{R}$	rate matrix (p. 32)
R	code rate in bits/symbol (p. 13)
`R`	purine nucleotide (p. 19)
R_c	columnwise compression rate (p. 71)
R_i	information content of ith binding motif column (p. 118)

$r^{(l)}$	probability of occurrence of the prefix of length l in the data (p. 103)
R_n	nucleotidewise compression rate (p. 72)
r_{Ts}/r_{Tv}	transition to transversion mutation rate ratio (p. 34)
$\boldsymbol{r}$	subsequence of length L of the received data stream $\boldsymbol{y}$ (p. 94)
$\mathfrak{r}$	root of the phylogenetic tree (p. 37)
S	strong nucleotide (p. 19)
$s_{h=0}$	bifix-free syncword (p. 105)
$\boldsymbol{s}'$	syncword like pattern recognized by threshold synchronizer (p. 105)
$\boldsymbol{s}'$	syncword like pattern recognized by threshold synchronizer (p. 106)
t_{uv}	evolutionary distance between phylogenetic tree nodes u and v (p. 37)
T	thymine nucleotide (p. 18)
τ	branch lengths of the phylogenetic tree (p. 37)
θ_j	evolutionary rate heterogeneity of a nucleotide position (p. 39)
$\mathcal{T}$	phylogenetic tree (p. 37)
W	weak nucleotide (p. 19)
$\boldsymbol{x}_\ell$	homologous nucleotides at the leaves of the phylogenetic tree (p. 37)
$\hat{x}_\mathfrak{r}$	estimate of the common ancestor nucleotide (p. 37)
Y	pyrimidine nucleotide (p. 19)
$\boldsymbol{y}$	received datastream (p. 91)

Bibliography

[AGM+90] Altschul, S., Gish, W., Miller, W., Myers, E. and Lipman, D.: Basic local alignment search tool. In: *Journal of Molecular Biology* Volume 215, No. 3, 1990, pp. 403–410.

[AJW+08] Alberts, B., Johnson, A., Walter, P., Lewis, J., Raff, M. and Roberts, K.: *Molecular Biology of the Cell*. 5th edition. Taylor & Francis, 2008.

[AK01] Antos, A. and Kontoyiannis, I.: Convergence properties of functional estimates for discrete distributions. In: *Random Structures and Algorithms* Volume 19, No. 3-4, 2001, pp. 163–193.

[Bar53] Barker, R. H.: Group synchronization of binary digital systems. In: *Communication Theory* 1953, pp. 273–287.

[Bas59] Basharin, G. P.: On a statistical estimate for the entropy of a sequence of independent random variables. In: *Theory of Probability and its Applications* Volume 4, 1959, pp. 333.

[Bat97] Battail, G.: Does information theory explain biological evolution. In: *Europhysics Letters* Volume 40, No. 3, Nov. 1997, pp. 343–348.

[BBS02] Benos, P. V., Bulyk, M. L. and Stormo, G. D.: Additivity in protein-DNA interactions: how good an approximation is it? In: *Nucleic Acids Research* Volume 30, No. 20, 2002, pp. 4442–4451.

[BCW90] Bell, T., Cleary, J. and Witten, I.: *Text Compression*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1990.

[BD95] Bajic, D. and Drajic, D.: Duration of search for a fixed pattern in random data: distribution function and variance. In: *Electronics Letters* Volume 31, No. 8, 1995, pp. 631–632.

[BDC+03] Brudno, M., Do, C., Cooper, G., Kim, M., Davydov, E., Program, N., Green, E., Sidow, A. and Batzoglou, S.: LAGAN and multi-LAGAN: Efficient tools for large-scale multiple alignment of genomic DNA. In: *Genome Research* Volume 13, No. 4, 2003, pp. 721–731.

[BEY06] Begleiter, R. and El-Yaniv, R.: Superior guarantees for sequential prediction and lossless compression via alphabet decomposition. In: *Journal of Machine Learning Research* Volume 7, 2006, pp. 379–411.

[BHCC01] Bulyk, M. L., Huang, X., Choo, Y. and Church, G. M.: Exploring the DNA-binding specificities of zinc fingers with DNA microarrays. In: *Proceedings of the National Academy of Sciences* Volume 98, No. 13, 2001, pp. 7158–7163.

[BKML+10] Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J. and Sayers, E. W.: GenBank. In: *Nucleic Acids Research* Volume 38, No. suppl_1, 2010, pp. D46–51.

[BKR+04] Blanchette, M., Kent, W. J., Riemer, C., Elnitski, L., Smit, A. F., Roskin, K. M., Baertsch, R., Rosenbloom, K., Clawson, H., Green, E. D., Haussler, D. and Miller, W.: Aligning multiple genomic sequences with the threaded blockset aligner. In: *Genome Research* Volume 14, No. 4, Apr 2004, pp. 708–715.

[Bla03] Black, D. L.: Mechanisms of alternative pre-messenger rna splicing. In: *Annual Review of Biochemistry* Volume 72, 2003, pp. 291–336.

[Bla07] Blanchette, M.: Computation and analysis of genomic multi-sequence alignments. In: *Annual Review of Genomics and Human Genetics* Volume 8, 2007, pp. 193–213.

[Boe67] Boehmer, A.: Binary pulse compression codes. In: *IEEE Transactions on Information Theory* Volume 13, No. 2, 1967, pp. 156–167.

[Bos99] Bossert, M.: *Channel Coding for Telecommunications*. John Wiley & Sons, Inc. New York, NY, USA, 1999.

[BP04] Bray, N. and Pachter, L.: MAVID: Constrained ancestral alignment of multiple sequences. In: *Genome Research* Volume 14, No. 4, 2004, pp. 693–699.

[BPM+04] Bejerano, G., Pheasant, M., Makunin, I., Stephen, S., Kent, W. J., Mattick, J. S. and Haussler, D.: Ultraconserved elements in the human genome. In: *Science* Volume 304, No. 5675, May 2004, pp. 1321–5.

[BSV05] Bajic, D., Stefanovic, C. and Vukobratovic, D.: Search process and probabilistic bifix approach. In: *Proceedings of the International Symposium on Information Theory* 2005, pp. 19–22.

[BVK07] Bollenbach, T., Vetsigian, K. and Kishony, R.: Evolution and multilevel optimization of the genetic code. In: *Genome Research* Volume 17, No. 4, 2007, pp. 401–404.

[CAB+03] Clamp, M., Andrews, D., Barker, D., Bevan, P., Cameron, G., Chen, Y., Clark, L., Cox, T., Cuff, J., Curwen, V., Down, T., Durbin, R., Eyras,

E., Gilbert, J., Hammond, M., Hubbard, T., Kasprzyk, A., Keefe, D., Lehvaslaiho, H., Iyer, V., Melsopp, C., Mongin, E., Pettett, R., Potter, S., Rust, A., Schmidt, E., Searle, S., Slater, G., Smith, J., Spooner, W., Stabenau, A., Stalker, J., Stupka, E., Ureta-Vidal, A., Vastrik, I. and Birney, E.: Ensembl 2002: accommodating comparative genomics. In: *Nucl. Acids Res.* Volume 31, No. 1, 2003, pp. 38–42.

[CBS+04] Cooper, G. M., Brudno, M., Stone, E. A., Dubchak, I., Batzoglou, S. and Sidow, A.: Characterization of evolutionary rates and constraints in three mammalian genomes. In: *Genome Research* Volume 14, No. 4, Apr 2004, pp. 539–548.

[CDAM07] Cao, M., Dix, T., Allison, L. and Mears, C.: A simple statistical algorithm for biological sequence compression. In: *Proceedings of the Data Compression Conference*. 2007, pp. 43–52.

[Che07] Check, E.: Genome project turns up evolutionary surprises. In: *Nature* Volume 447, No. 14, Jun 2007, pp. 760–761.

[CKL01] Chen, X., Kwong, S. and Li, M.: A compression algorithm for DNA sequences. In: *IEEE Engineering in Medicine and Biology Magazine* Volume 20, No. 4, 2001, pp. 61–66.

[CLMT02] Chen, X., Li, M., Ma, B. and Tromp, J.: DNAcompress: fast and effective DNA sequence compression. In: *Bioinformatics* Volume 18, No. 12, 2002, pp. 1696–1698.

[Cre93] Creighton, T. E.: *Proteins, Structures and Molecular Properties*. New York: Freeman, 1993.

[CT06] Cover, T. M. and Thomas, J. A.: *Elements of Information Theory*. Wiley-Interscience, 2006.

[CW84] Cleary, J. G. and Witten, I. H.: Data compression using adaptive coding and partial string matching. In: *IEEE Transactions on Communications* Volume 32, No. 4, April 1984, pp. 396–402.

[Dar59] Darwin, C.: *On The Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. London: John Murray, 1859.

[DGH+06] Dawy, Z., Goebel, B., Hagenauer, J., Andreoli, C., Meitinger, T. and Mueller, J. C.: Gene mapping and marker clustering using Shannon's mutual information. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* Volume 3, No. 1, 2006, pp. 47–56.

[DH04] Down, T. A. and Hubbard, T. J. P.: What can we learn from noncoding regions of similarity between genomes? In: *BMC Bioinformatics* Volume 5, No. 1, Sep 2004, pp. 131.

[DHHM05] Dawy, Z., Hagenauer, J., Hanus, P. and Mueller, J. C.: Mutual information based distance measures for classification and content recognition with applications to genetics. In: *Proceedings of the IEEE International Conference on Communications (ICC05)*. May 2005, pp. 820–824.

[DHL+08] Dingel, J., Hanus, P., Leonardi, N., Hagenauer, J., Zech, J. and Mueller, J.: Local conservation scores without a priori assumptions on neutral substitution rates. In: *BMC Bioinformatics* Volume 9, No. 1, 2008, pp. 190.

[DMA+01] Deloukas, P., Matthews, L., Ashurst, J., Burton, J., Gilbert, J., Jones, M., Stavrides, G., Almeida, J., Babbage, A., Bagguley, C. et al.: The DNA sequence and comparative analysis of human chromosome 20. In: *Nature* Volume 414, No. 6866, 2001, pp. 865–871.

[Don03] Donaill, D.: Why nature chose A, C, G and U/T: An error-coding perspective of nucleotide alphabet composition. In: *Origins of Life and Evolution of Biospheres* Volume 33, No. 4, 2003, pp. 433–455.

[DPA+07] Dix, T., Powell, D., Allison, L., Bernal, J., Jaeger, S. and Stern, L.: Comparative analysis of long DNA sequences by per element information content using different contexts. In: *BMC Bioinformatics* Volume 8, No. Suppl 2, 2007, pp. S10.

[DRS+03] Dermitzakis, E., Reymond, A., Scamuffa, N., Ucla, C., Kirkness, E., Rossier, C. and Antonarakis, S.: Evolutionary discrimination of mammalian conserved non-genic sequences (CNGs). In: *Science* Volume 302, No. 5647, 2003, pp. 1033–1035.

[FC96] Felsenstein, J. and Churchill, G. A.: A Hidden Markov Model approach to variation among sites in rate of evolution. In: *Molecular Biology and Evolution* Volume 13, No. 1, Jan 1996, pp. 93–104.

[Fel81] Felsenstein, J.: Evolutionary trees from DNA sequences: A maximum likelihood approach. In: *Journal of Molecular Evolution* Volume 17, No. 6, 1981, pp. 368–376.

[FKC+02] Frit, P., Kwon, K., Coin, F., Auriol, J., Dubaele, S., Salles, B. and Egly, J. M.: Transcriptional activators stimulate DNA repair. In: *Molecular Cell* Volume 10, No. 6, December 2002, pp. 1391–401.

[FWK03] Freeland, S., Wu, T. and Keulmann, N.: The case for an error minimizing standard genetic code. In: *Origins of Life and Evolution of the Biosphere* Volume 33, No. 4-5, 2003, pp. 457–477.

[FWS95] Friedberg, E., Walker, G. and Siede, W.: *DNA Repair and Mutagenesis*. American Society for Microbiology, 1995.

[GACE+00] Gur-Arie, R., Cohen, C., Eitan, Y., Shelef, L., Hallerman, E. and Kashi, Y.: Simple sequence repeats in Escherichia coli: abundance, distribution, composition, and polymorphism. In: *Genome Research* Volume 10, No. 1, 2000, pp. 62–71.

[GCJJPG+08] Gama-Castro, S., Jimenez-Jacinto, V., Peralta-Gil, M., Santos-Zavaleta, A., Penaloza-Spinola, M., Contreras-Moreira, B., Segura-Salazar, J., Muniz-Rascado, L., Martinez-Flores, I., Salgado, H. et al.: RegulonDB (version 6.0): gene regulation model of Escherichia coli K-12 beyond transcription, active (experimental) annotated promoters and Textpresso navigation. In: *Nucleic Acids Research* Volume 36, January 2008, pp. D120–4.

[GDHM05] Goebel, B., Dawy, Z., Hagenauer, J. and Mueller, J. C.: An approximation to the distribution of finite sample size mutual information estimates. In: *Proceedings of IEEE International Conference on Communications*, . Volume 2. May 2005, pp. 1102–1106.

[GG08] Gorman, J. and Greene, E.: Visualizing one-dimensional diffusion of proteins along DNA. In: *Nature* Volume 15, No. 8, 2008, pp. 768–774.

[GMS02] Greaves, D. J. and Montgomery-Smith, S. J.: Unforgeable marker sequences, 2002.

[GMW81] Gill, P., Murray, W. and Wright, M.: *Practical Optimization*. London: Academic Press, 1981.

[GR07] Gruenwald, P. D. and Rissanen, J.: *The Minimum Description Length Principle*. MIT Press Cambridge, MA, USA, 2007.

[GT93] Grumbach, S. and Tahi, F.: Compression of DNA sequences. In: *Proceedings of the Data Compression Conference*. 1993, pp. 340–350.

[GT94] Grumbach, S. and Tahi, F.: A new challenge for compression algorithms: Genetic sequences. In: *Journal on Information Processing and Management* Volume 30, No. 6, 1994, pp. 875–886.

[HAA+09] Hubbard, T., Aken, B., Ayling, S., Ballester, B., Beal, K., Bragin, E., Brent, S., Chen, Y., Clapham, P., Clarke, L. et al.: Ensembl 2009. In: *Nucleic Acids Research* Volume 36, 2009, pp. D690–7.

[HAC+92] Hampel, H., Arps, R., Chamzas, C., Dellert, D., Duttweiler, D., Endoh, T., Equitz, W., Ono, F., Pasco, R., Sebestyen, I., Starkey, C. J., Urban, S. J., Yamazaki, Y. and Yoshida, T.: Technical features of the JBIG standard for progressive bi-level image compression: Audio, visual and multimedia/hypermedia coding standardisation(ISO standard). In: *Signal processing. Image communication* Volume 4, No. 2, 1992, pp. 103–111.

[Hae01] Haensler, E.: *Statistische Signale: Grundlagen und Anwendungen*. Springer, 2001.

[Hag04] Hagenauer, J.: The EXIT chart - introduction to extrensic information transfer in iterative processing. In: *Proceedings of the European Signal Processing Conference*. September 2004.

[Hay98] Hayes, B.: The Invention of the Genetic Code. In: *American Scientist* Volume 86, No. 1, 1998, pp. 8–14.

[HBB+02] Hubbard, T., Barker, D., Birney, E., Cameron, G., Chen, Y., Clark, L., Cox, T., Cuff, J., Curwen, V., Down, T., Durbin, R., Eyras, E., Gilbert, J., Hammond, M., Huminiecki, L., Kasprzyk, A., Lehvaslaiho, H., Lijnzaad, P., Melsopp, C., Mongin, E., Pettett, R., Pocock, M., Potter, S., Rust, A., Schmidt, E., Searle, S., Slater, G., Smith, J., Spooner, W., Stabenau, A., Stalker, J., Stupka, E., Ureta-Vidal, A., Vastrik, I. and Clamp, M.: The Ensembl genome database project. In: *Nucleic Acids Research* Volume 30, No. 1, 2002, pp. 38–41.

[HDCH09] Hanus, P., Dingel, J., Chalkidis, G. and Hagenauer, J.: Source coding scheme for multiple sequence alignments. In: *Proceedings of the Data Compression Conference (DCC09)*. March 2009, pp. 183–192.

[HDCH10] Hanus, P., Dingel, J., Chalkidis, G. and Hagenauer, J.: Compression of whole genome alignments. In: *IEEE Transactions on Information Theory* Volume 56, No. 2, 2010, pp. 696–705.

[HDG+04] Hagenauer, J., Dawy, Z., Gobel, B., Hanus, P. and Mueller, J.: Genomic analysis using methods from information theory. In: *Proceedings of the IEEE Information Theory Workshop (ITW04)*. October 2004, pp. 55–59.

[HRY+03] Hardison, R. C., Roskin, K. M., Yang, S., Diekhans, M., Kent, W. J., Weber, R., Elnitski, L., Li, J., O'Connor, M., Kolbe, D., Schwartz, S., Furey, T. S., Whelan, S., Goldman, N., Smit, A., Miller, W., Chiaromonte, F. and Haussler, D.: Covariation in frequencies of substitution, deletion, transposition, and recombination during eutherian evolution. In: *Genome Research* Volume 13, No. 1, Jan 2003, pp. 13–26.

[Huf52] Huffman, D. A.: A method for the construction of minimum redundancy codes. In: *Procedings of the Institute of Radio Engineers*, . Volume 40. September 1952, pp. 1098–1101.

[HW06] Hanus, P. and Weindl, J.: Synchronization model of transcription initiation in prokaryotes. In: *Proceedings of the International Conference on Intelligent Systems for Molecular Biology Student Council Symposium (ISMB06)*. August 2006.

[IA07] Itzkovitz, S. and Alon, U.: The genetic code is nearly optimal for allowing additional information within protein-coding sequences. In: *Genome Research* Volume 17, No. 4, Apr 2007, pp. 405–412.

[JBCR74] Jelinek, F., Bahl, L., Cocke, J. and Raviv, J.: Optimal decoding of linear codes for minimizing symbol error rate. In: *IEEE Transactions on Information Theory* Volume 20, No. 2, 1974, pp. 284–287.

[KB04] Kopansky, A. and Bystrom, M.: Detection of aperiodically embedded synchronization patterns. In: *IEEE Transactions on Wireless Communications* Volume 3, No. 5, 2004, pp. 1386–1392.

[KBD+03] Karolchik, D., Baertsch, R., Diekhans, M., Furey, T. S., Hinrichs, A., Lu, Y. T., Roskin, K. M., Schwartz, M., Sugnet, C. W., Thomas, D. J., Weber, R. J., Haussler, D. and Kent, W. J.: The UCSC genome browser database. In: *Nucleic Acids Research* Volume 31, No. 1, Jan 2003, pp. 51–54.

[Kel79] Kelly, F. P.: *Reversibility and Stochastic Networks*. John Wiley & Sons, 1979.

[KKZ+06] Kuhn, R., Karolchik, D., Zweig, A., Trumbower, H., Thomas, D., Thakkapallayil, A., Sugnet, C., Stanke, M., Smith, K., Siepel, A. et al.: The UCSC genome browser database: update 2007. In: *Nucleic Acids Research* Volume 35, 2006, pp. D668–D673.

[Kni06] Knippers, R.: *Molekulare Genetik*. 9th edition. Georg Thieme Verlag, 2006.

[Kra49] Kraft, L.: *A device for quantizing, grouping, and coding amplitude-modulated pulses*. Master's Thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering, 1949.

[KT81] Krichevsky, R. and Trofimov, V.: The performance of universal encoding. In: *IEEE Transactions on Information Theory* Volume 27, No. 2, Mar 1981, pp. 199–207.

[Kul59] Kullback, S.: *Information theory and statistics*. New York: John Wiley & Sons, 1959.

[LAB+93] Lawrence, C., Altschul, S., Boguski, M., Liu, J., Neuwald, A. and Wootton, J.: Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. In: *Science* Volume 262, No. 5131, 1993, pp. 208–214.

[LBC+01] Li, M., Badger, J., Chen, X., Kwong, S., Kearney, P. and Zhang, H.: An information-based sequence distance and its application to whole mitochondrial genome phylogeny. In: *Bioinformatics* Volume 17, No. 2, 2001, pp. 149–154.

[Lew04] Lewin, B.: *Genes VIII*. Pearson Prentice Hall, 2004.

[LG76] Lambden, P. R. and Guest, J. R.: Mutants of Escherichia coli K12 unable to use fumarate as an anaerobic electron acceptor. In: *Journal of General Microbiology* Volume 97, No. 2, 1976, pp. 145–160.

[LLB+01] Lander, E., Linton, L., Birren, B., Nusbaum, C. and Zody, M.: Initial sequencing and analysis of the human genome. In: *Nature* Volume 409, 2001, pp. 860–921.

[LT87] Lui, G. and Tan, H.: Frame synchronization for Gaussian channels. In: *IEEE Transactions on Communications* Volume 35, No. 8, 1987, pp. 818–829.

[Mac02] Mac Donaill, A.: A parity code interpretation of nucleotide alphabet composition. In: *Chemical Communications* Volume 18, September 2002, pp. 2062–2063.

[Mas72] Massey, J.: Optimum frame synchronization. In: *IEEE Transactions on Communications* Volume 20, No. 2, 1972, pp. 115–119.

[Mat07] Mattick, J.: A new paradigm for developmental biology. In: *Journal of Experimental Biology* Volume 210, No. 9, 2007, pp. 1526.

[Men86] Mendel, G.: Versuche über Pflanzen-hybriden. In: *Verhandlugen des Naturforschenden Vereines in Brünn* Volume 4, 1886, pp. 3–47.

[MFP05] Mayrose, I., Friedman, N. and Pupko, T.: A Gamma mixture model better accounts for among site rate heterogeneity. In: *Bioinformatics* Volume 21 Suppl 2, No. suppl_2, Sep 2005, pp. ii151–158.

[MGH+03] Murray, R., Granner, D., Harper, H., Mayes, P. and Rodwell, V.: *Harper's Illustrated Biochemistry*. McGraw-Hill Medical, 2003.

[Mil55] Miller, G.: Note on the bias of information estimates. In: *Information Theory in Psychology: Problems and Methods* 1955, pp. 95–100.

[MM06] Mattick, J. and Makunin, I.: Non-coding RNA. In: *Human Molecular Genetics* Volume 15, No. 90001, 2006, pp. 17–29.

[MRH+07] Miller, W., Rosenbloom, K., Hardison, R. C., Hou, M., Taylor, J., Raney, B., Burhans, R., King, D. C., Baertsch, R., Blankenberg, D., Kosakovsky Pond, S. L., Nekrutenko, A., Giardine, B., Harris, R. S., Tyekucheva, S., Diekhans, M., Pringle, T. H., Murphy, W. J., Lesk, A., Weinstock, G. M., Lindblad-Toh, K., Gibbs, R. A., Lander, E. S., Siepel, A., Haussler, D. and Kent, W. J.: 28-Way vertebrate alignment and conservation track in the UCSC Genome Browser. In: *Genome Research* Volume 17, No. 12, 2007, pp. 1797–1808.

[MS64] Maury, J. L. and Styles, F. J.: Development of optimum frame synchronization codes for Goddard Space Flight Center PCM Telemetry Standards. In: *Proceedings of the National Telemetering Conference* 1964.

[MS01] Man, T. and Stormo, G.: Non-independence of Mnt repressor–operator interaction determined by a new quantitative multiple fluorescence relative

affinity (QuMFRA) assay. In: *Nucleic Acids Research* Volume 29, No. 12, 2001, pp. 2471–2478.

[MV04] Milenkovic, O. and Vasic, B.: Information theory and coding problems in genetics. In: *Proc of the International Workshop on Information Theory (ITW04)*. Oct 2004, pp. 60–65.

[Nat08] National Institutes of Health, National Human Genome Research Institute: Talking glossary of genetic terms. http://genome.gov/glossary, 2008.

[NFN08] Nishida, K., Frith, M. and Nakai, K.: Pseudocounts for transcription factor binding sites. In: *Nucleic Acids Research* 2008.

[NH71] Neuman, F. and Hofman, L.: New pulse sequences with desirable correlation properties. In: *Proceedings of the National Telemetering Conference* 1971, pp. 272–282.

[Nie73a] Nielsen, P.: On the expected duration of a search for a fixed pattern in random data (Corresp.). In: *IEEE Transactions on Information Theory* Volume 19, No. 5, 1973, pp. 702–704.

[Nie73b] Nielsen, P.: Some optimum and suboptimum frame synchronizers for binary data in Gaussian noise. In: *IEEE Transactions on Communications* Volume 21, No. 6, 1973, pp. 770–772.

[Nie97] Nielsen, R.: Site-by-site estimation of the rate of substitution and the correlation of rates in mitochondrial DNA. In: *Systematic Biology* Volume 46, No. 2, Jun 1997, pp. 346–353.

[Nor97] Norris, J.: *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.

[Och03] Ochman, H.: Neutral mutations and neutral substitutions in bacterial genomes. In: *Molecular Biology and Evolution* Volume 20, No. 12, 2003, pp. 2091–2096.

[Par06] Pardo, L.: *Statistical inference based on divergence measures*, . Volume 185 of *Statistics: Textbooks and Monographs*. Chapman & Hall/CRC, 2006.

[Pea82] Pearl, J.: Reverend Bayes on inference engines: A distributed hierarchical approach. In: *Proc of the AAAI National Conference on AI* 1982, pp. 133–136.

[PH99] Petrov, D. and Hartl, D.: Patterns of nucleotide substitution in Drosophila and mammalian genomes, 1999.

[PS84] Pabo, C. O. and Sauer, R. T.: Protein-DNA recognition. In: *Annual Review of Biochemistry* Volume 53, No. 1, 1984, pp. 293–321.

[Rad01] Radman, M.: Fidelity and infidelity. In: *Nature* Volume 413, No. 6852, September 2001, pp. 115.

[RDDD96] Rivals, E., Delahaye, J.-P., Dauchet, M. and Delgrange, O.: A guaranteed compression scheme for repetitive DNA sequences. In: *Proceedings of the Data Compression Conference*. March 1996, pp. 453.

[Ris76] Rissanen, J.: Generalized kraft inequality and arithmetic coding. In: *IBM Journal of Research and Development* Volume 20, No. 3, 1976, pp. 198–203.

[RL79] Rissanen, J. and Langdon, G.: Arithmetic coding. In: *IBM Journal of Research and Development* Volume 23, No. 2, March 1979, pp. 149–162.

[Rob95] Robertson, P.: *Optimal Frame Synchronization for Continuous and Packet Data Transmission*. VDI–Verlag, 1995.

[SBS93] Shaner, M. C., Blair, I. M. and Schneider, T. D.: Sequence logos: a powerful, yet simple, tool. In: *Proceedings of the International Conference on System Sciences*, . Volume 1. 1993, pp. 813–821.

[Sch80] Scholtz, R.: Frame synchronization techniques. In: *IEEE Transactions on Communications* Volume 28, No. 8 Part 2, 1980, pp. 1204–1213.

[SF98] Stormo, G. and Fields, D.: Specificity, free energy and information content in protein–DNA interactions. In: *Trends in Biochemical Sciences* Volume 23, No. 3, 1998, pp. 109–113.

[SGD+07] Sarkis, M., Goebel, B., Dawy, Z., Hagenauer, J., Hanus, P. and Mueller, J. C.: Gene mapping of complex diseases-A comparison of methods from statistics information theory, and signal processing. In: *IEEE Signal Processing Magazine* Volume 24, No. 1, 2007, pp. 83–90.

[Sha48] Shannon, C. E.: A mathematical theory of communication. In: *Bell Systems Technical Journal* Volume 27, July and October 1948, pp. 379–423 and 623–656.

[Spr02] Spring, J.: Genome duplication strikes back. In: *Nature Genetics* Volume 31, No. 2, 2002, pp. 128–130.

[SRR76] Seeman, N., Rosenberg, J. and Rich, A.: Sequence-specific recognition of double helical nucleic acids by proteins. In: *Proceedings of the National Academy of Sciences* Volume 73, No. 3, 1976, pp. 804–808.

[SRS05] Szpankowski, W., Ren, W. and Szpankowski, L.: An optimal DNA segmentation based on the MDL principle. In: *International Journal of Bioinformatics Research and Applications* Volume 1, No. 1, 2005, pp. 3–17.

[SS82] Storer, J. and Szymanski, T.: Data compression via textual substitution. In: *Journal of the Association for Computing Machinery* Volume 29, No. 4, 1982, pp. 928–951.

[SS90] Schneider, T. and Stephens, R.: Sequence logos: a new way to display consensus sequences. In: *Nucleic Acids Res* Volume 18, No. 20, 1990, pp. 6097–6100.

[SSGE86] Schneider, T. D., Stormo, G. D., Gold, L. and Ehrenfeucht, A.: Information content of binding sites on nucleotide sequences. In: *Journal of Molecular Biology* Volume 188, No. 3, 1986, pp. 415–431.

[SSS04] Sakata-Sogawa, K. and Shimamoto, N.: RNA polymerase can track a DNA groove during promoter search. In: *Proceedings of the National Academy of Sciences* Volume 101, No. 41, 2004, pp. 14731–14735.

[SSW93] Shannon, C. E., Sloane, N. J. A. and Wyner, A. D.: *Claude Elwood Shannon: Collected Papers*. IEEE Press Piscataway, NJ, USA, 1993.

[Sto00] Stormo, G.: DNA binding sites: representation and discovery. In: *Bioinformatics* Volume 16, No. 1, 2000, pp. 16–23.

[SW81] Smith, T. and Waterman, M.: Identification of common molecular subsequences. In: *Journal of Molecular Biology* Volume 147, No. 1, 1981, pp. 195–197.

[TAACT04] Touchon, M., Arneodo, A., Aubenton-Carafa, Y. and Thermes, C.: Transcription-coupled and splicing-coupled strand asymmetries in eukaryotic genomes. In: *Nucleic Acids Research* Volume 32, No. 17, 2004, pp. 4969–4978.

[TBS09] Tymoczko, J., Berg, J. and Stryer, L.: *Biochemistry: A Short Course*. WH Freeman & Co, 2009.

[TH02] Tuechler, M. and Hagenauer, J.: EXIT charts and irregular codes. In: *Annual Conference on Information Sciences and Systems*. 2002, pp. 748–753.

[TO07] Tomovic, A. and Oakeley, E.: Position dependencies in transcription factor binding sites. In: *Bioinformatics* Volume 23, No. 8, 2007, pp. 933.

[TVW97] Tjalkens, T. J., Volf, P. A. J. and Willems, F. M. J.: A context-tree weighting method for text generating sources. In: *In Proceedings of the Data Compression Conference*. Mar 1997, pp. 472.

[UVEB03] Ureta-Vidal, A., Ettwiller, L. and Birney, E.: Comparative genomics: genome-wide analysis in metazoan eukaryotes. In: *Nature Reviews Genetics* Volume 4, No. 4, 2003, pp. 251–262.

[WC53] Watson, J. D. and Crick, F. H. C.: Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. In: *Nature* Volume 171, 1953, pp. 737–738.

[WDKK96] Wingender, E., Dietze, P., Karas, H. and Kmrppel, R.: TRANSFAC: a database on transcription factors and their DNA binding sites. In: *Nucleic Acids Research* Volume 24, 1996, pp. 238–241.

[WDT+80] Wing, R., Drew, H., Takano, T., Broka, C., Tanaka, S., Itakura, K. and Dickerson, R. E.: Crystal structure analysis of a complete turn of B-DNA. In: *Nature* Volume 287, No. 5784, October 1980, pp. 755–758.

[Wik08] Wikipedia: Wikimedia commons — wikipedia, the free encyclopedia, 2008.

[Wil62] Williard, M. W.: Optimum code patterns for PCM synchronization. . Volume 5 of *Proceedings of the National Telemetering Conference*. 1962, pp. 1–5.

[WKW90] Woese, C., Kandler, O. and Wheelis, M.: Towards a Natural System of Organisms: Proposal for the Domains Archaea, Bacteria, and Eucarya. In: *Proceedings of the National Academy of Sciences* Volume 87, No. 12, 1990, pp. 4576–4579.

[WS04] Wasserman, W. and Sandelin, A.: Applied bioinformatics for the identification of regulatory elements. In: *Nature Reviews Genetics* Volume 5, No. 4, 2004, pp. 276–287.

[WST95] Willems, F. M. J., Shtarkov, Y. M. and Tjalkens, T. J.: The context-tree weighting method: Basic properties. In: *IEEE Transactions on Information Theory* Volume 41, No. 3, May 1995, pp. 653–664.

[Yan06] Yang, Z.: *Computational Molecular Evolution*. Oxford Series in Ecology and Evolution. Oxford University Press, 2006.

[Zha03] Zhang, J.: Evolution by gene duplication: an update. In: *Trends in Ecology and Evolution* Volume 18, No. 6, 2003, pp. 292–298.

[ZL77] Ziv, J. and Lempel, A.: A universal algorithm for sequential data compression. In: *IEEE Transactions on Information Theory* Volume 23, No. 3, May 1977, pp. 337–343.

[ZL78] Ziv, J. and Lempel, A.: Compression of individual sequences via variable-rate coding. In: *IEEE Transactions on Information Theory* Volume 24, No. 5, Sept. 1978, pp. 530–536.