Learning Inhomogeneous Parsimonious Markov Models with Application to DNA Sequence Analysis

Dissertation

zur Erlangung des

Doktorgrades der Naturwissenschaften (Dr. rer. nat.)

der

Naturwissenschaftlichen Fakultät III
Agrar- und Ernährungswissenschaften,
Geowissenschaften und Informatik

der Martin–Luther–Universität Halle–Wittenberg,

vorgelegt von

Herrn Ralf Eggeling
Geb. am 14. Februar 1983 in Magdeburg

Gutachter:
Prof. Dr. Ivo Grosse
Prof. Dr. Jesús Cerquides Bueno

Datum der Verteidigung: 20.11.2014

Abstract

Short oligonucleotides that are bound by regulator proteins play a pivotal role in gene regulation and are thus crucial for the complexity of life. Statistical modeling of these binding sites of a particular protein, i.e., inferring a sequence motif, with the incentive of predicting new instances, is one of the classic fields within bioinformatics. Most of the previous work is based on a comparatively simple model that assumes statistical independence among all nucleotides within the motif, which corresponds to the assumption of that the contribution of each nucleotide to binding affinity is additive.

Whereas there are indications that these assumption may be a simplified reflection of the natural process of protein-DNA binding, making use of additional features is to date limited by insufficient statistical models. Many approaches that are capable of taking into account complex features such as dependencies among adjacent nucleotides require an exponentially growing number of parameters. Attempting to fit a large number of parameters from a small number of observations inevitably leads to overfitting.

Here, we propose a new class of statistical models that allows modeling complex features in the data while simultaneously keeping the parameter space small in order to avoid overfitting. Using recently proposed parsimonious context trees, *inhomogeneous parsimonious Markov models* (PMMs) allow a fine grained adjustment of the number of parameters in principle.

Learning these models, which involves dealing with a variable structure and variable dimensionality of the parameter space, is challenging, though. Moreover, tasks like de novo motif discovery require learning a model in the presence of latent variables, which adds another layer of complexity to the learning problem. We thus propose and investigate different Bayesian and non-Bayesian approaches for learning PMMs, both from fully observable data and in the presence of latent variables. By case studies on several real-world data sets we investigate the advantages and drawbacks of the different approaches from computational point of view, and we find that robust learning of PMMs is indeed possible by methods inspired by the Minimum Description Length principle.

From biological point of view, we investigate the phenomenon of statistical dependencies within sequence motifs of DNA-binding proteins by a second set of case studies. Focusing on the enhancer-binding insulator protein CTCF in detail, and additionally investigating ChIP-seq data of fifty transcription factors in a broader study, we find that intra-motif dependencies are prevalent in nature, being rather the rule than the exception. We conclude that the most frequently used motif model to date is indeed oversimplified, but also find first-order dependencies among adjacent nucleotides to be the dominant additional feature.

Acknowledgments

The research leading to this dissertation was mainly conducted in the *bioinformatics* group at the computer science department of the Martin Luther University Halle–Wittenberg. In addition, I spent about ten months as visiting researcher in the *Complex Systems Computations (CoSCo)* research group of the Helsinki Institute for Information Technology (HIIT) at the University of Helsinki. I thank both institutions for providing an open, friendly, and efficient work environment. This work was funded by *Reisestipendium des allg. Stiftungsfonds der MLU Halle Wittenberg* and DFG (grant no. GR3526_1-1).

I am very grateful to my advisor Ivo Grosse, who guided me through all those years, indulged in enthusiastic scientific discussions that sometimes did last until Friday midnight, taught me to write papers, and organized funding. I am also grateful to Pierre-Yves Bourguignon for many important discussions pertaining parsimonious Markov models and the world of Bayesian inference in general. Furthermore, I owe my deepest gratitude to Petri Myllymäki and Teemu Roos for hosting me at HIIT, giving me the opportunity to work in the CoSCo group, and introducing me to the information-theoretic approach of statistical modeling. Obtaining a new perspective both on precise scientific questions and general worklife at a different institute in a different country proved to be (i) highly beneficial to this dissertation, and (ii) a lot of fun.

In addition, I thank Jan Grau and Jens Keilwagen for the development of the open source Java library Jstacs [1], which most implementations of this thesis are based on. I also thank all co-authors on publications that cover topics of this thesis, all members of the research groups *bioinformatics* and *bioinformatics and pattern recognition* at the Martin Luther University, and all members of the *CoSCo* research group at HIIT for countless valuable discussions.

Finally I thank my parents for giving birth to me – certainly a non-negligible contribution to this dissertation, and my entire family for their continuous support during the past three decades.

---

[1] `http://www.jstacs.de`

Published thesis contents

While this thesis is not strictly article-based, several parts of the content have been previously published in peer-reviewed journals and conference proceedings.

The content of Section 1.4, in particular the English-language presentation of the dynamic programming algorithm for finding optimal parsimonious context trees and the theoretical comparison of parsimonious and traditional context trees, as well as the majority of the content of Chapter 2 were published as conference paper *Inhomogeneous Parsimonious Markov Models* in the proceedings of ECMLPKDD 2013 [1]. I hereby declare that the copyright of the content of that paper is by Springer, available at:

`http://link.springer.com/chapter/10.1007/978-3-642-40988-2_21`

The majority of the content of Chapter 3 was published as conference paper *Robust learning of inhomogeneous PMMs* in the proceedings of AISTATS 2014 [2]. The content of Chapter 5 was published as paper *On the value of intra-motif dependencies of human insulator protein CTCF* in the journal PLOS ONE [3]. The majority of the content of Chapter 6 was published as conference paper *Gibbs sampling for parsimonious Markov models with latent variables* in the proceedings of PGM 2012 [4]. The key idea of Chapter 7 was presented at WITMSE 2013 and an extended abstract titled *Model selection in a setting with latent variables* was published in the corresponding proceedings [5].

Contents

1. Introduction **1**
1.1. Biological background 1
1.1.1. Transcription factor binding sites 2
1.1.2. Experimental detection of protein-DNA interaction 3
1.1.3. Splicing and splice sites 4
1.2. Computational challenges 5
1.2.1. Binding site recognition 5
1.2.2. De novo motif discovery 6
1.2.3. Splice site recognition 6
1.3. Models for sequence patterns 7
1.3.1. Position weight matrix model 7
1.3.2. Markov models 8
1.3.3. Bayesian networks 10
1.4. Parsimonious context trees 12
1.4.1. Definition 13
1.4.2. Finding optimal PCTs 13
1.4.3. Special cases 15
1.5. Objectives and outline of the thesis 16
1.5.1. Main objectives 16
1.5.2. Outline 17

2. Inhomogeneous PMMs **21**
2.1. Motivation 21
2.2. Model specification 22
2.2.1. Likelihood 23
2.2.2. Prior 23
2.2.3. Learning 24
2.3. Performance evaluation 25
2.3.1. Comparing prediction performance 26
2.3.2. Comparing tree structures 27

2.3.3. Model validation 30

3. Robust learning **33**
3.1. The MDL principle 34
3.1.1. Codelengths and probabilities 34
3.1.2. Two-part code 36
3.1.3. Normalized Maximum Likelihood distribution 36
3.2. Alternative learning approaches for PMMs 38
3.2.1. Structure scores 38
3.2.2. Parameter estimates 39
3.3. Empirical comparison 40
3.3.1. Influence of the ESS 40
3.3.2. Influence of the sample size 42
3.3.3. Comparing prediction performances 45
3.3.4. Running time considerations 47

4. Latent variable models **49**
4.1. Mixture models for DNA binding sites 49
4.1.1. Variables and parameters 50
4.1.2. Likelihood 50
4.1.3. Prior 51
4.2. Promoter models 51
4.2.1. Variables and parameters 52
4.2.2. Likelihood 53
4.2.3. Alternative definition of latent variables 54
4.3. Learning in the presence of latent variables 54
4.3.1. Parameter Learning 55
4.3.2. Structure learning 55
4.3.3. EM algorithm for motif discovery using PMMs 56

5. Dependencies within CTCF binding sites **59**
5.1. Insulator-binding protein CTCF 59
5.1.1. CTCF biology 59
5.1.2. Data sets 60
5.2. Classification experiment 60
5.2.1. Fragment-based classification 61
5.2.2. Finding optimal model complexity 62
5.2.3. Test on independent data and comparison with alternative models . . 63
5.2.4. Different cell lines 64

5.3. Motif analysis 66
5.3.1. Sequence logo 66
5.3.2. Mutual information 67
5.3.3. Optimal PCTs 68
5.3.4. PCTs and MI values 69
5.3.5. Conditional sequence logos 69

6. Model averaging with latent variables 75
6.1. Model averaging for mixture models 76
6.1.1. Gibbs sampling for mixtures of PMMs 77
6.1.2. Case studies 80
6.2. Model averaging for motif discovery 83
6.2.1. Algorithm 84
6.2.2. Case studies 84

7. Model selection with latent variables 89
7.1. Model selection for mixture model components 91
7.1.1. Optimization problem and algorithm 91
7.1.2. Case studies 93
7.2. Model selection within motif discovery 96
7.2.1. Algorithm 96
7.2.2. Case study 98

8. ENCODE ChIP-seq data analysis 101
8.1. Methods 101
8.1.1. ENCODE ChIP-seq data 101
8.1.2. Classification study 102
8.1.3. Binding site analysis 103
8.2. Results and discussion 103
8.2.1. Categorization of data sets 103
8.2.2. Classification results 106

9. Conclusions 115
9.1. Summary and lessons learned 115
9.1.1. Computational aspects 116
9.1.2. Biological aspects 118
9.2. Possible future projects 119
9.2.1. More sequence analysis 119
9.2.2. Machine learning 120

A. DP algorithm for PCT maximization 131

B. Additional derivations 135
B.1. Bayesian marginal likelihood score for PMMs 135
B.2. Factorized Sequential NML 136
B.3. EM algorithm for motif discovery 137

C. Additional results 141
C.1. CTCF study 141
C.2. ENCODE ChIP-seq study 143

1

Introduction

> *Bernard of Chartres used to compare us to dwarfs perched on the shoulders of giants. He pointed out that we see more and farther than our predecessors, not because we have greater height, but because we are lifted up and borne aloft on their gigantic stature.*
>
> John of Salisbury

In this thesis, we are interested in interdisciplinary questions at the borderline of molecular biology, bioinformatics, machine learning, and statistics. The precise topic comprises statistical modeling of short, functional oligonucleotides that play a pivotal role in gene regulation and are thus crucial for the complexity of life.

Such an interdisciplinary topic would of course be impossible to work on without relying on a substantial amount of previous work in different fields. In this chapter, we thus attempt to lay foundations for the following work by recapitulating knowledge that is fundamental for all chapters of the thesis. Additional concepts, which are required only for particular chapters, are introduced when needed for the first time.

First of all, we provide a basic introduction into the biological mechanisms of interest (Section 1.1) and an overview of the computational tasks that arise from different biological questions (Section 1.2). Afterwards, we discuss statistical models from literature that have been previously used for modeling short oligonucleotides (Section 1.3) and introduce parsimonious context trees (Section 1.4), which are the central data structure, this thesis is concerned about. Finally, we provide a guide through the main content of the thesis by describing the general scope of the research topics covered and giving a detailed outline of the different chapters (Section 1.5).

1.1. Biological background

Transcription, the synthesis of RNA from a DNA template, is the first step of gene expression (Figure 1.1). Transcription is catalyzed by the enzyme RNA polymerase in assembly with an entire complex of *basal transcription factors*, which have catalytic and stabilizing functions.

However, even though RNA polymerase and the basal cofactors are abundant in living cells, not every gene is transcribed at any time at the same rate during the life cycle of an organism. The complexity and flexibility of life itself, whether it may be the development of highly specialized types of tissues in multicellular organisms or the ability of swift adaptation to rapid changes of environmental conditions of bacteria, roots in differential expression of the same genomic information under certain conditions, which is regulated on different levels.

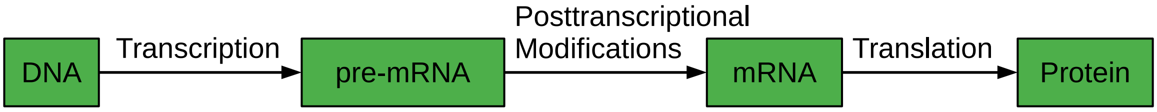


Figure 1.1.: **General pathway of gene expression in eukaryotes.** First, DNA is transcribed into pre-mRNA, catalyzed by an RNA-polymerase complex. After post-transcriptional modifications (splicing, editing, polyadenylation), mature mRNA is translated into a protein.

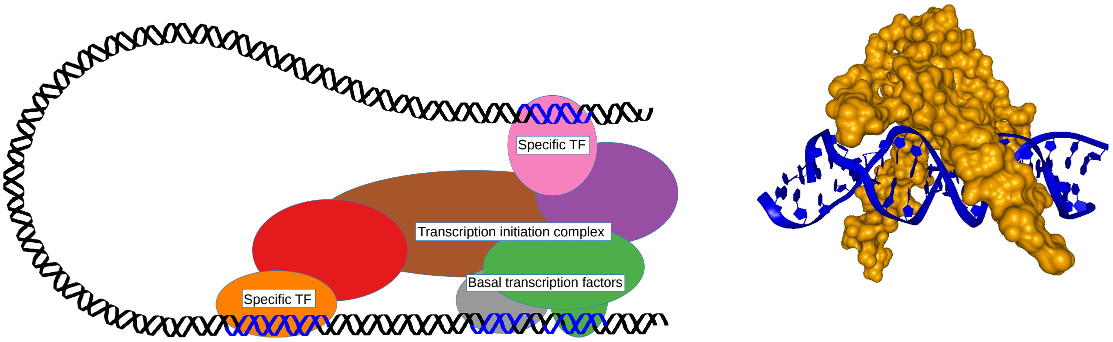
Gene silencing due to condensing of euchromatin into heterochromatin is an example for an epigenetic regulation that inhibits access to a particular genomic region [6]. Increased degradation of the mRNA of a specific gene is a way of post-transcriptional silencing by microRNAs and siRNAs [7]. In addition, regulation of translation also occurs [8].

The best understood principle of gene regulation, however, is the regulation of transcriptional initiation by regulating the proper formation of the transcription initiation complex, which is illustrated in Figure 1.2(a). This complex of RNA polymerase and basal transcription factors binds to a specific region of the DNA, which is called core promoter. The most common core promoter in eukaryotes is the well-known TATA-box and the binding transcription factor is the TATA binding protein (TBP). However, the affinity of this transcription preinitiation complex is usually rather low, leading to little or no significant gene expression.

In addition to basal transcription factors there are *specific transcription factors*, which may act as activators or repressors. Activators can bind to specific nucleotide sequences, which are known as *enhancers*. By interaction with basal transcription factors, they stabilize the preinitiation complex and thus increase its binding affinity to the promoter. Repressors work the opposite way, as they are capable of decreasing the promoter affinity of the preinitiation complex by blocking compulsory binding sites. In any case, the key to gene regulation is the fact that transcription factors (TFs) are specific to their binding sites, which are distinct oligonucleotides within the promoter region. The presence of binding sites of a specific TF in a particular promoter indicates a regulating function of that TF with respect to the corresponding gene.

1.1.1. Transcription factor binding sites

The high affinity is caused by biochemical interactions between amino acids in the DNA binding domain of the TF and nucleotides in the major groove of the DNA molecule, as depicted in Figure 1.2(b). Each transcription factor has a unique DNA binding domain with different amino acid combinations being exposed to the DNA. As a result, each transcription factor has a high affinity to a particular sequence. However, this sequence is usually not totally conserved, but there is a certain flexibility, as exchange of a particular nucleotide may slightly reduce binding affinity, without making interaction between TF and DNA impossible.



(a) Transcription initiation complex (b) Binding of a TF to a DNA strand

Figure 1.2.: **Protein-DNA interaction.** Figure 1.2(a) illustrates the formation of the transcription initiation complex, involving interaction of several proteins with DNA and among themselves. Figure 1.2(b) shows the three-dimensional interaction of a single protein in detail. The image was created with BALLview [9] from a 3D structure of USF [10] obtained through the Protein Data Bank in Europe [11].

Sequences with a high affinity to a TF are called *transcription factor binding sites* (TFBS), and have typically a length between 6 and 20 base pairs. The common features of binding sites of a TF are called *sequence motif*.

1.1.2. Experimental detection of protein-DNA interaction

There are several experimental technologies for investigating protein-DNA interaction, i.e., determining DNA-oligonucleotides that have a high binding affinity towards a particular protein. One in vitro approach is based on *systematic evolution of ligands by exponential enrichment* (SELEX), which functions by iteratively mutating a pool of oligonucleotides, which are perceived as potential binding sites, and enriching those candidates with a high binding affinity to the protein of interest [12]. Another in vitro alternative is offered by *protein-binding microarrays* [13, 14], which functions by directly measuring the protein-DNA affinity using hybridization on a microarray that contains all possible oligonucleotides of a certain length. However, this technology is limited to comparatively short sequence motifs, since all possible binding sites need to be present on the microarray.

One of the most popular methods to date for detecting sequence fragments that are associated with a TF in vivo combines chromatin immunoprecipitation (ChIP) with high-throughput DNA sequencing, and is thus called ChIP sequencing, or just ChIP-seq [15, 16, 17]. ChIP can be used to extract DNA that is bound by a particular protein, by (i) cross-linking the protein-DNA complex, (ii) fragmenting DNA (for example by sonication), and (iii) extracting the protein of interest with its bound target DNA using a specific antibody.

After dissolving the protein-DNA complex, the resulting DNA fragments can be identified by high-throughput sequencing [18, 19]. The final tasks of the experimental pipeline are computational: (iv) read mapping, i.e., mapping the sequenced reads to the corresponding genome by tools such as MAQ [20], BWA [21], or Bowtie [22], and (v) peak calling, i.e., identifying genomic regions which are highly enriched with mapped reads, and are thus are called ChIP-peaks, using tools such as MACS [23], F-Seq [24], or PeakSeq [25]. Those genomic regions that are identified by ChIP-seq peaks are then considered to be bound by the protein of interest.

One may assume that those genomic regions contain a common sequence motif, even though in reality this does not necessarily need to be the case. For example some TFs bind to DNA only indirectly by interacting with other DNA-binding proteins. In those cases we would not necessarily expect to find one distinct overrepresented motif in a ChIP-seq data set of that TF.

1.1.3. Splicing and splice sites

In eukaryotic cells, the RNA product of transcription is not used immediately to synthesize protein. Instead, this pre-mRNA is subject to several post-transcriptional modifications resulting in mature mRNA, which is then translated by ribosomes into protein. Besides capping, polyadenylation, and editing, the most important form of modification is called splicing, which was first discovered in 1977 [26].

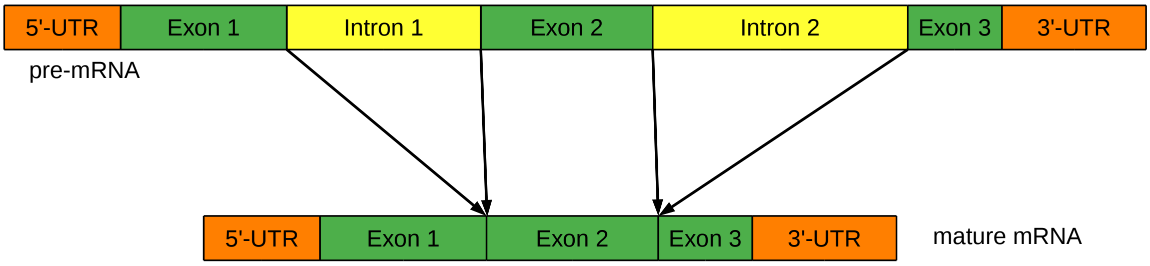


Figure 1.3.: **Simplified illustration of the splicing mechanism.** This example shows a gene consisting of three exons and two introns. Both introns are removed from the pre-mRNA, and the remaining exons are spliced together, forming the mature mRNA. Untranslated regions (UTRs) surround the coding sequence and contain several regulatory sequences.

Figure 1.3 shows a simplified illustration of the process. Parts of the mRNA that do not code for protein, which are called *introns*, are removed, and the remaining sequences, which are called *exons*, are ligated together to form one continuous mRNA molecule.

Some RNA molecules have the ability to catalyze the splicing of themselves, which is called self-splicing. The vast majority of splicing reactions, however, is catalyzed by a multidomain complex, the spliceosome. It consists of five small nuclear ribonucleoprotein particles (snRNPs), which consist of one small nuclear RNA (snRNA) and several proteins, each. Assembled to one complex, they are able to recognize the exon-intron boundaries, split the phosphodiester bonds, and ligate the exons.

Figure 1.4 presents the detailed structure of an intron. The boundary at the 5' end of the intron is called *splice donor site*. It contains a conserved `GU` dinucleotide (`GT` in DNA notation) at the first two positions of the intron [27], which is a key feature for computational recognition of donor sites. The nucleotides upstream and downstream of this dinucleotide also contribute to the signal that is recognized by the spliceosome. Even though they are not conserved, there exists a notable consensus sequence.

The boundary at the 3' end of the intron is called *splice acceptor site*. It contains a highly conserved `AG` dinucleotide at the last two positions of the intron.

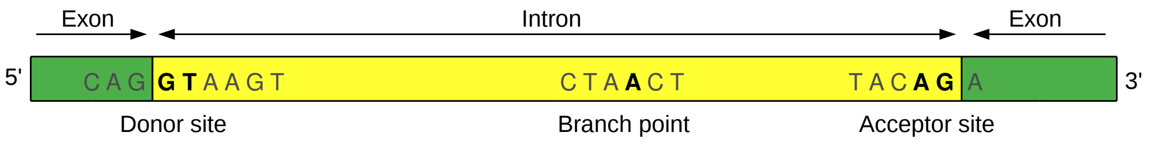


Figure 1.4.: **Schematic illustration of the exon-intron structure.** The splicing mechanism recognizes the nucleotide sequences both at exon-intron boundary (donor site), and at intron-exon boundary (acceptor site).

1.2. Computational challenges

There are several computational challenges that are typically associated with DNA and RNA binding sites.

1.2.1. Binding site recognition

In some cases there is a set of binding sites that are bound by a particular TF given. Sources are either wet-lab experiments, such SELEX or universal arrays (Section 1.1.2), previous computational predictions, or a combination of both. Often collections of binding sites for a TF are aggregated in publicly available or commercial databases, such as Jaspar [28] or TRANSFAC® [29].

Here, the computational challenge are from the task of finding locations of potential additional binding sites in the genome that are sufficiently similar to the already known instances. A common approach is to learn a probability distribution from this training data under a statistical model, in order to utilize the probability of arbitrary sequence being generated from this model as score of that sequence being bound by the TF of interest.

We obtain a somewhat simpler variant of this problem, when not a set of binding sites but only the sequence motif in form of a probability distribution, is available. In both cases, a probability distribution over binding sites of a certain length can be used to classify previously unseen data as binding sites in relation to some background distribution.

1.2.2. De novo motif discovery

In other cases, neither binding sites nor the sequence motif of a TF of interest are known. As a consequence, a representation of putative binding sites has to be inferred from a set of long sequences of possibly different length. This problem is often called *de novo motif discovery*, or short *motif discovery*. Input sequences for motif discovery can be promoter sequences, which are extracted according to similar expression patterns of their corresponding genes. Another source are direct experiments, which yield genomic regions that bind the TF of interest, such as ChIP-seq (Section 1.1.2).

Despite considerable efforts in the field, motif discovery remains a computational problem for which there is no completely satisfying solution yet, even though a plethora of algorithms for this task have been proposed during the last decades. Some of the first algorithms were Gibbs Motif Sampler [30] and MEME [31], the former being based on Gibbs sampling [32], whereas the latter uses the Expectation Maximization algorithm [33].

More sophisticated approaches to solve the motif discovery problem, including enhancements of the aforementioned algorithms [34, 35, 36], focus on different aspects of motif discovery and/or a combination thereof. Some algorithms detect complex cis-regulatory modules [37, 35, 38], others infer the width of the motif [39, 40] or a distribution of positions of binding sites [41, 42, 40] from data. Another class of algorithms utilizes phylogenetic information for recognizing motifs in orthologous sequences of different species [43, 44, 45]. Others focus on a discriminative learning approach [46, 47, 45, 40, 48] in order to predict a motif that is highly specific for a certain data set with respect to a control data set.

1.2.3. Splice site recognition

The recognition of splice sites, i.e., predicting new splice sites based on a set of given training sequences, is very similar to TFBS recognition. However, there are two major differences. First, there are orders of magnitude more splice sites available than TFBS for one particular factor exist, so the training data sets are substantially larger. Second, splice sites are located at exon-intron boundaries, and at least the vast majority of them (canonic splice sites) are centered around the conserved dinucleotide. From this follows that the prediction problem is actually a true classification problem, as there is negative data available: sequences that contain the conserved dinucleotide but are verified to be not located at an exon-intron boundary can be considered as negative data.

Table 1.1.: **Exemplary position weight matrix.** It contains the position-specific nucleotide frequencies of human transcription factor RFX5, obtained from database Jaspar.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
`A`	0.09	0.15	0.00	0.05	0.31	0.00	0.48	0.06	0.00	0.79	0.95	0.00	0.54	0.25	0.22
`C`	0.41	0.21	0.49	0.95	0.56	0.22	0.00	0.00	0.73	0.13	0.05	1.00	0.20	0.09	0.36
`G`	0.14	0.22	0.33	0.00	0.02	0.00	0.52	0.91	0.10	0.08	0.00	0.00	0.00	0.57	0.32
`T`	0.36	0.42	0.18	0.00	0.11	0.78	0.00	0.03	0.17	0.00	0.00	0.00	0.26	0.09	0.10

1.3. Models for sequence patterns

In this section, we discuss established models and model classes for DNA sequence patterns. We consider each position in a functional binding site of width W (splice site, TFBS, etc.) as random variable $X_1, \dots, X_W$. We intend to assign a probability $P(X_1, \dots, X_W)$ to each possible realization of the random variables based on a limited amount of training data. Since the number of possible realizations grows exponentially with W, simplifications w.r.t. the sufficient statistics that are taken into account have to be made. The purpose of a statistical model is to provide an optimal tradeoff between extracting relevant features and model complexity.

1.3.1. Position weight matrix model

The most widespread model for modeling transcription factor binding sites, splice sites, and other functional oligonucleotides is the position weight matrix (PWM) model [49, 50], which assumes statistical independence among all positions in the sequence, which corresponds to the factorization

$$P(X_1, \dots, X_W) = \prod_{\ell=1}^{W} P(X_\ell). \tag{1.1}$$

The sufficient statistics according to the PWM model are the observed mononucleotide frequencies at each position. Both the sufficient statistics and the parameters estimated from them can be perceived as nucleotide weights and can be arranged as a matrix with rows corresponding to nucleotides and columns corresponding to positions in the sequence, and this matrix representation is origin of the term position weight matrix. We show an exemplary PWM for the binding sites of human transcription factor RFX5, extracted from TFBS database Jaspar [28], in Table 1.5. The PWM is also called position-specific scoring matrix (PSSM), which is, albeit being a more accurate description, a less common term in bioinformatics literature.

One advantage of the PWM is the intuitive representation as sequence logo [51], and as such the PWM has become almost equivalent to the sequence motif of a protein as displayed

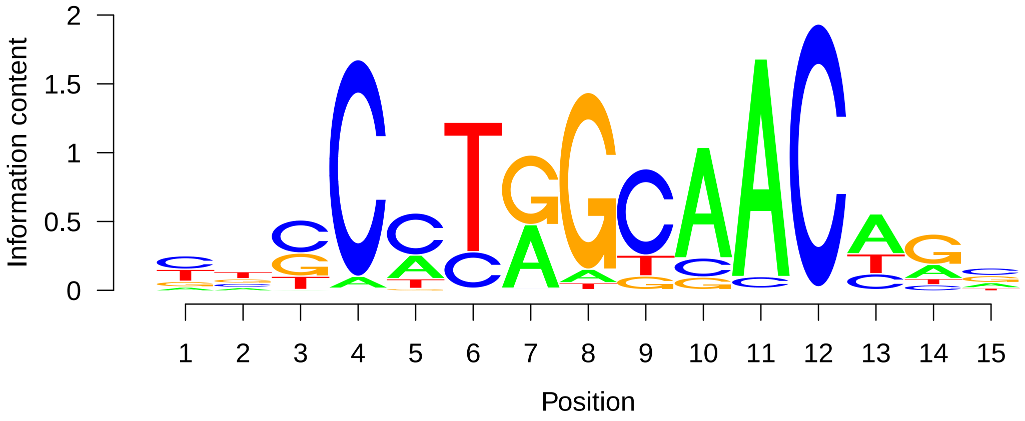


Figure 1.5.: **Exemplary sequence logo.** It visualizes the PWM of transcription factor RFX5 as displayed in Table 1.1.

by databases like TRANSFAC® [29] or Jaspar [28]. In a sequence logo the height of a nucleotide stack at a particular position corresponds to $2 + H(\vec{p})$, which is here called *information content*, with H being the entropy in bits and $\vec{p}$ being the relative nucleotide frequencies at the position. The individual nucleotides are ordered from top to bottom according to their relative frequency and their size is also scaled according to that measure. We show an exemplary sequence logo in Figure 1.5, which corresponds to the PWM of transcription factor RFX5 that is shown in Table 1.1.

However, since the PWM model assumes statistical independence among all positions in the motif, which corresponds to the biophysical assumption of additive binding energies of all nucleotides, it may neglect relevant features in the data. While that independence assumption might be reasonable as a first approximation, there are many indications that a set of independent relative nucleotide frequencies is not sufficient for characterizing a set of binding sites, and studies for different types of transcription factors have shown that the independence assumption of a PWM model is not completely justified [52, 53, 54, 55, 56]. As a consequence, several alternative models that take into account statistical dependencies among nucleotides in the motif have been proposed.

1.3.2. Markov models

The weight array model (WAM) [57] is a simple extension of the PWM model as it may take into account first-order statistical dependencies among adjacent nucleotides in the motif, and thus corresponds to the factorization

$$P(X_1, \ldots, X_W) = P(X_1) \prod_{\ell=2}^{W} P(X_\ell | X_{\ell-1}). \tag{1.2}$$

The likelihood of the WAM is thus a product of conditional probability distributions (CPDs). This model assumes conditional statistical independences of random variable X_ℓ from random variables $X_1, \ldots, X_{\ell-2}$ given random variable $X_{\ell-1}$, which is less restrictive than the unconditional statistical independence assumptions of the PWM model. The WAM can be perceived as a position-specific Markov chain and is thus also called inhomogeneous Markov model (MM) of order one.

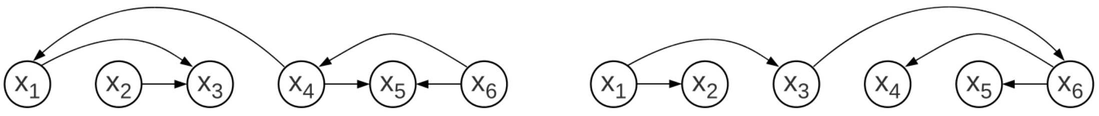
In analogy, inhomogeneous Markov models of order d may take into account statistical dependencies of order d among neighboring nucleotides. They factorize the joint distribution according to

$$\begin{aligned} P(X_1, \ldots, X_W) = & P(X_1)P(X_2|X_1) \cdot \ldots \cdot P(X_d|X_1, \ldots, X_{d-1}) \\ & \cdot \prod_{\ell=d+1}^{W} P(X_\ell|X_{\ell-d}, \ldots, X_{\ell-1}). \end{aligned} \tag{1.3}$$

An inhomogeneous MM thus assumes conditional statistical independences of random variable X_ℓ from random variables $X_1, \ldots, X_{\ell-d-1}$ given random variables $X_{\ell-d}, \ldots, X_{\ell-1}$. The class of inhomogeneous MMs is nested, i.e., the model of order d is included in the model of order $d + 1$. Setting $d = 1$ returns a WAM, and setting $d = 0$ would technically yield a PWM model, which is thus sometimes also called inhomogeneous MM of order 0. While inhomogeneous MMs of order $d > 1$ are more expressive than their special cases PWM model and WAM, they require a larger number of model parameters.

At this stage, some readers may benefit from clarifying terminology. In most communities, the term “Markov model” refers to a model that generates a series of observations of potentially infinite length. Homogeneous Markov models (Markov chains) are the widely used special case where only one transition matrix (CPD) exists, and the term inhomogeneous MM implies only the existence of at least two transition matrices. While the here defined model certainly has the property of multiple transition matrices, it is actually a further special case, where the modeled sequences have fixed length with position-specific transition matrices. However, since the term inhomogeneous MM is established in bioinformatics literature for these models [58], and since we do not need to distinguish to the more general case in this thesis, we stick to this terminology.

While inhomogeneous MMs take into account statistical dependencies of a certain order among adjacent nucleotides, only, permuted Markov models [59] define an inhomogeneous MM over an arbitrary permutation of the random variables. Due to this increased flexibility, permuted Markov models are – at least in theory – capable of modeling long-range dependencies among non-adjacent nucleotides. However, learning the optimal permutation of random variables is an $\mathcal{NP}$-hard problem.



(a) general Bayesian network (b) Bayesian tree

Figure 1.6.: **Bayesian network representation.** We show DAGs for a general Bayesian network and for the special case of a Bayesian tree for sequence patterns of width $W = 6$.

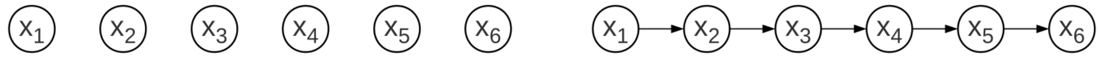
1.3.3. Bayesian networks

All of the previously discussed models are generalized by a highly flexible model class. Bayesian networks (BNs) [60, 61] are capable of modeling any intra-motif dependencies that can be expressed through an directly acyclic graph (DAG) G, when a node corresponds to a particular random variable and the set of parents of this node in the DAG corresponds to the set of random variables that are conditioned upon in the statistical model. In other words, the DAG of a Bayesian network represents the factorization of the joint distribution into an arbitrary product of conditional distributions

$$P(X_1, \ldots, X_W) = \prod_{\ell=1}^{W} P(X_\ell | \operatorname{pa}(X_\ell)), \tag{1.4}$$

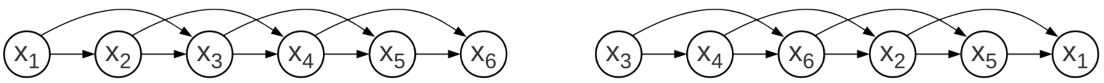
where $\operatorname{pa}(X_\ell)$ returns the parents of X_ℓ in G, for which one example is shown in Figure 1.6(a). As such, BNs are a flexible and powerful model class that has a plethora of applications, which are by no means limited to bioinformatics and computational biology. While the DAG of a BN can be easily visualized and interpreted, there are two important facts to keep in mind. First, the absence of an edge represents a conditional statistical independence assumption, whereas the presence of an edge does not convey information, except of the number of parameters the model uses. For example, even if there is an edge between node A and B, the two corresponding random variables can be independent if all CPDs are equal. Second, even though the DAG contains directed edges, it does not automatically imply causality. An edge from node A to node B does not necessarily imply that A is a cause of B. As a bottom line, the interpretation of the DAG of a BN is limited to being a visualization of conditional independence assumptions of the probabilistic model.

One key challenge in learning Bayesian networks is inferring a DAG from data, a problem that is often called *structure learning.* There are two fundamentally different approaches for learning a structure [62]. The first approach is called constraint-based structure learning and relies on statistical independence tests in order to determine the presence or absence of an edge in the DAG. The second approach is score-based structure learning, which formulates BN structure learning as *model selection* problem. Here, structure learning consists



(a) PWM model

(b) Weight array model



(c) 2nd-order inhomogeneous MM

(d) 2nd-order permuted MM

Figure 1.7.: **DAGs of further special cases of Bayesian networks.**

of (i) assigning a score, which typically assumes the form of a penalized likelihood, to each structure and (ii) performing an exhaustive search through the space of all possible structure in order to find the structure with the maximal score. Since the number of DAGs grows superexponentially with the number of nodes, which corresponds to the number of random variables, finding the optimal BN structure w.r.t. a given score is an $\mathcal{NP}$-hard problem [63].

Some algorithms attempt to solve the structure learning problem locally using greedy strategies [64, 65], but there is are no guarantees about the global quality of the found solution. There are several algorithms for finding the globally optimal Bayesian networks that are based on dynamic programming [66, 67, 68, 69], but they are limited to a relatively small number of variables.

Structure learning becomes simpler when not the total space of all possible BN structures is considered, but the optimum within a subspace, defining a restricted model class, is to be found. One prominent example are Bayesian trees, which are Bayesian networks were the number of parent nodes is set to one for each node with the except of one parent-less node, which is the root of the tree. One example for a Bayesian tree is shown in Figure 1.6(b). Here, structure learning can be achieved in polynomial time, using the Chow-Liu algorithm [70]. Bayesian trees can be also perceived as a generalization of the WAM, capable of modeling statistical dependencies among non-adjacent positions without increasing the number of model parameters at the cost of neglecting some dependencies among adjacent positions.

In addition, all of the previously mentioned models can be perceived as special cases of a general Bayesian network where each random variable corresponds to one single motif position, i.e., each model can be expressed using a particular, fixed, DAG. The PWM model can be perceived as an empty Bayesian network (Figure 1.7(a); the DAG has no edges). Outside bioinformatics and computational biology, this model is also often called *independence model*, as there is full statistical independence among all random variables. Likewise, the representation of the WAM is a DAG where the sole parent of each node is its direct predecessor as illustrated in Figure 1.7(b). Inhomogeneous Markov models and their generalization that allows permutation of the random variables are also both special cases of BNs as shown in Figure 1.7(c) and Figure 1.7(b). From this follows that both PWM model and WAM can be considered as inhomogeneous Markov models, since they can be obtained by enforcing additional conditional independences, i.e., by removing edges from the DAG.

Inspired by inhomogeneous Markov models and permuted Markov models of order d, it is also possible to define Bayesian networks of order d, which contain all DAGs where $W - d$ variables have exactly d parents, and the remaining d variables have $d-1, \dots, 0$ parents. This restriction has the consequence that for a given value of d, all BN(d) have the same number of parameters, which can be a desirable feature. Furthermore, the restriction to a fixed number of parent nodes simplifies structure learning.

1.4. Parsimonious context trees

All models discussed in Section 1.3 share a common problem: The number of parameters for modeling the distribution of one random variable grows exponentially with the number of parent variables that are conditioned upon. For inhomogeneous Markov models, permuted Markov models, and Bayesian networks of order d, the number of parameters of the full model grows exponentially with d. General Bayesian networks, which are capable of making use of fully flexible parent node selection, can adapt the complexity for each random variable by selecting a different number of parents, yet the total number of parameters is dominated by the variable with the highest number of parents.

A high number of parameters in relation to the number of data points that these parameters are to be estimated from is often problematic. Whereas a complex model always achieves a higher likelihood on the training data than a simpler special case, this does not automatically hold for independent test data. Overly complex models learned on insufficient data often generalize poorly, as they adapt to random noise in the training data. This effect is known as *overfitting* does not only apply to the models mentioned in Section 1.3, but is a well-known problem in statistical learning.

As a closely related problem, the number of parameters of homogeneous Markov models grows exponentially with model order. In order to cope with this problem, parsimonious Markov models have been proposed [71, 72] as extensions of homogeneous Markov models. Parsimonious Markov models replace the transition matrix of the Markov chain by a reduced CPD table that groups several observations in the condition. This grouping of possible observations is organized by a data structure that is called *parsimonious context tree* (PCT). Apart from their use in parsimonious Markov models, PCTs have already been applied to extend Hidden Markov Models to Parsimonious Hidden Markov Models [73].

1.4.1. Definition

A PCT τ of depth d for alphabet $\mathcal{A}$ is a rooted, balanced tree. Each node of a PCT is labeled by a non-empty subset of $\mathcal{A}$, except for the root, which is labeled by the empty subset. The set of labels of all children of an arbitrary inner node forms a partition of $\mathcal{A}$.

It follows that the cross product of the symbol sets encountered along each path from a leaf to the root defines a non-empty subset of $\mathcal{A}^d$, which we call *context*. Hence, a context is a set of context words, and the set of the contexts of all leaves of a PCT forms a partition of $\mathcal{A}^d$. Thus, the PCT is a data structure that represents a partition of the whole set of context words.

For example, the PCT of depth two for the four-letter DNA alphabet $\mathcal{A} = \{\mathtt{A,C,G,T}\}$ that is shown in Figure 1.8 encodes the contexts $\{\mathtt{A}\} \times \{\mathtt{A}\}$, $\{\mathtt{C,G}\} \times \{\mathtt{A}\}$, $\{\mathtt{T}\} \times \{\mathtt{A}\}$, $\{\mathtt{A,G}\} \times \{\mathtt{C,G,T}\}$, and $\{\mathtt{C,T}\} \times \{\mathtt{C,G,T}\}$. The context sequences are read in the same direction as they appear in the data, so traversing the PCT top-down corresponds to looking into the past with distant nodes in the tree corresponding to distant symbols in the sequence. The first layer of nodes, which directly succeed the root, thus corresponds to the direct predecessor of the symbol of interest, i.e., the rightmost symbol in the context word.

A PCT of depth d interpolates between two extreme cases: a *minimal* tree with only one leaf, which represents the union of all context words into one set, and a *maximal* tree with $|\mathcal{A}^d|$ leaves, each of which represents a single context word.

1.4.2. Finding optimal PCTs

Finding the optimal out of an overexponential number of possible PCTs (with respect to model order and alphabet size) without computing the score for every single PCT explicitly is challenging. This problem can be solved by a dynamic programming (DP) algorithm similar to the context tree maximization algorithm [74]. The algorithm runs on a data structure that we call the *extended PCT* of depth d and that we denote by $\mathcal{T}_d^{\mathcal{A}}$. In contrast to a PCT, the children of a node of an extended PCT do not form a partition of alphabet $\mathcal{A}$, but

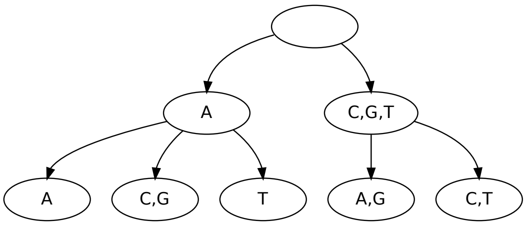


Figure 1.8.: **Example PCT of depth 2 over DNA alphabet.** It encodes the partition of all 16 possible context words into subsets {AA},{CA,GA},{TA},{AC,AG,AT,GC,GG,GT}, and {CC,CG,CT,TC,TG,TT}.

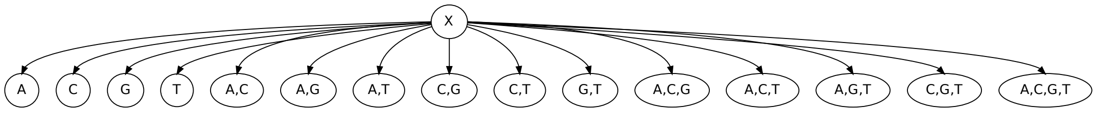


Figure 1.9.: **Structure of an inner node and its children in extended PCT.** We show an arbitrary inner node (labeled by x) and its children in the extended PCT over DNA alphabet. The labels of all children form $\mathcal{P}_{\geq 1}(\mathcal{A})$.

rather encompass all elements of $\mathcal{P}_{\geq 1}(\mathcal{A})$, which is the power set of $\mathcal{A}$ minus the empty set (Figure 1.9). The leaves of an extended PCT are thus all possible leaves, identified by their label concatenation up to the root, that may occur in any PCT of same depth and alphabet.

Let $\mathcal{N}(\mathcal{T})$ denote the set of nodes of an extended PCT $\mathcal{T}$, n one element of $\mathcal{N}(\mathcal{T})$, and $r(\mathcal{T})$ the root of $\mathcal{T}$. Each node can be uniquely identified by the label concatenation on the path from this node up to the root of the extended PCT. Let $s(n)$ denote the score of the optimal PCT subtree rooted at n. Let $\mathcal{C}(n)$ denote the set of all children of n in the extended PCT. Let $\mathcal{V}(\mathcal{C}(n))$ denote the set of all valid child combinations, i.e., all subsets of children whose labels form a partition of $\mathcal{A}$. Let further $\mathcal{L}(\mathcal{T})$ denote the leaves and $\mathcal{I}(\mathcal{T})$ the remaining inner nodes of $\mathcal{N}(\mathcal{T})$. Using this notation, we specify the dynamic programming approach in Algorithm 1, which consists of a single function for computing the optimal PCT subtree rooted at an arbitrary node of the extended PCT.

Algorithm 1 Dynamic programming for finding optimal PCT subtrees

```
findOptimalSubtree($n$)
  if $n \in \mathcal{L}(\mathcal{T}_d^{\mathcal{A}})$ then
    compute $s(n)$
  end if
  if $n \in \mathcal{I}(\mathcal{T}_d^{\mathcal{A}})$ then
    for all $m \in \mathcal{C}(n)$ do
      findOptimalSubtree($m$)
    end for
    for all $v \in \mathcal{V}(\mathcal{C}(n))$ do
      $s(v) := \sum_{m \in v} s(m)$
    end for
    $v^* := \operatorname{argmax}_{v \in \mathcal{V}(\mathcal{C}(n))} s(v)$
    $s(n) := s(v^*)$
    for all $m \in \mathcal{C}(n) \setminus v^*$ do
      remove $m$ and subtree below
    end for
  end if
```

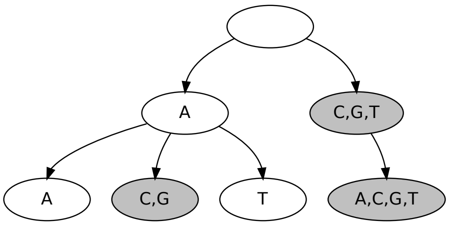


Figure 1.10.: **Example CT of depth 2 over DNA alphabet.** Pruned contexts are here shown as pseudo-nodes (displayed in gray) in order to achieve depth two for all possible contexts and thus allow a visualization of the CT in PCT-style.

Applying this function to the root of the extended PCT, that is, calling the function `findOptimalSubtree`$(r(\mathcal{T}_d^{\mathcal{A}}))$, yields the optimal PCT.

The algorithm can be intuitively described as bottom-up reduction of the extended PCT towards a valid PCT by selecting at each inner node the locally optimal PCT subtree. We provide a step-by-step explanation of the mode of operation of the algorithm using the smallest meaningful example, which comprises the task of finding an optimal PCT of depth $d = 2$ over a ternary alphabet, in Appendix A.

The correctness of the algorithm follows from the property that the score of a PCT is a sum of leaf scores, which further implies that the score of a PCT subtree rooted at node n depends, apart from its own structure, only on the nodes on the path from n up to the root, but is independent of the structure of the PCT subtrees rooted at siblings of n.

The complexity of the DP algorithm is given by the size of the extended PCT, which must be completely traversed, multiplied by the number of valid child combinations, for which a score must be computed in each inner node of the extended PCT. Whereas the former is exponential with the base being the number of possible subsets of $\mathcal{A}$, the latter is equivalent to the Bell number $B_{|\mathcal{A}|}$.

1.4.3. Special cases

Context trees (CTs), which are used by variable order Markov models (VOMMs) [75], are special cases of PCTs. In sequence analysis, CTs have been used in permuted variable length Markov Chains (VLMC) [76], variable order Bayesian networks (VOBN) [77], and inhomogeneous VOMMs, which are a special cases of both permuted VLMC and VOBN.

The differences between CTs and PCTs arise from a different concept of tree-building. Whereas the idea of building CTs is to prune a maximal tree by removing unimportant subtrees, the idea of PCTs is to fuse nodes if subtrees and corresponding CPDs are not sufficiently different. Since removing nodes can be also expressed by fusing them into one pseudo-node [78], CTs are special cases of PCTs (Figure 1.10).

The opposite does not hold, though. There are many PCTs that represent a set of contexts that cannot be represented by CTs, since the notion of pruning yields several limitations of the possible CT structures that are relaxed by PCTs. Two structural features distinguish PCTs from CTs. First, an inner node in a PCT may have an arbitrary number of fused children as long as their labels form a partition of $\mathcal{A}$, whereas a CT allows at most one fused child, the pseudo-node. Second, a PCT allows arbitrary subtrees below a fused node, whereas a CT allows only a completely fused node as single child of a fused parent, which is equivalent to removing the entire subtree below the first occurrence of a fused node.

1.5. Objectives and outline of the thesis

In this section, we provide an overview of the different topics covered in this thesis. In Section 1.5.1 we discuss the two main objectives, that is, the main scientific questions addressed in the course of this work. We provide a detailed outline of the content of the different chapters and the logical connections among them in Section 1.5.2.

1.5.1. Main objectives

In this thesis, we study several closely related problems, which may yet be attributed to one out of two main objectives. Both main objectives can be formulated as an attempt to answer a very general key question each.

Main objective I is computational, contributing to the fields of machine learning and probabilistic modeling. Here the key question is:

> Can parsimonious context trees by applied to the problem of modeling DNA binding site data and what are appropriate learning principles for that task?

We intend to utilize parsimonious context trees to design a statistical model for the task of modeling dependencies among adjacent entries in a categorical data matrix, i.e., DNA binding sites. The incentive is here to obtain a fine-grained model class that allows a more flexible adaption of the parameter space to the problem at hand than offered by fixed order Markov models and variable order Markov models. Such a fine-grained model class, however, renders the option of manually picking one particular model based on expert knowledge impossible. Hence, we also desire adequate learning algorithms to infer a model of appropriate complexity from data – no matter whether this data is fully observable or whether latent variables are involved, as it is the case for de novo motif discovery. Especially the latter problem, dealing with variable model complexity in the presence of latent variables, is a widely unexplored field in machine learning yet, and we thus aim at a solution that may be generally applicable, and is widely independent of the precise model class involved.

Main objective II is more biological, namely contributing to understanding of protein-DNA interaction. Here the key questions is:

> What is the quantitative and qualitative nature of intra-motif dependencies for transcription factor binding sites?

Whereas there are indications that dependencies do exist for the binding motifs of some selected proteins, it is unknown to date how prevalent they actually are in nature. We thus intend to generate evidence for the presence or absence of dependencies of a certain order within a large number of data sets. In addition, we intend to study the precise nature of putative dependencies, quantitatively by comparing dependencies within a motif in a position-specific manner, and qualitatively by investigating the difference between different features, i.e., different conditional probability distributions, for the same position. In order to achieve that goal and in order to make more complex sequence motifs easier to appreciate for the experimental scientist, we propose a visualization method that is comparable to the popular sequence logo representation.

The computational and biological objectives are by no means independent of each other. On the contrary, both main objectives are heavily intertwined, and in some sense their connection resembles a chicken-and-egg problem. Without substantial evidence for intra-motif dependencies, studying complex statistical approaches for modeling them can hardly be motivated. However, with present statistical models and learning methods, which are prone to overfitting when attempting to take into intra-motif dependencies, it may be impossible to collect such evidence in the first place. This is the reason for investigating both aspects simultaneously, and this thesis is thus an attempt to resolve this chicken-and-egg problem.

Main objective I is primarily dealt with in Chapters 2, 3, 6, and 7, whereas main objective II is the focus of Chapters 5 and 8. In the following section, we sketch the distribution of the topic over the different chapters, the logical connection among them, and their contribution to the two main objectives in more detail.

1.5.2. Outline

Here, we briefly sketch the outline of the thesis and the motivation of the different chapters. The distribution of the topics over the different chapters as well as the logical connection among the chapters and the main objectives they contribute to is shown in Figure 1.11.

In Chapter 2, we introduce inhomogeneous parsimonious Markov models (PMMs) as the model class of interest in order fulfill the first subtask of main objective I. We specify a prior distribution and propose a Bayesian model selection approach for learning a model from fully observable data, i.e., a set of aligned sequences of equal length. Using a small real-world data set, we demonstrate that the model class is capable of a superior prediction performance in relation to alternative approaches using traditional context trees. While the

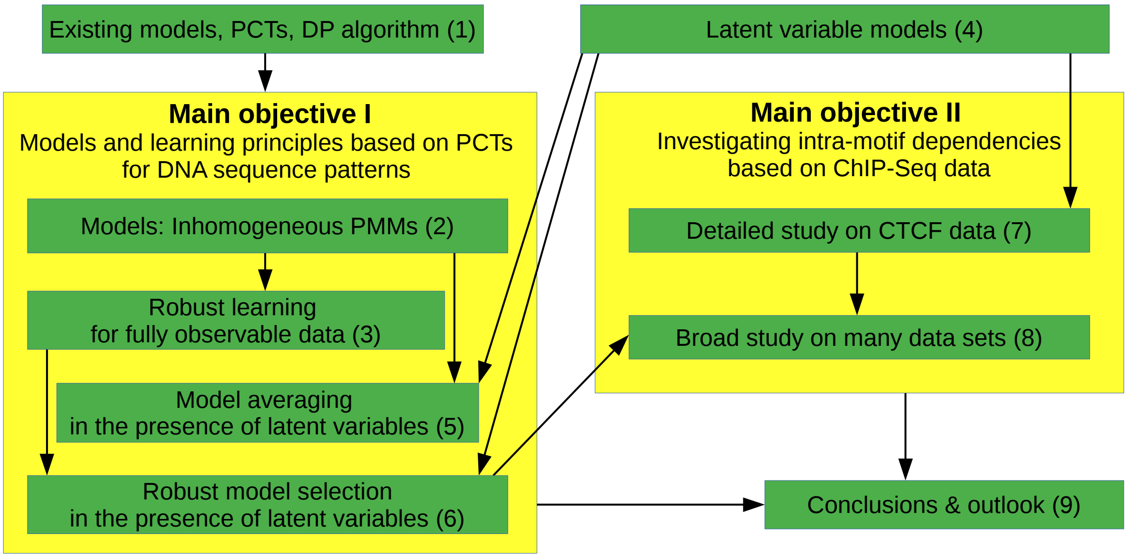

Existing models, PCTs, DP algorithm (1)
Latent variable models (4)
Main objective I
Models and learning principles based on PCTs for DNA sequence patterns
Models: Inhomogeneous PMMs (2)
Robust learning for fully observable data (3)
Model averaging in the presence of latent variables (5)
Robust model selection in the presence of latent variables (6)
Main objective II
Investigating intra-motif dependencies based on ChIP-Seq data
Detailed study on CTCF data (7)
Broad study on many data sets (8)
Conclusions & outlook (9)

Figure 1.11.: **Flowchart of the thesis content**. Arrows denote main logical connections among different topics. Numbers in brackets correspond to the chapter a particular topic is dealt with in.

results indicate that PMMs are worth studying further, we have to resort to cross-validation for hyperparameter-tuning of the structure prior in order to obtain them. Whereas the same applies to models with traditional context trees, such a brute-force approach is not satisfying, since it is not particularly aesthetic and requires a substantial amount of computational effort.

We thus study alternative learning approaches for fully observable data in Chapter 3, further contributing to main objective I. Several of these structures scores for model selection and methods for parameter estimation were originally proposed for Bayesian networks and are mainly, but not solely, motivated by the minimum description length principle. By empirical comparison on splice donor sites and TFBS data, we find that BIC as structure score combined with fsNML for estimating the probability parameters is the most appropriate method for learning PMMs, yielding robust results without requiring any hyperparameter-tuning.

Whereas Chapter 2 and Chapter 3 are concerned with the comparable simple case of fully observable data, i.e., learning from a set of aligned sequences of the same length, the true challenge is given by the presence of latent variables. Chapter 4 thus serves as kind of a second introduction, where we recapitulate different latent variable models from literature, namely mixture models and models for motif discovery. We discuss some previous attempts to learn models with variable structure in a setting where latent variables occur, such as a mixture of Bayesian trees. We find that the only previously existing method that deals with learning possibly multiple models with variable dimensionality of the parameter space is based on approximating the posterior maximum of the entire latent variable model using an EM algorithm. However, this approach has the theoretical problem of not yielding a consistent estimate of the model structures, but asymptotically picks the largest possible candidate. Hence massive hyperparameter-tuning is here inevitable for obtaining any models of reasonable complexity.

Despite the theoretical weaknesses, resulting in a high computational effort for practical application, we apply this EM algorithm in Chapter 5 in order to address some of the biological questions that are given by the main objective II. We perform a first motif discovery using PMMs for ChIP-seq data sets of the of human insulator protein CTCF. We find that the CTCF motif is indeed more complex than previously thought, and that a PMM can utilize the additional features that are neglected by the PWM model to improve motif discovery substantially. Moreover, we propose in Chapter 5 a visualization method of those features that we dub conditional sequence logo, as it is inspired by the traditional sequence logo representation of a PWM. Since we obtain this promising results only using massive cross-validation for hyperparameter-tuning, it is infeasible to apply exactly this method to a large number of data sets, which is the incentive for studying alternative possibilities of dealing with variable structures with a different number of parameters in the presence of latent variables.

As a first idea, we refrain from seeking an optimal at all, but intend to perform Bayesian model averaging over the space of all PMMs. Further contributing to main objective I, we derive in Chapter 6 a Gibbs sampling algorithm that generates structure and parameter samples from the posterior, which can be subsequently used for approximating the posterior for a fully Bayesian prediction. The challenge is here to sample a particular PCT structure from its conditional distribution given data and latent variables, which can be achieved by a modification of the PCT maximization algorithm. We find for both latent variable models of interest that the Gibbs sampler quickly converges and that model averaging might a robust alternative to the EM algorithm, as Bayesian model averaging behaves similar to Bayesian model selection with respect to prior influence. However, whereas the Gibbs sampling algorithm itself is fast, the entire model averaging approach has the drawback that the Bayesian prediction becomes infeasible for many practical applications as the running time scales linearly in the number of sampled parameter sets.

This observation once again increases the desire to conduct model selection in the presence of latent variables, which is the topic of Chapter 7. Inspired by the Gibbs sampler, and also motivated by the lessons learned in Chapter 3, we propose a stochastic algorithm that performs model selection for arbitrary latent variable models according to arbitrary learning principles. We empirically show both for mixtures of PMMs and for motif discovery with PMMs that the method converges quickly and yields a robust and fast prediction and classification, which finally fulfills main objective I of this thesis, answering the raised question positively.

Using this algorithm, we are capable of addressing main objective II on a larger scale than it was possible in Chapter 5, where we focused on one single, albeit were important, DNA-binding protein. In Chapter 8, we perform a motif discovery within various ChIP-seq data sets from the ENCODE project and investigate the prevalence of intra-motif dependencies. Here, we find that first-order dependencies exist in sequence motifs of almost all relevant transcription factors, with several examples showing further higher-order dependencies.

We summarize the results of this thesis and further discuss the main conclusions in Chapter 9. In addition, we also provide a glimpse to potential future research topics that arise from the lessons learned in the course of this work.

Dealing with complexity is an inefficient and unnecessary waste of time, attention and mental energy. There is never any justification for things being complex when they could be simple.

Edward de Bono

2

Inhomogeneous PMMs

It is indeed a philosophical question whether a model as an approximation to a real world process should be as simple or as complex as possible. A common opinion states that true beauty and elegance is to be found in simplicity of an explanation for a process that just seemed to be utterly complex at first glance. The field of theoretical physics is full of instances where these desires are satisfied, with Einstein's $E = mc^2$ being probably the most famous example.

However, there is no clear justification why nature should favor simplicity, and the field of biology appears to be rather governed by diversity, complexity, and inaccuracy. It is because of the apparent difficulty of establishing simple and elegant theories that research in modern molecular biology is mainly conducted by (i) gathering a vast amount of empirical evidence through wet-lab experiments, and (ii) analyzing the collected data by statistical and computational methods.

2.1. Motivation

We discussed in Section 1.3 that various statistical models of different complexity for inferring sequence motifs from TFBS data exist. The PWM model is obviously the most simple model among the given alternatives, but comparing other presented alternatives is not trivial: is a BN represented by a DAG in Figure 1.6(a) more complex than the second-order inhomogeneous MM in Figure 1.7(c)? On the one hand, one may argue that the dependency structure of the latter is more regular in relation to the general BN under consideration, which is thus more complex. On the other hand, the second-order inhomogeneous MM has more free parameters than the shown BN, allowing a higher flexibility and thus complexity of the learned probability distribution.

In this thesis, we follow the second notion and associate the *model complexity* with the number of free parameters of the model. Hence, a sparse Bayesian network with a seemingly irregular DAG, possibly inferred by a complicated learning algorithm, is considered less complex than a given, regular dependency structure with many CPDs. From that point of view, we indeed strive for eliminating complexity in the sense that we intend to learn statistical dependencies from the data with as simple models as possible.

Unfortunately, existing models as described in Section 1.3 are not very suitable for that task, as they do not allow a fine-grained selection of optimal model complexity. For inhomogeneous MMs, the only possible choice is the model order, which allows selecting among different model complexities on an exponential scale. BNs allow variable-specific selection of parent nodes, but for each individual variable, the possible model complexities scale exponentially as well.

Variable order Markov models [76] and variable order Bayesian networks [77] are attempts of performing a more fine-grained adaption of the model complexity to data at hand, but traditional context trees are limited in many aspects as discussed in Section 1.4. Here, we propose a more general model class that consists of an inhomogeneous Markov model that may use parsimonious context trees for reducing the parameter space at each position. The incentive is to design, in the spirit of the opening quote of this chapter, a model class and corresponding learning principle that allows taking into account statistical dependencies of possibly higher order with simple models that make use of few parameters and are thus less prone to overfitting than their more complex alternatives.

2.2. Model specification

In this section, we formally define inhomogeneous parsimonious Markov models, and propose an appropriate prior distribution. We discuss a possible learning approach for PMMs that is based on Bayesian model selection.

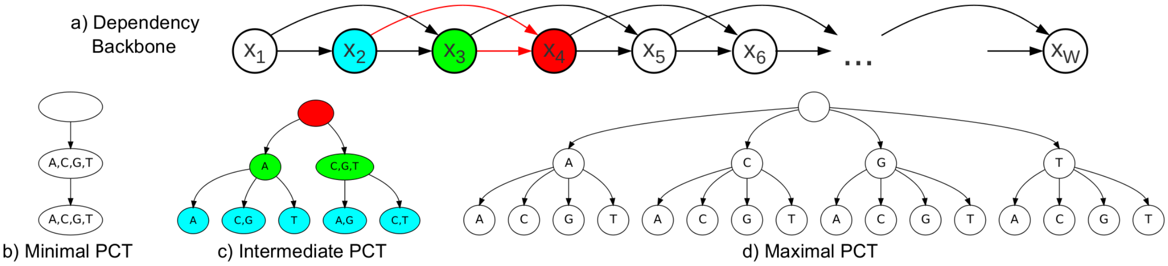


Figure 2.1.: **Second-order inhomogeneous PMM.** Figure a) shows the general dependency structure among the random variables, corresponding to the positions in the motif as DAG in a BN representation. Figure b) shows a minimal PCT of depth 2, which is locally equivalent to a PWM model, since all contexts are merged. Figure c) shows an intermediate PCT of depth 2. If we assume that this tree is used at position 4 in the motif, the nodes are colored according to the random variables they correspond to in the backbone of Figure a). Figure d) shows a maximal PCT of depth 2, which is locally equivalent to a second-order inhomogeneous MM, since none of the contexts are merged.

2.2.1. Likelihood

An inhomogeneous PMM of order d for sequence patterns of width W is based on exactly W PCTs (Figure 2.1), which we denote by $\vec{\tau} = (\tau_1, \ldots, \tau_W)$. For the ease of presentation, we exclude the first d PCTs, which have an increasing depth of $0, \ldots, d-1$, from the following discussion. Since there is a bijective mapping from the leaves of a PCT to the corresponding contexts, we can perceive a PCT as a set of contexts as well as a set of leaf nodes. Hence, we denote a single context by c, the number of context words represented by a context by $|c|$, and the set of all contexts in a PCT by τ itself.

We denote the conditional probability of observing a symbol $a \in \mathcal{A}$ given that the concatenation of the preceding d symbols is in c by θ_{ca}. We denote the model parameters of a single position by $\left(\tau, (\vec{\theta}_c)_{c \in \tau}\right)$ and all model parameters by

$$\Theta = \left(\tau_\ell, (\vec{\theta}_{\ell c})_{c \in \tau_\ell}\right)_{\ell \in (1, \ldots, W)}$$

We now define the likelihood of an inhomogeneous PMM for a data set $\mathbf{x}$ consisting of N sequences of width W by

$$P(\mathbf{x}|\Theta) = \prod_{\ell=1}^{W} \prod_{c \in \tau_\ell} \prod_{a \in \mathcal{A}} (\theta_{\ell ca})^{N_{\ell ca}}, \tag{2.1}$$

where $N_{\ell ca}$ is the number of occurrences of symbol a at position ℓ in all sequences of $\mathbf{x}$ where the concatenation of the symbols from position $\ell - d$ to position $\ell - 1$ is in c.

After having properly defined an inhomogeneous PMM, we drop the explicit reference to the inhomogeneity for the rest of this thesis for the sake of convenience and generally assume inhomogeneity of all Markovian models in discussion including fixed-order Markov models and VOMMs. In situations where we need to refer to the homogeneous variants, we do so explicitly.

2.2.2. Prior

We attempt to learn the model using Bayesian framework, so we need a prior distribution even if we do not intend to actually incorporate a priori knowledge. From a truly Bayesian point of view, a model is essentially defined only by the combination of both likelihood and prior, but such an extremely subjective viewpoint makes the definition of a prior difficult. In fact, the most important property of a prior is not that it reflects our prior belief, which may be rather vague anyway, but that all required derivations can be carried out efficiently and yield closed-form results. We thus choose a conjugate prior in order to analytically compute the posterior distributions and we assume – inspired by Bayesian networks – local and global parameter independence [60]. We thus define the prior of a PMM by

$$P(\Theta) = P(\vec{\tau}) \prod_{\ell=1}^{W} \prod_{c \in \tau_\ell} P(\vec{\theta}_{\ell c}), \tag{2.2}$$

where $P(\vec{\tau})$ is the prior probability of all PCTs $\vec{\tau}$, and could thus be referred to as structure prior, and $P(\vec{\theta}_{\ell c})$ is the prior over the conditional probability parameters of one particular context c at position ℓ. We specify the structure prior by

$$P(\vec{\tau}) \propto \prod_{\ell=1}^{W} \kappa^{|\tau_\ell|}, \tag{2.3}$$

where $|\tau_\ell|$ denotes the number of leaves of τ_ℓ. It depends on one scalar hyperparameter $\kappa \in (0, \infty)$, which can be used to influence the number of leaves and thus the complexity of the model, interpolating between the two extreme cases: When $\kappa \to +\infty$, the model that has maximal PCTs at all positions, and is thus equivalent to a fixed order Markov model, receives a prior probability of one. Conversely, when $\kappa \to 0$, the model that has minimal PCTs at all positions, and is thus equivalent to an independence model, receives full prior support. For the local parameter priors $P(\vec{\theta}_{\ell c})$ we choose Dirichlet distributions with hyperparameters $\vec{\alpha}_{\ell c}$, that is,

$$P(\vec{\theta}_{\ell c}) = \frac{1}{\mathcal{B}(\vec{\alpha}_{\ell c})} \prod_{a \in \mathcal{A}} (\theta_{\ell c a})^{\alpha_{\ell c a} - 1}, \tag{2.4}$$

where $\mathcal{B}$ denotes the multinomial beta-function, which is defined by

$$\mathcal{B}(\vec{n}) = \frac{\prod_{i=1}^{d} \Gamma(n_i)}{\Gamma\left(\sum_{i=1}^{d} n_i\right)}, \tag{2.5}$$

where Γ is the gamma function, which is defined by $\Gamma(n) = \int_0^\infty t^{n-1} e^{-t} dt$ for real-valued n, and $\Gamma(n) = (n-1)!$ for the special case of n being an integer.

In this work, we further restrict the parameter priors to symmetric Dirichlet distributions. Following the equivalent sample size (ESS) concept [60], we obtain a natural computation of the pseudocounts from the equivalent sample size η that is inspired by Bayesian networks, namely $\alpha_{\ell c a} = \frac{\eta |c|}{|\mathcal{A}|^{d+1}}$.

2.2.3. Learning

Learning PMMs consists, comparable to many other probabilistic graphical models, of structure and parameter learning, with the former being the more challenging task.

In order to learn the structure of the model, we intend to find the parsimonious context trees that maximize $P(\vec{\tau}|\mathbf{x})$. Since $P(\mathbf{x})$ is constant w.r.t. the tree structures, it is sufficient to maximize

$$P(\vec{\tau}, \mathbf{x}) = \int P(\mathbf{x}|\Theta)P(\Theta)d\Theta_{\vec{\tau}}, \tag{2.6}$$

where $\Theta_{\vec{\tau}}$ denotes here the conditional probability parameters within Θ for given PCT structures $\vec{\tau}$. Solving the remaining integral (Appendix Section B.1), we obtain the optimization problem

$$\forall_{\ell=1}^{W} : \hat{\tau}_\ell := \operatorname*{argmax}_{\tau_\ell} \prod_{c \in \tau_\ell} \kappa \, \frac{\mathcal{B}(\vec{N}_{\ell c} + \vec{\alpha}_{\ell c})}{\mathcal{B}(\vec{\alpha}_{\ell c})}. \tag{2.7}$$

The target function for PCT optimization is thus a product over local marginal likelihoods for all contexts multiplied by the structure prior constants for each context, and the full structure learning problem reduces to optimizing each PCT in the model individually.

Having determined all optimal PCTs, we estimate their conditional probability parameters according to the posterior mean [79]. It is in general defined by $\hat{\theta} = \int_\theta \theta P(\theta|\mathbf{x})d\theta$ and yields for PMMs

$$\forall_{\ell=1}^{W} \forall_{c \in \tau_\ell} \forall_{a \in \mathcal{A}} : \hat{\theta}_{\ell ca} := \frac{N_{\ell ca} + \alpha_{\ell ca}}{N_{\ell c \cdot} + \alpha_{\ell c \cdot}}. \tag{2.8}$$

A common task is prediction, i.e., computing the probability of a data point $\vec{x}_{N+1}$ after having observed N data points $(\vec{x}_1, \ldots, \vec{x}_N)$. In the Bayesian setting, this is done by integrating over the space of parameters, which is in resonance with structure learning, where the target function is the probability of the model structure given data, integrated over all parameters. Here, we obtain for PMMs

$$P(\vec{x}_{N+1}|\mathbf{x}, \vec{\tau}) = \int P(\vec{x}_{N+1}|\Theta) \prod_{\ell=1}^{W} P(\vec{\theta}^{\tau_\ell}|\mathbf{x})d\Theta_{\vec{\tau}},$$

which is equivalent to computing $P(\vec{x}_{N+1}|\vec{\tau}, \hat{\Theta}_{\vec{\tau}})$, where $\hat{\Theta}_{\vec{\tau}}$ is the posterior mean of the parameters (Equation 2.8) of all PCTs in the model.

2.3. Performance evaluation

In a first study, we apply PMMs to the prediction of DNA binding sites of the eukaryotic transcription factor C/EBP [80] and compare it with VOMMs. The C/EBP data set consists of $N = 96$ DNA binding sites from human and mouse, retrieved from the TRANSFAC® database [29]. These binding sites are aligned sequences of fixed length $W = 12$ over the DNA alphabet $\mathcal{A} = \{\texttt{A},\texttt{C},\texttt{G},\texttt{T}\}$.

2.3.1. Comparing prediction performance

The Bayesian learning approach for PMMs and VOMMs described above allows influencing the complexity of the model via the structure prior (Equation 2.3). Since it is not immediately clear which value of hyperparameter κ translates to which model complexity, specifying the structure prior manually is not a trivial task. While a uniform prior over all structures, which we obtain by setting $\kappa = 1$, may appear as a reasonable option in the absence of a priori knowledge, it might yield a model structure that is not optimal for prediction and related tasks.

Hence, in a first study we investigate the performance of third-order PMMs and VOMMs for different model complexities. Even though statistical models are often used for classification purposes (e.g. positive vs negative sites), we here focus on prediction as it is the main subtask within many classification approaches but does not require the choice of a negative data set and a corresponding statistical model, which both may influence results heavily.

Since the C/EBP data set is rather small, we perform a leave-one-out cross-validation (CV). We remove the i-th sequence from the data set, learn a model on the remaining 95 sequences using $\eta = 1$ for the parameter prior, and compute the predictive probability of the i-th sequence. We repeat this procedure for $i = 1, \ldots, 96$, compute the average number of leaves of the models, and compute the arithmetic mean of the 96 logarithmized predictive probabilities, as well as the corresponding standard errors.

In Figure 2.2 we plot, for both model classes, mean log predictive probability against average model complexity, quantified by the number of leaves of all context trees in the model, which is proportional to the total number of parameters. We choose values of κ that cover the whole range of model complexity, interpolating from the minimal model with only 12 leaves, which is equivalent to the independence model, to the maximal model with 597 leaves, which is equivalent to the third-order Markov model.

We observe that for low model complexities (less than 50 leaves) PMMs yield a substantially higher prediction than VOMMs. For high model complexities both approaches show a similar prediction, which is, however, lower than the prediction achieved by a simple independence model, which indicates that overfitting occurs. These results are interesting in two aspects. First, PMMs are capable of utilizing statistical dependencies in the data for improving prediction if the structure prior is chosen well. A uniform structure prior corresponds here to a model of approximately 110 leaves, which confirms that using a uniform prior is not an optimal choice. Second, VOMMs are barely capable of benefiting from statistical dependencies, no matter how well the structure prior is chosen. This raises the question why PMMs are capable of finding a good compromise between modeling dependencies and avoiding overfitting whereas VOMMs are not.

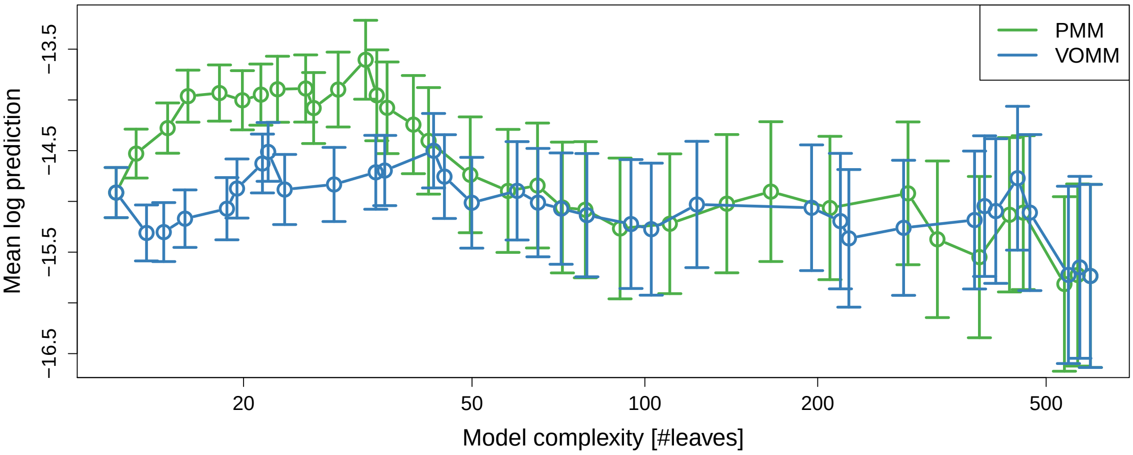


Figure 2.2.: **Prediction versus model complexity for PMMs and VOMMs.** For both model classes, we plot the mean logarithmic prediction resulting from a leave-one-out cross-validation experiment on the C/EBP data set against different model complexities, which are proportional to the number of parameters, obtained by varying the structure prior hyperparameter κ. Error bars show double standard error.

2.3.2. Comparing tree structures

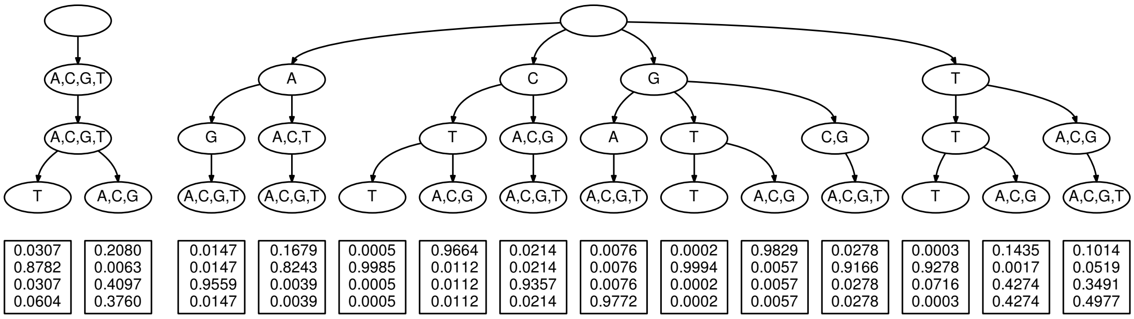
In a second study, we attempt to answer that question by comparing learned model structures of PMM and VOMM. We choose for both model classes the values of κ that yield the highest mean log prediction in the leave-one-out CV experiment of Figure 2.2. For the PMM, this is $\kappa = e^{-2.5}$ with an average leaf number of 32.6 and a mean log prediction of -13.6, whereas for the VOMM this is $\kappa = e^{-1.8}$ with an average leaf number of 42.8 and a mean log prediction of -14.5. We use those structure priors to learn two models on the complete C/EBP data set of 96 sequences and scrutinize the resulting models in the following.

The resulting PMM and VOMM have 32 and 43 leaves respectively, which is in resonance with the average leaf numbers of the leave-one-out CV experiment. First, we analyze how the total leaf numbers of both models distribute over the 12 trees (Table 2.1).

Even though the VOMM has a higher total leaf number than the PMM, this does not apply for each of the 12 individual trees. In some cases (positions 5, 8, 9, and 11), the CT of

Table 2.1.: Leaf numbers for all trees of best third-order PMM and VOMM.

Position	1	2	3	4	5	6	7	8	9	10	11	12	Σ
PMM	1	1	4	3	2	3	1	2	6	5	3	2	32
VOMM	1	1	4	1	12	2	1	3	8	2	7	1	43



(a) PCT (b) CT

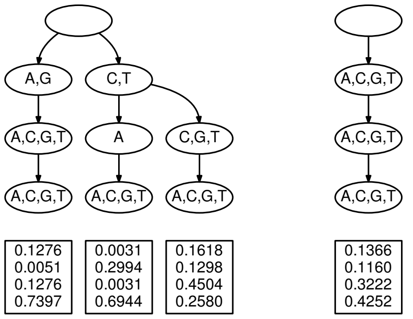
Figure 2.3.: **PCT and CT at position 5 of the C/EBP motif.** We choose for PMM and VOMM the optimal structure prior constant κ with respect to the leave-one-out experiment of Figure 2.2. Next, we learn both models using their respective optimal structure prior on the complete data set of 96 sequences and depict both the PCT of the PMM and the CT of the VOMM at position 5.

the VOMM is indeed more complex than the PCT of the PMM. In other cases (positions 4, 6, 10, and 12) the opposite holds, even though the absolute difference in complexity is here generally smaller, which is the reason of the overall higher complexity of the VOMM. So it might be worthwhile to compare PCT and CT structures for both groups in detail. To this end, we choose position 5 and position 4, both representing extreme cases within their own groups.

In Figure 2.3, we show the PCT of the PMM and the CT of the VOMM at position 5. Since tree structures can be only partially interpreted without knowing the underlying conditional probability distributions, we plot for both trees the conditional probability parameters for each context, estimated according to Equation 2.8, in rectangular boxes below the corresponding leaf node.

The PCT in Figure 2.3(a) has only two leaves, so it partitions all context words into only two sets. The first and the second layer of the tree are completely fused, so the first and second predecessor symbol does not contribute any information to the probability distribution at position 5. At the third layer, however, the context words are partitioned in two subsets according to the observed symbol at the third predecessor (position 2). Observing a `T` at position 2 yields a high conditional probability of 0.8782 for finding a `C` at position 5, whereas any other symbol at position 2 yields a low conditional probability of 0.063 for a `C` at the fifth position. Conversely, for the second context, the conditional probability of finding `A`, `G`, and `T` is highly increased. This shows that there is a strong statistical dependence between positions 5 and 2, and a PCT is capable of exploiting it with only two parameters sets, which can be estimated comparatively robustly from 96 data points (partitioned into two sets of sizes 67 and 29 respectively).

The CT in Figure 2.3(b) has twelve leaves, but many of the contexts represent only



(a) PCT (b) CT

Figure 2.4.: **PCT and CT at position 4 of the C/EBP motif.** The experimental setup is identical to that of Figure 2.3.

few context word occurrences in the data set. For example, the first, fourth, and ninth leaves represent only a single sequence in the data set each. Hence, the reliability of the corresponding parameter estimates is highly questionable. The reason why such a context tree is learned despite the indication of overfitting is the strong statistical dependence among position 5 and 2. Leaves number two, three, seven, and ten represent most of the context words that are combined in the first leaf of the PCT in Figure 2.3(a). But since a CT does not allow a split in the tree structure below a fused node, the only possibility to learn this third-order dependency is a broad tree with many dispensable parameter sets.

We conclude that one reason for the inability of the VOMM to effectively capture dependencies in this data set is its structural limitation of not being capable of "skipping" a position, which may lead to strong overfitting, if skipping positions would actually be required.

In Figure 2.4, we display the PCT and CT at position 4. The CT of Figure 2.4(b) is completely pruned, resulting in a minimal tree corresponding to full statistical independence. At this position the VOMM does not suffer from overfitting, but it may neglect existing dependencies.

The PCT at the same position has three leaves, resulting in three different parameter sets. Each leaf represents a substantial amount of sequences from the data set (24, 10, 62), so the parameter estimates may not be completely unreliable. We observe that the first leaf yields a high conditional probability of 0.7397 for a `T`, given the symbol on the preceding position being either `A` or `G`. The second and third leaves represent the other contexts that have either `C` or `T` at the previous position. In addition, the second leaf represents the subset of context words that have an `A` at the second predecessor (position 2). The corresponding conditional probability of a `T` is 0.6944, whereas `A` and `C` rarely occur. If the symbol at the second predecessor is not `A`, then `G` has the highest probability at position 4 (third leaf).

This implies that a certain amount of statistical dependencies exist among the fourth position of the C/EBP data set and its predecessors, and that these dependencies can be modeled – at least to some degree – by a PCT. A PCT is capable of splitting the contexts at any layer so that there is more than one fused node. This feature may be required to properly represent statistical dependencies at position 4. A CT is not capable representing these splits, so it here rather neglects statistical dependencies completely. This indicates that VOMMs are not necessarily always overfitted compared to PMMs, but also the opposite, underfitting due to structural limitations, may occur.

We may conclude that, compared to the third-order PMM, the third-order VOMM is both over- and underfitted. The PMM is better capable of using the full potential of the inhomogeneity of the model, since it yields – in average over all positions – a better tradeoff between capturing dependencies and reducing the parameter space.

2.3.3. Model validation

The previous two experiments show that a PMM is capable of modeling dependencies in a small real world data set and how it finds a reasonable balance in avoiding both over- and underfitting due to its structural flexibility. However, as observed in Figure 2.2, the prediction performance depends on the choice of the structure prior, for which true a priori knowledge is rarely available.

Hence, we must devise a method that can automatically provide us with a reasonable choice for κ in order to validate the model class against other alternatives. To this end, we perform in each step of the CV described in Section 2.3.1, another internal CV on the 95 training sequences. We then choose in the i-th step that κ that yields the highest mean log prediction in the CV on the 95 training data sequences, learn a model on that data set, and compute the predictive probability of the i-th sequence. Finally, we average all logarithmized predictive probabilities and use that single number for evaluating the performance of the model class.

In Figure 2.5, we compare PMMs, VOMMs, and Markov models of order 1-3. In addition, we also consider the independence model (PWM model), which neglects all dependencies but is despite its simplicity to date the most popular choice for modeling DNA binding sites as discussed in Section 1.3. For the independence model and the fixed order Markov models, there is no internal cross-validation.

We find that the independence model yields a mean log prediction value of -14.91. A first-order Markov model improves it to a value of -14.56, showing that taking into account first-order dependencies is reasonable. Second- and third-order Markov models yield a lower prediction than the independence model. This is not surprising since we expect overfitting for complex models when sample size is as small as 96 data points.

First- and second-order VOMM are on par with the independence model. Despite reducing the parameter space, they are – on this data set – not capable of utilizing statistical dependencies effectively. The third-order VOMM yields even a lower prediction, comparable to a

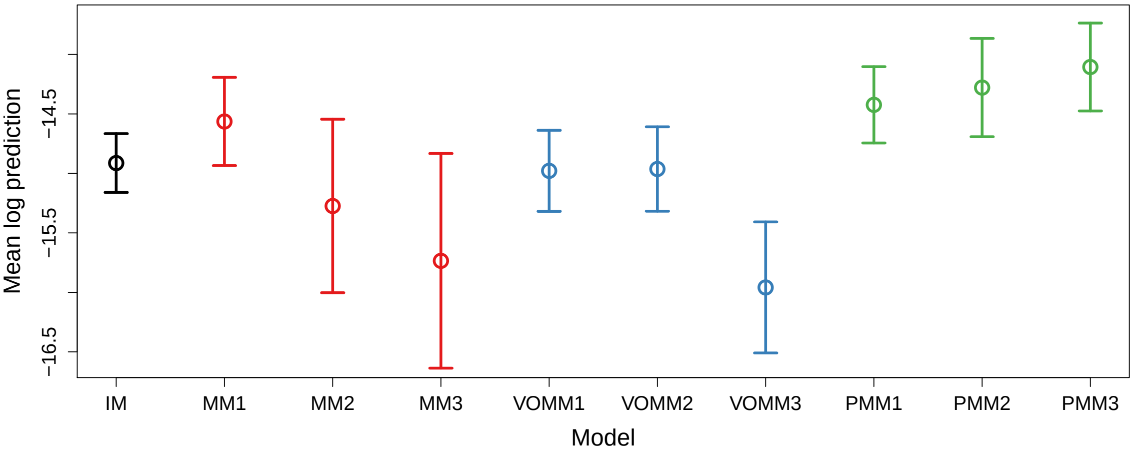


Figure 2.5.: **Prediction performance of different models.** We show the prediction performance of the independence model (IM), fixed order MMs, VOMMs, and PMMs of orders 1-3. The experimental setup is identical to that of Figure 2.2. For the parsimonious and variable order models, we perform an additional internal cross-validation on the $N - 1$ training sequences for determining the optimal structure prior hyperparameter κ.

third-order Markov model, hence clearly overfitting occurs. This also shows that here the internal CV fails, since it selects – for some iteration steps – at more than one position very complex and thus poorly generalizing tree structures, comparable to that in Figure 2.3(b).

The first-order PMM yields a mean log prediction value of -14.42, which is comparable to that of the first-order Markov model. Apparently, overfitting is not a serious problem for the first-order Markov model, so the potential reduction of the parameter space yields only a small improvement. In contrast to fixed order MMs, PMMs of second and third order continue to increase prediction performance. The overall best prediction comes from a third-order PMM with a mean log prediction value of -14.1, which is slightly lower than the best prediction in Figure 2.2, but close to the average prediction within the range of reasonable complexities (15 to 40 leaves).

We summarize that PMMs yield a higher prediction of C/EBP binding sites than the independence model, than fixed order Markov models, and than variable order Markov models. Among the three PMMs, the third-order PMM yields the overall highest prediction. Hence, PMMs are capable of exploiting dependencies in the small data of only 96 sequences set effectively, whereas VOMMs are incapable of achieving that due to their structural limitations. This makes it tempting to speculate that PMMs might be a reasonable model class for other types of sequential data as well, especially when certain dependencies among non-neighboring positions exist and when the sample size is comparatively small. In this thesis, we do not pursue the road of different applications any further, but rather use the promising results as an incentive for studying structure and parameter learning within this model class itself in more detail.

Bottom line

In this chapter, we introduced inhomogeneous PMMs as a combination of inhomogeneous Markov models and parsimonious context trees, completing the first task within main objective I (Figure 1.11). Following a Bayesian approach, we proposed a learning scheme based on Bayesian model selection for structure learning and parameter estimation through the mean posterior principle. We demonstrated that using inhomogeneous PMMs is justified, since they can yield an increased prediction performance in comparison to inhomogeneous MMs and VOMMs. However, we also observed that the prediction performance depends heavily on the choice of the prior. The uniform structure prior appeared to be not the optimal choice, making cross-validation for hyperparameter-tuning necessary. In the next chapter, we thus investigate alternative learning principles in order to avoid time-consuming hyperparameter-tuning.

Since all models are wrong the scientist cannot obtain a "correct" one by excessive elaboration. On the contrary following William of Occam he should seek an economical description of natural phenomena. Just as the ability to devise simple but evocative models is the signature of the great scientist so overelaboration and overparameterization is often the mark of mediocrity.

George E. P. Box

3

Robust learning

In the previous chapter, we learned that PMMs constitute a promising alternative to VOMMs and traditional MMs of fixed order. In order to infer PMMs from data, we proposed a Bayesian learning approach that involves a specification of structure prior and parameter prior, which we both defined to depend on a single hyperparameter each (κ and η). Using Bayesian model selection, we were capable of learning a set of PCTs, which is optimal according to the Bayesian marginal likelihood. We observed that (i) the choice of the hyperparameters is critical in order to obtain models of appropriate complexity, and (ii) straightforward choices, such as $\kappa = 1$ (uniform structure prior) and $\eta = 1$ do not yield optimal solutions. In order to cope with this problem, we tuned the hyperparameter κ via internal leave-one-out CV on the training data in order to yield model structures that are optimal for prediction on this particular data set.

In doing so, we designed a learning scheme that does not actually perform Bayesian model selection, but rather model selection via CV. The Bayesian formulation of the optimization problem and the DP algorithm for solving it are essentially only tools to reduce the exponentially large space of possible model structures, which cannot all be tested via CV, to a relatively small number of candidates for κ. Whereas generalizing a known model by introducing additional parameters followed by exhaustive tuning of those parameters on data is a popular strategy in statistical learning, the usefulness of such an approach should be put into question.

A critical mind may point out that it is not a particularly elegant but rather brute-force learning approach, and this criticism is valid indeed. Depending on the size of the problem at hand, hyperparameter-tuning by CV is not just inelegant, but also infeasible for large-scale applications.

The deeper reason for the need of hyperparameter-tuning is the need for a prior in order to apply the machinery of Bayesian model selection, despite the fact that we do not actually have reasonable prior belief. The choice of the parameter prior is mainly influenced by mathematical convenience, as the most important property of a prior distribution is to be conjugate so that the integrals in the derivations can be solved analytically. The reduction to a single hyperparameter, which is often called concentration parameter as it determines the concentration of the probability mass in the symmetrical Dirichlet distribution, is simply a consequence of lack of knowledge about the biological system at hand. A uniformly chosen structure prior may also express our absence of prior belief, but it does not yield optimal results as we observed in Chapter 2.

In this chapter, we further study the empirical behavior of the Bayesian learning approach. In addition, we investigate alternative possibilities of learning PMMs that avoid the specification of hyperparameters and compare them with the Bayesian approach. In order to do so, we briefly introduce the *Minimum Description Length* (MDL) principle in the following section, while referring for a more detailed introduction to a well-written tutorial [81].

3.1. The MDL principle

The MDL principle, which was proposed by Rissanen is a series of publications starting in 1978 [82], is a statistical inference method that has its roots in information theory, which is in contrast to both frequentist and Bayesian methods, which are rooted in probability theory. Probability theory assumes the data to be generated from a distribution of a model, and the learning task is then formulated with the goal of inferring this "true" model and distribution from data. However, in reality it is often debatable whether such a true model exists at all, and the applications covered in this thesis are by no means an exception. There is probably no molecular biologist who assumes that a TFBS was actually generated from on a parsimonious Markov model, a Bayesian network, or any other statistical model, but nevertheless the learning principles dealt with so far inevitably make this assumption.

The MDL principle takes a different viewpoint, dismissing any notion of a true model and a true distribution that did generate the data. Instead, the MDL philosophy only attempts to find a model that exploits regularities in the data well, which Grünwald summarized in the following statement [83]:

"The fundamental idea behind the MDL principle is that any regularity in a given set of data can be used to compress the data, i.e. to describe it using fewer symbols than needed to describe the data literally."

In other words, it is valid to state that, according to MDL, learning from data is equivalent to compressing data, i.e., finding an optimal data compression is equivalent to finding an optimal representation of the regularities in the data, which is in the end the very purpose of statistical learning.

3.1.1. Codelengths and probabilities

While the relationship between data compression and assignment of probabilities to certain events may not obvious at first glance, there exists indeed a direct connection between both concepts. Consider an arbitrary sequence of discrete observations $\vec{x} \in \mathcal{A}^N$, which could be either a DNA sequence or some text of natural language, for instance. In order to encode the data with the aim of being able to unambiguously decode it at a later stage, we define a prefix code $C : \mathcal{A} \to \{0,1\}^*$, using codewords $C_1, \ldots, C_{|\mathcal{A}|}$ of possibly different lengths $\mathcal{L}_1, \ldots, \mathcal{L}_{|\mathcal{A}|}$. The Kraft–McMillan inequality [84, 85] now states that for any prefix code C

$$\sum_{i=1}^{|\mathcal{A}|} 2^{-\mathcal{L}_i} \leq 1 \tag{3.1}$$

holds. If the inequality is strict, C is partially redundant, and a better code for the data exists. If we obtain an equality, C is a complete code. For a given code, we consider the right side of the Kraft–McMillan inequality as constant, i.e., $z = \sum_{i=1}^{|\mathcal{A}|} 2^{-\mathcal{L}_i}$.

Now we define a probability distribution $\vec{p}$ by

$$\forall_{i=1}^{|\mathcal{A}|} : p_i = \frac{2^{-\mathcal{L}_i}}{z} \tag{3.2}$$

from which follows that

$$\forall_{i=1}^{|\mathcal{A}|} : \mathcal{L}_i = \lceil -\log_2 z p_i \rceil . \tag{3.3}$$

In case of a complete code, z vanishes from both equations. With Equation 3.2 and Equation 3.3 we now have established the basic connection between code lengths and probabilities: high probabilities correspond to short code lengths and vice versa. This is quite intuitive, as it is reasonable to assign the shortest code lengths to the words that appear most frequently in the data when the goal is to minimize the total code length.

It is common to assume two further idealizations [86]: First, code lengths are allowed to assume non-integer values, which guarantees the existence of a complete code for any probability distribution, rendering the ceiling in Equation 3.3 needless. Second, instead of the binary logarithm, which yields code lengths in bits, often the natural logarithm is used, yielding a code lengths in nats. Under these assumptions, the relationship between code lengths and probabilities is simply

$$\forall_{i=1}^{|\mathcal{A}|} : \mathcal{L}_i = -\log p_i . \tag{3.4}$$

In Chapter 2, we evaluated different models for the task of learning a probability distribution for each model from training data and and computing the (log)-probability of previously unseen test data. This task could have been also phrased as optimizing a code length function on training data and evaluating the model by the code length on previously unseen test data.

The MDL principle follows the second notion and seeks the model that yields the minimal description length. However, description length does not only involve the code length of the data given a particular model, but also the code length of the model itself, which serves as regularization to avoid overfitting. Here it is where different notions of implementing the MDL principle enter the stage. In the following, we consider arbitrary data $\mathbf{x} \in \mathcal{X}$, a model (structure) $\xi \in \mathcal{S}$, and the corresponding parameter space θ_ξ, which contains all probability distributions of the model.

3.1.2. Two-part code

One straightforward instantiation of the MDL principle seeks

$$\hat{\xi} = \min_{\xi \in \mathcal{S}} \mathcal{L}(\mathbf{x}|\theta_\xi) + \mathcal{L}(\theta_\xi) \tag{3.5}$$

which literally minimizes the sum of the code length needed for describing the data given the model plus the code length for describing the model itself, and is thus called *two-part code*. While $\mathcal{L}(\mathbf{x}|\theta_\xi)$ is essentially the negative log-likelihood of the data given the model due to the direct relationship between probabilities and code lengths, it is not immediately clear which form $\mathcal{L}(\theta_\xi)$ assumes. Rissanen proposed a quantization of the continuous parameter space with precision $\frac{1}{\sqrt{N}}$, with N being the number of data points [82], i.e., θ is discretized using $\sqrt{N}$ bins. Let $k = |\theta_\xi|$ denote the dimensionality of the parameter space of model ξ, then the number of possibly different parameter values is given by $\sqrt{N}^k$. We now consider the logarithm of this quantity to correspond to $\mathcal{L}(\theta_\xi)$, the code length of the model. MDL model selection using the two-part code formulation thus assumes the form

$$\hat{\xi}_{\text{tpc}} = \min_{\xi \in \mathcal{S}} \left(-\log_2 P(\mathbf{x}|\hat{\theta}_\xi(\mathbf{x})) + \frac{k}{2} \log_2 N \right) \tag{3.6}$$

where $\hat{\theta}_\xi(\mathbf{x})$ denotes the maximum likelihood estimate of θ_ξ on $\mathbf{x}$, and tpc is here the abbreviation for two-part code. Hence, the MDL two-part code corresponds to the Bayesian Information Criterion (BIC) [87], even though that is derived from an entirely different perspective, namely as large-sample approximation of the Bayesian marginal likelihood.

3.1.3. Normalized Maximum Likelihood distribution

The two-part code is one approach of MDL model selection, albeit not the only one. Recall that the goal is to select one model ξ from a model class $\mathcal{M}$. A different approach is to associate each $\xi \in \mathcal{M}$ with one distribution $P_\xi(\mathbf{x})$ that is representative for all distributions that are contained in the model ξ[1]. The code length of the data $\mathbf{x}$ under such a *universal distribution*, which does not need to be a member of $\mathcal{M}$, is then considered as the code length of the data under the corresponding model.

---

[1] Recall that a model is a family of probability distributions.

It remains to be decided which properties a distribution has to possess in order to be universal, i.e., to be representative for an entire family of distributions. The best possible distribution in ξ for $\mathbf{x}$ is the maximum likelihood distribution $P(\mathbf{x}|\hat{\theta}_\xi(\mathbf{x}))$. A distribution Q is called universal distribution (relative to ξ) if

$$\lim_{N\to\infty} \frac{1}{N}\left[\log\frac{1}{Q(\mathbf{x})} - \log\frac{1}{P(\mathbf{x}|\hat{\theta}_\xi(\mathbf{x}))}\right] = 0. \tag{3.7}$$

The term within the brackets, which is called *regret*, can be interpreted as the difference between the code length of the distribution Q and the best possible distribution in the family. A distribution is thus called universal w.r.t. a model if its regret grows sub-linear with sample size. In addition, the regret is also used to evaluate the quality of a universal distribution.

We now observe that the two-part code defines a universal distribution, since its regret is $\frac{k}{2}\log_2 N$ which grows only logarithmic with sample size N. However, there are universal distributions that yield a smaller regret and could thus be considered to be more suitable for representing a family of distributions. One example is the (Bayesian) mixture distribution, which is equivalent to the Bayesian marginal likelihood, and its regret has been shown to be upper-bounded by that of the two-part code [88], which is not surprising since BIC has been shown to be asymptotically equivalent to the Bayesian marginal likelihood.

An alternative is the *normalized maximum likelihood* (NML) distribution [89], which is defined as

$$P_{\text{NML}}(\mathbf{x}) = \frac{P(\mathbf{x}|\hat{\theta}_\xi(\mathbf{x}))}{\sum_{\mathbf{x}} P(\mathbf{x}|\hat{\theta}_\xi(\mathbf{x}))} \tag{3.8}$$

It can be easily shown that the regret of the NML is simply the logarithm of the denominator, which is constant w.r.t. to $\mathbf{x}$. From that follows that the NML distribution is the solution to the minimax problem

$$\min_Q \max_{\mathbf{x}} \left[\log\frac{1}{Q(\mathbf{x})} - \log\frac{1}{P(\mathbf{x}|\hat{\theta}_\xi(\mathbf{x}))}\right] = P_{\text{NML}}, \tag{3.9}$$

which means that it minimizes the worst-case regret.

The main challenge in the practical application of NML is the computation of the denominator, which requires to sum over all possible data sets of size N, and this summation becomes intractable quickly. Hence, exact computation of the NML normalizing constant is to date possible only for a handful of simple models [90, 86, 91, 92], but approximations for more complex settings are available [93].

3.2. Alternative learning approaches for PMMs

Learning PMMs consists of (i) structure learning of every PCT in the model, and (ii) estimation of the condititional probability parameters of each PCT as discussed in Chapter 2. Both tasks can be carried out according to different learning principles, some of which are based on Bayesian statistics, whereas others originate from the MDL principle. In this section, we summarize scoring functions for optimizing the structure and estimating the probability parameters that are offered by the different learning principles and are applicable to PMMs.

3.2.1. Structure scores

Structure learning is algorithmically challenging, but can be solved by the dynamic programming algorithm (Section 1.4.2), which can be used for optimizing any score function that satisfies the so called decomposability property [60].

In the Bayesian setting, the structure score of the ℓ-th PCT is proportional to the local posterior probability $P(\tau_\ell|\mathbf{x})$ of the PCT τ_ℓ given data set $\mathbf{x}$, which may be further conditioned upon prior hyperparameters. Using the prior specification of Section 2.2.2, we obtain

$$S_{\mathrm{BDeu}}(\tau_\ell|\mathbf{x},\eta,\kappa) = \sum_{c\in\tau_\ell} \log \frac{\kappa\mathcal{B}\left(\vec{N}_{\ell c} + (\alpha,\ldots,\alpha)_{|\mathcal{A}|}\right)}{\mathcal{B}\left((\alpha,\ldots,\alpha)_{|\mathcal{A}|}\right)}, \tag{3.10}$$

with $\alpha = \frac{\eta|c|}{|\mathcal{A}|^{d+1}}$, where η is the equivalent sample size (ESS) parameter. $\mathcal{B}$ denotes here the multinomial beta function, and $(a,\ldots,a)_b$ denotes a b-dimensional vector filled with constants a. This score is equivalent to the *BDeu* score of Bayesian networks and we name it accordingly. The only conceptual difference is the existence of a second hyperparameter, κ, which originates from the structure prior and is as such a peculiarity of PMMs. Setting $\kappa = 1$ corresponds to a uniform distribution over all structures, which then yields a BDeu score in its most widely used form [94].

Since the Bayesian scoring criterion is similar to that of Bayesian networks, it is natural to apply other scoring criteria that have been proposed for that model class to PMMs as well. One possible alternative is the *factorized NML* (fNML) criterion [95] for learning Bayesian networks, in order to approximate the NML distribution of the full model by a product of independently normalized terms. For PMMs, the fNML score can be written as

$$S_{\mathrm{fNML}}(\tau_\ell|\mathbf{x}) = \sum_{c\in\tau_\ell} \log \left(\frac{\left(\frac{N_{\ell ca}}{N_{\ell c\cdot}}\right)^{N_{\ell ca}}}{C^{|\mathcal{A}|}_{N_{\ell c\cdot}}} \right), \tag{3.11}$$

where C^a_b is the stochastic complexity of a multinomial distribution with a being the number of categories and b being the sample size. C^a_b can be computed using a linear-time recursive algorithm [90] or by the so called Szpankowski approximation [96]. The fNML score as such does not require any explicit prior assumptions. However, it is close to the Bayesian marginal likelihood using a Jeffreys' prior, which is a Dirichlet distribution with hyperparameters $\frac{1}{2}$ in this setting, thus violating the equivalent sample size condition. In such a Bayesian interpretation, using fNML would also imply using a uniform prior over all model structures. The fNML score yields a consistent estimator of the model structure [95].

A generally applicable score, which has an interpretation both in Bayesian statistics and in information theory, is the *Bayesian Information Criterion* (BIC) [87]. While originally derived as a large-sample approximation of the Bayesian marginal likelihood, it is sometimes also referred to as MDL score, since it corresponds to the two-part encoding of model structure and data given the model as discussed in Section 3.1.2. The BIC score can be written as

$$S_{\mathrm{BIC}}(\tau_\ell|\mathbf{x}) = \sum_{c\in\tau_\ell} 2\log\left(\left(\frac{N_{\ell ca}}{N_{\ell c\cdot}}\right)^{N_{\ell ca}}\right) - |\tau_\ell|(|\mathcal{A}|-1)\log(N). \tag{3.12}$$

It is known to penalize additional parameters rather strictly, so it is typically more prone to underfitting than to overfitting. Furthermore, BIC can be seen as an approximation of both the fNML score and the Bayesian marginal likelihood.

Another generally applicable option is the Akaike Information Criterion (AIC) [97], which can be written as

$$S_{\mathrm{AIC}}(\tau_\ell|\mathbf{x}) = \sum_{c\in\tau_\ell} 2\log\left(\left(\frac{N_{\ell ca}}{N_{\ell c\cdot}}\right)^{N_{\ell ca}}\right) - 2|\tau_\ell|(|\mathcal{A}|-1). \tag{3.13}$$

In contrast to the aforementioned methods, AIC is not consistent. However, it is generally expected to perform better than BIC in cases where the true model is not in the set of candidates. There is a refinement of AIC for finite sample sizes [98], but that score is not decomposable among random variables, which is a key property required for learning PMMs using the dynamic programming algorithm for finding optimal PCTs.

3.2.2. Parameter estimates

Once the model structure is learned, we also need to fix the model parameters in order to be able to use the resulting probability distribution for tasks such as prediction.

In the Bayesian setting, the parameters can be estimated according to the mean posterior principle [79], yielding

$$\hat{\theta}^{\mathrm{MP}}_{\ell ca}(\eta,\mathbf{x}) = \frac{N_{\ell ca} + \frac{\eta|c|}{|\mathcal{A}|^{d+1}}}{N_{\ell c\cdot} + \frac{\eta|c|}{|\mathcal{A}|^{d}}}, \tag{3.14}$$

when following the prior specification of Section 2.2.2. For PMMs, mean posterior estimates directly correspond to a prediction method integrating over the parameter space as discussed in Section 2.2.3, hence they are in resonance to structure learning using the Bayesian marginal likelihood of Equation 3.10.

An alternative is here also offered by the NML distribution. For parameter learning in Bayesian networks, *factorized sequential NML* (fsNML) estimates have been proposed in order to estimate probability parameters in accordance with the fNML structure learning score [93]. For PMMs, the fsNML estimate writes as

$$\hat{\theta}_{\ell ca}^{\mathrm{fsNML}}(\mathbf{x}) = \frac{e(N_{\ell ca})(N_{\ell ca}+1)}{\sum_{b\in\mathcal{A}} e(N_{\ell cb})(N_{\ell cb}+1)}, \tag{3.15}$$

where $e(N) = (\frac{N+1}{N})^N$ for $N > 0$ and $e(0) = 1$. The derivation of the fsNML estimate from $P_{\mathrm{sNML}}(y|\vec{x})$, the sequential NML distribution of a single symbol given data, is shown in Appendix Section B.2. Due to its minimax optimality properties, using fsNML for sequential prediction, where the t-th prediction is based on the $t-1$ previous observations, $t \in \{1, \ldots, N\}$, yields a predictive performance that is almost as good as the optimal parameter estimates, which are obtained using the maximum likelihood estimator with full data [93].

3.3. Empirical comparison

In this section, we empirically compare the Bayesian and MDL-based methods for structure learning and parameter estimation. We study the behavior of the aforementioned methods with respect to the choice of the prior hyperparameter (Section 3.3.1) and the sample size (Section 3.3.2) using the splice site data set of Yeo and Burge, which was originally used for evaluating Maximum Entropy Models [99]. It consists of 12,624 experimentally verified human splice donor sites, which are the sequences of length $W = 7$ over the four letter alphabet $\mathcal{A} = \{\mathtt{A},\mathtt{C},\mathtt{G},\mathtt{T}\}$. These sequences have been split into training data ($\mathbf{x}_{\mathrm{train}}$) and test data ($\mathbf{x}_{\mathrm{test}}$) at a ratio of 2:1 [99], and we rely on the same partion for the following experiments. This data set is particularly suitable for comparing scoring criteria for PMMs since (i) it is known that comparatively strong dependencies among adjacent positions in the sequences exist and (ii) the large number of data points originating from the same source offers the possibility to study the influence of the sample size. As such, we use splice donor sites mainly as benchmark data, whereas we evaluate TFBS data, which might be a practically more relevant application of PMMs, in Section 3.3.3.

3.3.1. Influence of the ESS

In a first study, we compare the performance of Bayesian and NML-approximating scoring criteria.

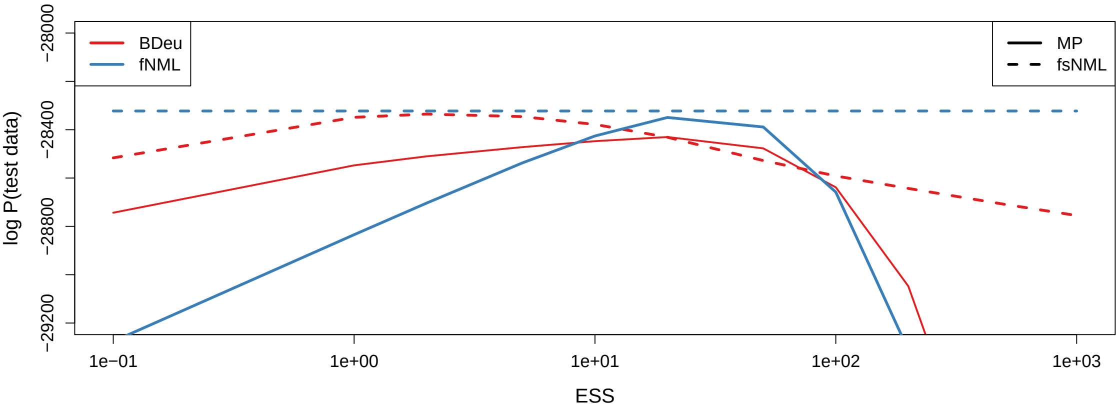


Figure 3.1.: **Prediction performance versus ESS hyperparameter.** We compare different combinations of structure scores and parameter estimates on the splice site data. The structure score is indicated by the color of each line, with BDeu displayed in red and fNML displayed in blue. The parameter estimate is indicated by the shape of the line, with solid being MP and dashed being fsNML. Hence, the solid red line displays the traditional Bayesian method, whereas the dashed blue line displays the hyperparameter-free NML method. The other two lines are influenced by the ESS either only in structure learning (dashed, red) or in parameter learning (solid, blue).

We sample $N = 500$ sequences from $\mathbf{x}_{\text{train}}$, learn structure $\hat{\vec{\tau}}$ and probability parameters $\hat{\vec{\theta}}_{\hat{\vec{\tau}}}$ of a third-order PMM, and compute $\log P(\mathbf{x}_{\text{test}}|\hat{\vec{\Theta}})$, where $\hat{\vec{\Theta}} = (\hat{\vec{\tau}}, \hat{\vec{\theta}}_{\hat{\vec{\tau}}})$. We repeat this procedure 10^3 times, and average the resulting log predictive probabilities in order to let the error caused by subsampling become negligible.

Using this procedure, we compare the performance of the NML-approximating method, using fNML as structure score and fsNML as parameter estimate, with the Bayesian method, using BDeu as structure score and MP as parameter estimate. The latter offers the possibility to incorporate prior knowledge through the equivalent sample size η and the structure prior hyperparameter κ. Recall that setting the structure prior hyperparameter $\kappa = 1$ results in a uniform structure prior, so that we are able to separately investigate the influence of the ESS. In addition, we perform cross-comparisons by combining the fNML structure score with the MP parameter estimate and the BDeu structure score with the fsNML parameter estimate. While theoretically difficult to justify, this might be helpful to evaluate – at least to some extent – the influence of the ESS on structure and parameter learning separately. In Figure 3.1, we plot the prediction performance of the various combinations of structure scores and parameter estimates against the ESS, ranging from 10^{-1} to 10^3.

We observe that the performance of the Bayesian method (BDeu-MP) depends strongly on the ESS, with too large values leading to more dramatic degradation in performance than too small values. Moreover, the NML-approximation (fNML-fsNML) yields a higher prediction than the Bayesian method, no matter which ESS is chosen for the latter. This is surprising, since intuitively the Bayesian method should excel for at least some particular choices of the prior – even if those well-performing choices are typically unknown.

We find a possible explanation of this observation by investigating the cross-comparisons of Bayesian and NML-approximating methods. Since the optimum of the BDeu-fsNML curve (dashed, red) is located at a smaller ESS value than the fNML-MP curve (solid, blue), the optimal ESS for structure learning seems to be not necessarily the optimal ESS for parameter learning. With the comparatively small sample size of $N = 500$, structure learning requires a small ESS, whereas parameter learning requires a larger ESS, providing a better parameter-smoothing.

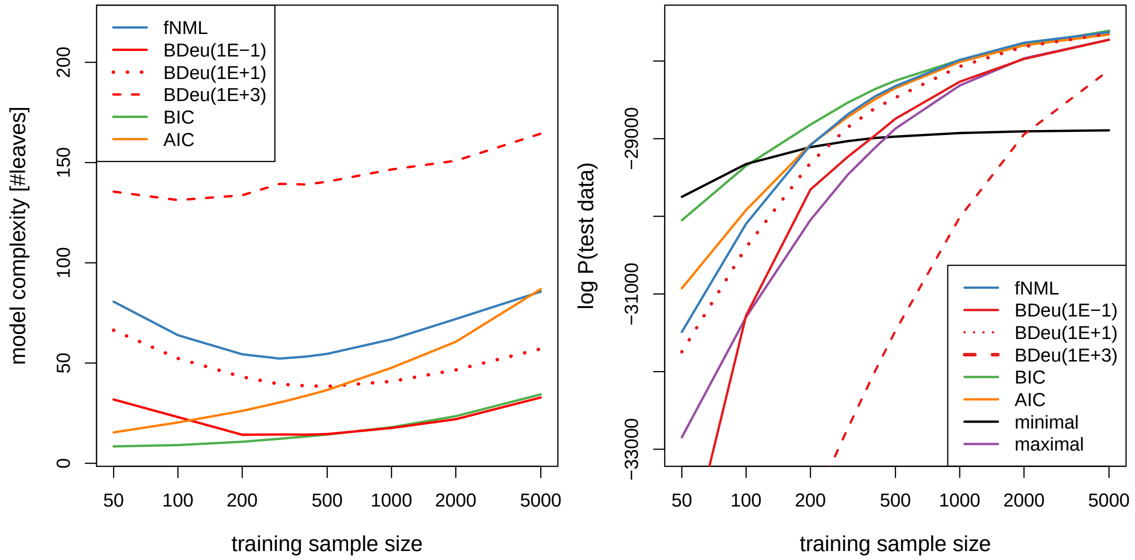
We also observe that non-optimal choices of the ESS have worse consequences for parameter learning than for structure learning. The error arising from a false model structure is bounded by the prediction performance of the most overfitting model structure, which corresponds here to a third-order Markov model. However, the error that may arise from parameter learning is virtually unlimited, since the parameter estimates may get totally dominated by the prior, either yielding estimates close to maximum likelihood (very small ESS) or blurred towards a uniform distribution (very large ESS). We may thus conclude that fsNML is here a safe choice for parameter learning, as it avoids the explicit specification of any parameter prior.

3.3.2. Influence of the sample size

After having studied the effect of the ESS at one particular sample size, we now focus on the influence of the sample size on structure learning and prediction. Here, we take a closer look at structure learning, by investigating the influence of different structure scores on the complexity of the learned models. To this end, we use a similar experimental setting as described in Section 3.3.1. We sample N sequences from $\mathbf{x}_{\text{train}}$, learn the structures $\hat{\vec{\tau}}$ of third-order models with different scoring criteria, and compute the total number of leaves of all PCTs in a learned model structure, which is proportional to the number of model parameters and can thus be referred to as *model complexity*. We repeat the procedure 10^3 times and average the resulting model complexities. We perform this procedure for different values of N, ranging from 50 to 5000, and plot in Figure 3.2(a) the model complexity against the sample size.

For the BDeu score, we observe that the model complexity depends on the ESS: the larger ESS, the larger is the model. This effect has been extensively studied for Bayesian networks [100], so it is not surprising that the same applies for PMMs as well.

For all methods, there is a general trend of increasing model complexity with increasing sample size from $N = 500$ onwards, but even when using 5000 data points, we do not get close to the maximal model, which has 277 leaves. However, for BDeu and fNML the complexity also increases when sample size becomes very small ($N < 300$). Whereas this seems to be counter-intuitive at first glance, it can be explained with an extreme case: When the sample size decreases to zero, there is no observed data, all terms originating from the parameter



(a) Model complexity (b) Prediction

Figure 3.2.: **Influence of sample size on model complexity and prediction.** Figure (a) shows the complexity of the learned model structures with respect to the sample size and different structure scores: BIC, AIC, fNML and BDeu with three different ESS values. Figure (b) shows the corresponding prediction performance versus sample size using fsNML parameter estimates for all scores but BDeu, which uses MP parameter estimates. In addition, the performance of the minimal model (independence model) and the maximal model (third-order Markov model) are shown.

prior cancel out, and as a consequence the structure prior dominates model selection. If we assume that for identical optimal scores one of the candidate structures is selected at random, we obtain models of average complexity w.r.t. the total space of candidate structures. When observing few data points, BDeu collects evidence in favor of either simple or complex models, but the starting point is – in accordance with the uniform structure prior – a model of average complexity. Since fNML implicitly also assumes a uniform structure prior, it is not surprising that a similar effect appears for that score as well. Even if the data was actually generated by an independence model, and thus contained no statistical dependencies at all, many data points would be required to convince both scores of such an extreme hypothesis. BIC, on the other hand, shows a different behavior, as it is known to penalize model complexity heavily in general. But in contrast to BDeu(1E-1), which is similar to BIC for large sample sizes, the model complexity for BIC is almost monotonically increasing, and for very small sample sizes BIC selects almost an independence model.

In a third study, we investigate how different structure scores influence the prediction performance when the sample size varies. We mostly focus on prediction using the fsNML parameter estimate, and compare BIC, AIC and fNML structure scores with the extreme cases of independence model and third-order Markov model. We also include the Bayesian method of BDeu score and MP parameter estimate in the comparison, using the same ESS values as in Figure 3.2(a). The experimental setup is here identical to that of the previous experiment, but now we compute predictive probabilities for different sample sizes N and show the results in Figure 3.2(b).

First, we observe that the independence model is optimal when the sample size is smaller than 100 data points. This is intuitively clear since all more complex models should be overfitted at a certain point when sample size becomes very small. Conversely, the third-order Markov model is strongly overfitted when the sample sizes are small, performing significantly worse than the independence model until the sample size increases to more than 400-500 data points. From that on, however, the third-order Markov model clearly outperforms the independence model, which shows that statistical dependencies among adjacent sequence positions do exist in the splice site data set.

Using PMMs, we are capable of interpolating between both special cases, if the structure score yields reasonable models. BIC and fsNML scores perform well for large sample sizes, as they are superior to the third-order Markov model and thus also superior to the independence model.

There are differences for small sample sizes though, which could be expected by merely considering the model complexity for varying sample sizes, as displayed in Figure 3.2(a), and by the hypothesis that small models might be good for small sample sizes, as the comparison of the fixed structure models illustrates. For small sample sizes, BIC yields model structures that are only slightly more complex than the independence model, so they also get close to it in terms of predictive probability. However, with increasing sample size, BIC is still capable of capturing the important dependencies well, so it does not suffer from learning very sparse models. In comparison, fNML performs significantly worse for small sample sizes, and this method yields a similar prediction performance compared to BIC only for more than 500 data points.

AIC performs similarly to fNML in terms of prediction, unless the sample sizes are very small where AIC yields a smaller model and thus a better prediction. However, AIC is for all sample sizes inferior to BIC, which is not surprising as BIC is known to penalize models harsher than AIC.

We find that the Bayesian approaches are here inferior to the other three methods that are capable of structure learning, no matter how the ESS is chosen. Whereas $\eta = 10^{-1}$ yields a model complexity that is rather close to that of BIC, it nevertheless suffers from (i) the effect of rising model complexity for small sample sizes and (ii) unfavorable parameter estimates. An ESS value of $\eta = 10^3$ yields for most sample sizes (i) strongly overfitted model structures and (ii) parameter estimates that are close to a uniform distribution. A reasonable ESS value of $\eta = 10$ performs similar to fNML, yet it suffers from the problem of different ESS optima for structure and parameter learning as discussed in Section 3.3.1.

We summarize that, for this data set, the independence model is optimal if we have less than 100 data points available for estimating the distribution from. For sample sizes between 100 and 500, BIC yields the best tradeoff in finding a model structure that captures dependencies while avoiding overfitting, and even for very small sample sizes ($N < 100$), the difference in prediction compared to the independence model is rather small. For $N > 500$ most structure scores perform similar, even though the learned model complexities still differ to a large extent as observable in Figure 3.2(a). For large sample sizes, the negative effect of some dispensable parameter sets may be negligible, which is supported by the comparatively good performance of the third-order Markov model for $N > 2000$, even though it never catches up to the parsimonious models yet.

We learned in the previous section that the Bayesian methods are here hampered by the influence of the ESS on both structure and parameter learning. Especially regarding the cross comparison in Figure 3.1, we speculate that it is in fact impossible to find one ESS value that is optimal for both structure and parameter learning. In practice, this can be dealt with by modifying the structure prior hyperparameter κ, which then overshadows the ESS influence on the model complexity. However, it is intuitively neither clear which value κ to choose in order to obtain which model complexities, nor which combination of κ and η may be optimal for prediction. To this end, an internal CV on the training data can be used for determining an optimal prior choice as demonstrated in Chapter 2. However, this increases the computational effort dramatically and may limit the large-scale applicability of PMMs.

3.3.3. Comparing prediction performances

We speculate that a prediction framework using BIC or fNML structure score in combination with fsNML parameter estimate could represent a reasonable alternative to the Bayesian approach with hyperparameters optimized via CV. In order to test this hypothesis, we next perform a study on several practically relevant data sets. We focus on transcription factor binding sites from the publicly available database Jaspar [28]. We select all available data sets containing more than 100 sequences, since we observed in Figure 3.2(b) that below that sample size the independence model can be expected to yield optimal predictions even if strong dependencies exist in the data. We obtain 15 different data sets, which vary in sequence length from 8 to 21 and in sample size from 101 to 4311, covering the whole range of complexity currently known to appear in TFBS biology. For all data sets, we evaluate the prediction performance in a CV experiment.

For the Bayesian approach, we use an additional internal CV (leave-one-out CV for all data sets with $N < 1000$, 10-fold CV for the rest) for optimizing the hyperparameters. We use three different candidates (1,10,100) for the ESS and 20 different values for the structure prior hyperparameter κ, interpolating between minimal and maximal model.

The results are shown in Table 3.1. In most cases, all three methods (BIC-fsNML,

Table 3.1.: **Prediction on Jaspar TFBS data sets.** We show the performance of the three different methods using 20 different data sets. BDeu-MP uses an internal CV on training data for tuning structure prior hyperparameter κ and ESS η.

data set	L	N	BIC-fsNML	fNML-fsNML	BDeu-MP (CV)
EWSR-FLI	18	101	-0.32	-0.40	**-0.21**
HIF1A::ARNT	8	103	-4.51	-4.64	**-4.50**
NFYA	16	116	**-13.39**	-14.27	-13.55
Myc	10	227	**-6.25**	-6.29	-6.27
ESR2	18	357	**-15.46**	-15.62	-15.58
ESR1	20	475	**-18.40**	-18.52	-18.44
Zfx	14	481	**-10.37**	-10.41	-10.38
Stat3	10	613	-4.44	**-4.42**	-4.44
Sox2	15	669	-11.59	-11.67	**-11.58**
Foxa2	12	808	-7.35	-7.36	**-7.32**
PPARG::RXRA	15	864	-12.58	**-12.52**	-12.54
FOXA1	11	897	**-6.46**	-6.53	-6.54
CTCF	19	908	-13.65	-13.72	**-13.62**
GABPA	11	993	-5.94	-5.96	**-5.93**
Pou5f1	15	1356	-10.44	**-10.42**	-10.44
REST	21	1607	-12.60	-12.63	**-12.59**
STAT1	15	2085	**-11.97**	-11.99	-11.99
Esrrb	12	3661	-7.44	**-7.43**	-7.45
Tcfcp2I1	14	4079	-11.06	-11.02	**-11.01**
Klf4	10	4311	-5.07	**-5.07**	-5.07

fNML-fsNML, and BDeu-MP with double CV) are fairly similar, which implies that the hyperparameter-free methods are indeed as good as the Bayesian method that uses the exhaustive internal CV for hyperparameter-tuning.

One interesting example is NFYA, where BIC leads to a clearly increased prediction performance compared to fNML and where it also predicts better than the Bayesian approach. This can be explained by the fact that a small data set may require a rather simple model despite containing strong statistical dependencies. So NFYA is an example of a situation that we simulated in Section 3.3.2 by subsampling as shown in Figure 3.2(b). As previously observed, fNML and also BDeu may have difficulties to learn a simple model structure if there is not a sufficient amount of data available that supports such an extreme hypothesis.

Hence, BIC might be the best structure score if it is combined with fsNML parameter estimates and if the sample size is small ($N < 500$) in relation to the maximal model complexity and the total number of possible structures. However, it might be possible that BIC underfits when there are extremely strong and diverse statistical dependencies, requiring large model structures. If the optimal model complexity is above average or if sample size is comparatively large ($N > 500$), fNML might be the more robust choice.

These observations actually yield a vague prior knowledge about model complexity for analyzing further data sets, as we now expect comparatively sparse models to perform well.

However, since it is difficult to translate this vague knowledge into precise values for the structure prior hyperparameter κ, using BIC constitutes a reliable method of obtaining rather sparse model structures and thus expressing our recently gained vague prior belief.

3.3.4. Running time considerations

We observed that BIC/fNML-fsNML might be an alternative learning approach to Bayesian methods if there is no or only vague prior knowledge available, and the prior must be tuned by CV. Cross-validation multiplies the time complexity of the entire learning algorithm by KC, where K is the number of holdouts, and C is the number of different prior values to be tested. In Section 3.3.3, we used $K = 10$ for the large data sets ($N > 1000$), and $K = N - 1$ for the remaining ones.[2] The number of different prior choices C is the product of different ESS values η (3 in our studies) and different structure prior hyperparameters κ (20 in our studies). In practice, learning one third-order PMM from the Pou5f1 data set on a 2.5 GHz processor takes 1.8 seconds using BIC-fsNML and 1.6 seconds using fNML-fsNML. The running time is here dominated by the DP algorithm for finding the optimal model structures. Learning a similar model with the Bayesian approach using CV takes 1,036 seconds, which is indeed close to a factor of 600. For the other large data sets, the running time ratio is similar close to the expectation. For smaller data sets, this is even more unfavorable for the Bayesian method, since leave-one-out CV has to be used to obtain robust hyperparameter estimates.

The running time depends on the number K and C, but without a doubt compromises with respect to both values might reduce the difference between methods and yet not decrease the performance substantially. However, it entails the danger of obtaining unreliable estimates, which is probably worse than needlessly investing more time. Moreover, transforming the problem of choosing adequate hyperparameters into choosing appropriate sets of hyperparameters for the CV is essentially not avoiding user interference at all.

Bottom line

In this chapter, we investigated different methods for learning PMMs from fully observable data with the incentive of eliminating the necessity of hyperparameter-tuning, thereby contributing to main objective I (Figure 1.11). By empirical studies on benchmark data sets, we found that the factorized sequential NML estimate is a safe choice for obtaining probability parameters, as it always provides a certain parameter smoothing without dominating the data. For structure learning, BIC is a surprisingly good choice, especially when sample sizes are small or the expected optimal model complexity is below average. Since the Bayesian and to some extent also the fNML score are tailored towards recovering some true model

---

[2] The N-th sequence is used as test data point in the outer CV.

structure, they require a lot evidence for selecting a very simple model. For prediction of DNA binding sites, however, comparatively simple models, which take into account higher-order dependencies while having only a moderately increased parameter space in relation to the independence model, show the best empirical performance. Here, the approximation error of BIC for small sample sizes becomes an advantage, since this structure score yields sparse PCT structures when the sample size is small, which is exactly the desired behavior for learning PMMs. We conclude that for learning PMMs from fully observable data, the combination of BIC as structure score and fsNML as parameter estimate is the preferable learning method – at least for the application of modeling DNA binding sites.

There are things known and there are things unknown,
and in between are the doors of perception.

Aldous Huxley

4

Latent variable models

In the previous two chapters, we discussed a setting where a single distribution is to be inferred from a fully observable set of training sequences, which corresponds to the comparatively simple problem of binding site recognition as specified in Section 1.2.1. Often, however, the situation is more complex due to the presence of latent variables, which arise in several settings, two of which are relevant for this thesis.

In the first setting, the statistical pattern that represents a set of binding sites may not be fully explainable by a single probability distribution. Instead, it may be possible that those binding sites are highly heterogeneous, thus being described appropriately by a mixture of two or more distributions. In such a setting, latent variables are commonly used to model the – unobserved – indication of which distribution each of the data points is associated with, resulting in a *mixture model*, which we formally define in Section 4.1.

The second setting comprises models for motif discovery (Section 1.2.2) that assume sequences of arbitrary length to contain a recurring pattern. We call these models *promoter models*, even though the input data do not necessarily need to be promoters but can also be ChIP-positive fragments, for instance. Promoter models require latent variables for representing the unknown binding site configuration (position, strand orientation) in the input data. We formally define promoter models in Section 4.2.

Whereas both mixture models and promoter models are motivated differently and also have different applications, the algorithmic challenges arising from learning in the presence of latent variables are almost identical in both cases, and we discuss them in Section 4.3.

4.1. Mixture models for DNA binding sites

The input data consists of a set of aligned sequences of identical length, as it is typically the case for binding site recognition (Section 1.2.1). However, we assume those sequences to not follow one single common distribution, but being a mixture of at least two distributions (Figure 4.1), with latent variables indicating the distribution that each sequence follows.

The intuition of these mixture models is to capture complex statistical features rather by two or more comparatively simple distributions instead of trying to fit one very complex

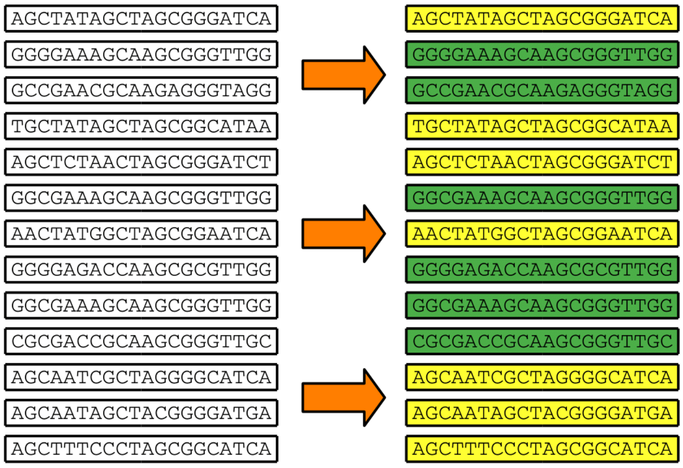


Figure 4.1: **Illustration of the rationale behind a two-component mixture model for DNA binding sites.** Each DNA sequence follows one out of two possible component distribution, but it assignment of a particular sequence to its component is unknown, and thus handled by a latent variable. A possible realization, which may be not obvious at first glance in the list of sequences on the left, is shown on the right using green and yellow color.

distribution to data. As such, using a mixture model is typically a choice that is made by the user and not inherent to the statistical problem that is to be solved. Mixture models typically perform well if the data consist of several distinct subpopulations of data points, and learning a mixture model can thus also be perceived as solving a clustering problem. One widespread example for continuous data is a mixture of Gaussian distributions [101]. In the case of DNA binding sites, mixture models can be motivated by the assumption that DNA-binding proteins may have different binding modes and may change their binding affinity substantially when their confirmation is modified by some biochemical process.

4.1.1. Variables and parameters

We consider the general case of a K-component mixture model for DNA sequence patterns, i.e., sequences of fixed width W . The assignment of each of the N sequences to one of the K components is specified by the latent variable vector $\vec{u} = (u_1, \ldots, u_N) \in \{1, \ldots, K\}^N$. Each component k of the mixture model may be an arbitrary statistical model parameterized by Θ_k. Both the model as well as the model class of different components may be different. We denote the complete set of parameters by $\mathbf{\Theta} = (\vec{\pi}, \Theta_1, \ldots, \Theta_K)$, where $\vec{\pi} = (\pi_1, \ldots, \pi_K)$, and π_k contains the probability of the k -th mixture component, which can either be specified by the user or be inferred from data.

4.1.2. Likelihood

The conditional likelihood of the mixture model, i.e., the probability of data given parameters with given realization of the latent variables, writes as

$$P(\mathbf{x}|\vec{u}, \Theta_1, \ldots, \Theta_K) = \prod_{k=1}^{K} P(\mathbf{x}_{\{\vec{u}=k\}}|\Theta_k) \tag{4.1}$$

with

$$\mathbf{x}_{\vec{u}=k} = (x_i)_{i|u_i=k} \tag{4.2}$$

denoting all sequences that are assigned to component k by the latent variables $\vec{u}$. The probability over the space of latent variables is given by

$$P(\vec{u}|\vec{\pi}) = \prod_{i=1}^{N} \pi_{u_i}, \tag{4.3}$$

which we next use to define the likelihood of a mixture model by

$$P(\mathbf{x}|\boldsymbol{\Theta}) = \sum_{\vec{u}} P(\mathbf{x}|\vec{u}, \boldsymbol{\Theta}) P(\vec{u}|\boldsymbol{\Theta}). \tag{4.4}$$

The definition of the mixture model is modular in the sense that for each component a different model or even a different model class can be used. The most simple example for modeling DNA binding sites is a mixture of two PWM models, but it is also possible to define a three-component mixture model where, e.g., the first component is a PMM, the second component is a Bayesian tree, and the third component is a PWM model.

4.1.3. Prior

In the Bayesian setting, we define a prior for the mixture model that factorizes as

$$P(\boldsymbol{\Theta}) = P(\vec{\pi}) \prod_{k=1}^{K} P(\Theta_k), \tag{4.5}$$

where $P(\Theta_k)$ denotes the prior distribution of the k -th component, and $P(\vec{\pi}) = \mathrm{Di}(\vec{\pi}|\vec{\beta})$ is a Dirichlet distribution with hyperparameters $\vec{\beta} = (\beta_1, \ldots, \beta_K)$. $P(\vec{\pi})$ is only needed if the distribution over the components is to be estimated. However, often this distribution is set externally, e.g., to a uniform distribution.

4.2. Promoter models

For the problem of motif discovery, the use of latent variables is not a choice anymore, but is inherent to the learning problem. Here, the input consists of a set of N non-aligned sequences of arbitrary length, i.e., $\mathbf{x} = (\vec{x}_1, \ldots, \vec{x}_N)$ with $\vec{x}_i = (x_{i1}, \ldots, x_{iL_i})$. The task is to infer at least one statistical pattern describing putative binding sites located at unknown positions in those sequences, and the latent variables indicate the locations and configuration of the binding sites. Whereas mixture models have many applications in all fields of science, the motif discovery problem is specific to DNA sequence analysis. Within that field, however, it is one of the key problems with a large amount of previous work as discussed in Section 1.2.2.

Promoter models may differ in the number of different patterns that may occur in the data set, the number of different binding sites per sequence, and the order and dependency structure among clusters of multiple binding sites. Complex approaches attempt to model entire cis-regulatory modules [38, 102]. However, complex models on the promoter level that allow more than one binding site per sequence also cause a substantially increased computational complexity.

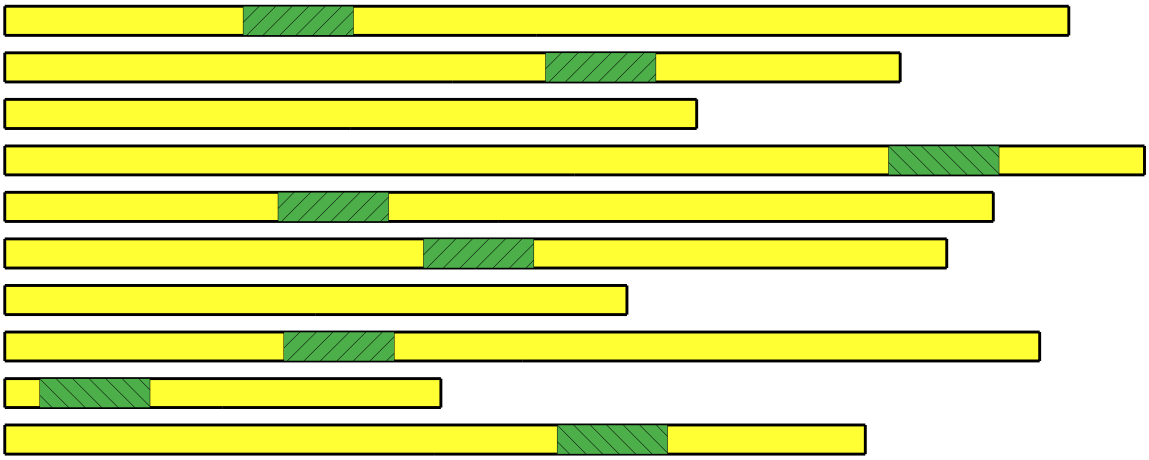


Figure 4.2.: **ZOOPS model assumptions**. Each DNA sequence in the data set of possibly different length (yellow) may contain one binding site of the same transcription factor (green), either on forward (45° hatching) or reverse complement ($-45°$ hatching) strand. Existence, location, and strand orientation of binding sites is typically unknown and thus handled by latent variables.

As we attempt in this thesis to increase complexity in a different aspect (motif model), we here use the simple *ZOOPS* promoter model, which assumes **z**ero or **o**ne **o**ccurrence of a binding site **p**er **s**equence [103]. Sometimes it is also referred to as NOOPS (noisy OOPS) model [47]. Although being far from biological reality, it is widespread [30, 41, 47, 40], since its simplicity offers several advantages including a linear time complexity with respect to input size and a high modularity since arbitrary component models can be used. While not following directly from the ZOOPS terminology, and hence not always be considered as part of a ZOOPS model, we also enforce two additional features, by (i) setting the number of motives that may occur in the data to one, and (ii) allowing a binding site to be located on the forward strand or on the reverse complementary strand. When referring to ZOOPS model in the following, we refer to this special case.

4.2.1. Variables and parameters

The ZOOPS model combines an arbitrary *motif model* of width W, parameterized by Θ_{m}, with an arbitrary *flanking model*, parameterized by Θ_{f}. Whereas the motif model can be a

PMM as defined in Chapter 2 or any of the models discussed in Section 1.3, the flanking model is typically a homogeneous Markov model (Markov chain).

The binary latent variable u_i handles the situation that the i-th sequence does ($u_i = 1$) or does not ($u_i = 0$) contain a binding site. We denote the probability of a binding site existing in an arbitrary sequence by ν, i.e., $P(u_i|\nu) = \nu^{\delta_{u_i,1}}(1-\nu)^{\delta_{u_i,0}}$ for $i \in (1,\ldots,N)$. We model the position of the binding site of width W by the latent variable $v_i \in \{1,\ldots,L_i - W + 1\}$ and assume binding sites to be uniformly distributed, i.e., $P(v_i) = \frac{1}{L_i - W + 1}$. Since the binding site may occur on both strands, we introduce a third latent variable $s_i \in \{0,1\}$, which indicates whether the binding site occurs on the forward strand ($s_i = 1$) or on the reverse complement strand ($s_i = 0$). We denote the probability of finding a binding site on the forward strand as σ , that is, $P(s_i|\sigma) = \sigma^{\delta_{s_i,1}}(1-\sigma)^{\delta_{s_i,0}}$ for $i \in (1,\ldots,N)$. We combine the latent variables of the complete data set by $\vec{u} = (u_1,\ldots,u_N)$, $\vec{v} = (v_1,\ldots,v_N)$, and $\vec{s} = (s_1,\ldots,s_N)$ and the parameters pertaining the distribution of the latent variables by $\Theta_\mathrm{c} = (\nu,\sigma)$.

In analogy to mixture models, the distributions over latent variables can be estimated from data, but it is also not completely unreasonable to set them externally. We obtain an important special case when setting $\nu = 1$, which results in an *OOPS* (**o**ne **o**ccurrence of a binding site **p**er **s**equence) model.

4.2.2. Likelihood

The conditional likelihood of a sequence x_i given latent variables and parameters is thus given by

$$\begin{aligned} P(\vec{x}_i|v_i, s_i, \Theta_\mathrm{m}, \Theta_\mathrm{f}) = {} & P_\mathrm{f}(\vec{x}_i|\Theta_\mathrm{f})^{\delta_{u_i,0}} \\ & \cdot P_\mathrm{f}(x_{i,1},\ldots,x_{i,v_i-1}|\Theta_\mathrm{f}) \\ & \cdot P_\mathrm{f}(x_{i,v_i+W},\ldots,x_{i,L_i}|\Theta_\mathrm{f})^{\delta_{u_i,1}} \\ & \cdot P_\mathrm{m}(x_{i,v_i},\ldots,x_{i,v_i+W-1}|\Theta_\mathrm{m})^{\delta_{u_i,1}\delta_{s_i,1}} \\ & \cdot P_\mathrm{m}(\mathrm{rc}(x_{i,v_i},\ldots,x_{i,v_i+W-1})|\Theta_\mathrm{m})^{\delta_{u_i,1}\delta_{s_i,0}}, \end{aligned} \tag{4.6}$$

where $\delta_{a,b}$ denotes the Kronecker delta, which returns 1 if $a = b$ and 0 otherwise. For the entire data set $\mathbf{x}$, we assume statistical independence among individual sequences, hence

$$P(\mathbf{x}|\vec{u}, \vec{v}, \vec{s}, \Theta_\mathrm{m}, \Theta_\mathrm{f}) = \prod_{i=1}^{N} P(\vec{x}_i|u_i, v_i, s_i, \Theta_\mathrm{m}, \Theta_\mathrm{f}) \tag{4.7}$$

The probability over the space of latent variables is given by

$$P(\vec{u}, \vec{v}, \vec{s}|\Theta_{\mathrm{c}}) = \prod_{i=1}^{N} P(u_i|\nu)P(v_i)P(s_i|\sigma). \tag{4.8}$$

For computing the (marginal) likelihood of sequence $\vec{x}_i$ given parameters $\mathbf{\Theta} = (\Theta_{\mathrm{m}}, \Theta_{\mathrm{f}}, \Theta_{\mathrm{c}})$, we sum over all latent variables, that is,

$$P(\vec{x}_i|\mathbf{\Theta}) = \sum_{u_i} \sum_{v_i} \sum_{s_i} P(\mathbf{x}|\vec{u}, \vec{v}, \vec{s}, \Theta_{\mathrm{m}}, \Theta_{\mathrm{f}})P(\vec{u}, \vec{v}, \vec{s}|\Theta_{\mathrm{c}}). \tag{4.9}$$

The marginal likelihood of the entire data set is given by

$$P(\mathbf{x}|\mathbf{\Theta}) = \sum_{\vec{u}} \sum_{\vec{v}} \sum_{\vec{s}} P(\mathbf{x}|\vec{u}, \vec{v}, \vec{s}, \Theta_{\mathrm{m}}, \Theta_{\mathrm{f}})P(\vec{u}, \vec{v}, \vec{s}|\Theta_{\mathrm{c}}). \tag{4.10}$$

4.2.3. Alternative definition of latent variables

While the separate definition of latent variables for binding site occurrences u_i, binding site position v_i, and strand orientation s_i is intuitive and easily extendable, it has the disadvantage of v_i and s_i being meaningless when $u_i = 0$. An equivalent representation of binding site occurrence uses a single latent variable $h_i \in (0, \ldots, 2(L_i - W + 1))$ for the i sequence with

$$h_i = u_i(2v_i - s_i), \tag{4.11}$$

from which follows that

$$u_i = \delta_{h_i,0} \tag{4.12}$$

$$v_i = \left\lfloor \frac{h_i}{2} \right\rfloor \tag{4.13}$$

$$s_i = h_i \bmod 2. \tag{4.14}$$

Both of the equivalent definitions of latent variables of a ZOOPS model have their merits, and which of them is more suitable may depend on the task at hand. The representation using triplets (u_i, v_i, s_i) might be more intuitive at first glance, whereas the combined definition is closer to practical implementation.

4.3. Learning in the presence of latent variables

In this section, we discuss standard approaches of learning probability parameters and model structures within latent variable models. In comparison to fully observable data, learning in the presence of latent variables is often substantially more difficult.

4.3.1. Parameter Learning

Due to the unknown realization of the latent variables, the full likelihood of the model cannot be computed efficiently, and thus no closed-form expressions for parameter estimation according to principles such as maximum likelihood (ML) or maximum a posteriori (MAP) can be derived. However, optimizing the parameters in the presence of latent variable by numerical optimization is a common task that can be accomplished by the *Expectation-Maximization* (EM) algorithm [33].

The EM algorithm is an iterative algorithm that alternates between an expectation (E) step and a maximization (M) step, monotonically increasing the likelihood. In the E step, the algorithm computes the expected log likelihood given the current model parameters. In the M step, it computes new model parameters by maximizing the expected log likelihood resulting from the E step.

For a continuous parameter space, the EM algorithm has been shown to converge to a local maximum or saddle point of the target function [33]. While originally derived for maximum likelihood estimation, the EM algorithm can be easily generalized to maximize the posterior of a model [104]. Since target functions in practically relevant applications have several local maxima, multiple restarts of the algorithm with different initialization are required to increase the probability of finding the global maximum.

4.3.2. Structure learning

Whereas parameter learning in the presence of latent variables is a well-studied field of research, structure learning in the presence of latent variables is less explored.

For Bayesian networks, there are solutions to the structure learning problem for *missing data*, i.e., missing entries in the data matrix, which is a simplified form of the general latent variable problem, which typically involves variables that are not only missing from the experiment but can not be observed in principle. Here, a BIC score or the Bayesian marginal likelihood can be optimized using the *structural EM algorithm* [105] and the *Bayesian structural EM algorithm* [106]. However, these algorithms do not apply to the general case where the data set is partitioned into subsets according to the state of latent variables and different models structures have to be estimated based on that partitioning.

For this general case, following a standard EM derivation by allowing a variable structure as additional, discrete parameter ξ, yields an M step that searches the best pair (ξ, θ_ξ) of structure and corresponding probability parameters in order to find the latent variable model with the highest target function. In case that target function is the likelihood, this results in model selection via the ML principle within the M step of the algorithm. However, model selection according to the ML principle always yields the largest model structure with the highest dimensionality of the parameter space – at least when the model class is nested. In the Bayesian setting, there may be a prior, resulting in model selection according to the MAP principle. However, asymptotically the prior is dominated by the data, so model selection according to the MAP principle is asymptotically equivalent to model selection according to the ML principle. Hence, a structure score based on the MAP principle is not a consistent estimator.

For these reasons, structure learning within an EM algorithm has mostly been used for models with variable structure but fixed number of parameters. One example are Bayesian trees (BTs), where the tree structure is unknown, all BTs have the same number of parameters. For these models, learning in the presence of latent variables via the EM algorithm is doable both according to the ML and to the MAP principle [104]. The generalization of this approach to Bayesian networks is straightforward under the condition that all network structures under consideration have the same number of parameters, which can be achieved by using the class of BN(d) for any fixed value of d as discussed in Section 1.3.3.

For mixtures of PMMs, an EM algorithm for finding structure and model parameters that maximizes the posterior based on an adaptation of the method for mixtures of trees [104] has been proposed [107]. However, this algorithm is extremely sensitive to the selection of hyperparameters, especially the choice of the structure prior is crucial. Since the structure score that results from the modified EM algorithm is not consistent, the structure prior needs to counterbalance growing sample size. For practical relevant data sets, a uniform structure prior ($\kappa = 1$) generally yields the largest possible PCT structures, as data dominates the prior and model selection is essentially conducted according to ML. As a consequence, smaller values for κ are required to obtain non-maximal PCTs, but it is unknown which values yield reasonable model complexities. Hence, hyperparameter-tuning is – in contrast to the Bayesian model selection on fully observable data – not only beneficial to improve performance for small data, but essential for conducting any reasonable model selection.

4.3.3. EM algorithm for motif discovery using PMMs

The EM algorithm for mixtures models with PMM components [107] can be adapted to the problem of motif discovery.

In order to learn a PMM from a set of promoter sequences, we apply the maximum a posteriori principle to the ZOOPS model (Section 4.2). We intend to solve the following optimization problem

$$\begin{aligned}\hat{\Theta} &= \operatorname*{argmax}_{\Theta} P(\Theta|\mathbf{x}) \\ &= \operatorname*{argmax}_{\Theta} \frac{P(\mathbf{x}|\Theta)P(\Theta)}{P(\mathbf{x})} \\ &= \operatorname*{argmax}_{\Theta} \log P(\mathbf{x}|\Theta) + \log P(\Theta),\end{aligned}$$

which cannot be solved analytically due to the presence of latent variables $\vec{v}$ (existence a binding sites), $\vec{u}$ (positions of binding sites) and $\vec{s}$ (strand orientation of binding sites). However, the maximum of the posterior density of the model can be approximated by a modified EM algorithm [33]. This EM algorithm updates iteratively weighted estimates $\gamma^{(t)}$ for the latent variables based on parameters $\Theta^{(t)}$ (E step) and parameters $\Theta^{(t+1)}$ based on weighted estimates of latent variables $\gamma^{(t)}$ (M step). Derivation of E step and M step of follows standard techniques and can be found in Appendix Section B.3.

The modified EM algorithm only guarantees to monotonically increase the posterior value in each iteration step, so it can miss the global optimum since the posterior landscape is often not convex, which is often the case in practice. In order to cope with this problem, it is necessary to start the algorithm multiple times and use the result with the largest posterior. We found ten restarts of the EM algorithm, with each EM instance continued until the log-posterior difference between two subsequent iterations is smaller than 10^{-6}, to be a reasonable compromise between invested running time and stability of the obtained results for many practical motif discovery applications.

Bottom line

In this chapter, we introduced latent variable models that are relevant for the remaining topics of this thesis, namely applying PMMs to mixture models and promoter models for motif discovery, with the ZOOPS model being the most relevant instantiation of the latter. We discussed that parameter learning can here not be carried out analytically, so iterative algorithms, such as the EM algorithm, have to be resorted to. Structure learning in the presence of latent variables is a comparatively unexplored field with theoretically sound solutions available only for some special cases, such as mixtures of Bayesian trees or learning a general Bayesian network structure for partially missing data. For mixtures of PMMs, a learning approach that applies the MAP principle to both structure and parameter learning has been previously proposed, which yields, however, inconsistent structure learning that is asymptotically equivalent to model selection according to maximum likelihood. Despite this undesirable theoretical properties, the EM algorithm is a first possibility to learn mixture models with PMM components, and we discussed that the same approach can be also applied to the ZOOPS model for the task of motif discovery. In the following chapter, we use this algorithm to study the applicability of PMMs to the problem of motif discovery by focusing on one particular DNA-binding protein, studying the nature of putative intra-motif dependencies and their impact on motif discovery performance.

Empirical explorations ultimately change our understanding of which questions are important and fruitful and which are not.

Lawrence M. Krauss

5

Dependencies within CTCF binding sites

In Chapter 2, we observed that PMMs might be helpful for inferring a distribution from a small set of given transcription factor binding site for the task of prediction. In this chapter, we now apply PMMs to the problem of motif discovery from ChIP-seq data. Hereby, we focus on the empirical results, attempting to answer two related but yet different questions for one single exemplary transcription factor. From algorithmic point of view, we investigate to which degree taking into account intra-motif dependencies by using a PMM as motif model improves motif discovery. From biological point of view, we analyze the sequence motif predicted by a PMM, in order to identify features that may be more complex than a simple PWM can possibly convey.

5.1. Insulator-binding protein CTCF

For studying these questions, we focus on the CCCTC binding protein, more commonly known as *CTCF* [108], which plays a key role in many cellular processes.

5.1.1. CTCF biology

Most transcription factors are capable of enhancing or silencing gene expression by specifically binding to DNA elements that are called enhancers or silencers. In addition, there are DNA elements which may block the interaction of enhancers and promoters, providing another molecular switch in gene regulation. Those elements are called *insulators*, and the corresponding proteins are *insulator binding proteins* [109]. In addition, insulators can also act as chromatin barriers [109], preventing the spread of heterochromatin when being bound by their corresponding insulator binding protein. CTCF, which belongs to the class of zinc finger proteins, is the most important insulator binding protein in humans [110].

High-throughput methods, including ChIP-chip and ChIP-seq, have been applied to identify the location of CTCF binding sites in a variety of cell types and found that the number of binding sites in mammalian genomes is in the order of tens of thousands [111, 112, 113]. The binding site of CTCF is thought to comprise approximately 20 base pairs [112], which is large relative to many of the best-studied vertebrate transcription factor binding sites. However, only a few positions in CTCF binding sites show strong conservation between sites, and many CTCF binding sites that have been repeatedly identified show much divergence from the consensus sequence, suggesting that additional modes of binding might exist [114]. For these reasons, CTCF is an ideal candidate for studying putative intra-motif dependencies.

5.1.2. Data sets

For all experiments in this chapter, we use ChIP-seq data of CTCF from the ENCODE project [115] for different cell lines, available via the UCSC table browser [1]. If not specified otherwise, we use the human embryonic stem cells data (H1-hESC) for exemplifying the method and for studying properties of CTCF binding sites.

The data is already preprocessed, i.e., the steps of mapping the ChIP-seq reads to the genome using MAQ [20] and peak calling via F-seq [24] have already been performed. We obtain a file in UCSC narrowPeak format, which contains a list of genome coordinates (chromosome, start position, end position) and corresponding scores. Despite there are between 59,000 to 90,000 potential sequences, many of them seem comprise only few base pairs and often having large p-values. We discard all coordinates with a p-value greater than 10^{-16}, i.e., we only keep coordinates with the minimal p-value. For H1-hESC, we retain 3,264 ChIP-seq positive sequences with lengths ranging from 189 bp to 888 bp.

For performing classification experiments, we also need a set of putative sequences not being bound by CTCF. In order to keep general properties (such as GC-content of the DNA) similar in both data sets, we construct a negative data set by the following procedure. For each sequence in the positive data set, we extract its adjacent sequences of the same length from the human genome and add it to the negative data set. Formally written, for each positive sequence with coordinates (i, j), we add the sequences with the coordinates $(2i - j - 1, i - 1)$ and $(j + 1, 2j - i + 1)$ to the negative data set. In a final filtering step, we discard – if necessary – all negative sequences that partially overlap with positives to obtain two disjoint data sets.

5.2. Classification experiment

It is not entirely obvious how to evaluate different models for motif discovery, as there is typically no ground truth with respect to true binding sites available. One approach is to construct an artificial data set, where binding sites from databases are inserted into real or artificial promoter sequences. While this approach has the advantage that we obtain a ground truth w.r.t. binding site locations and could thus directly measure the effectiveness

---

[1] `http://genome.ucsc.edu/cgi-bin/hgTables?org=Human`

Table 5.1.: **ChIP-seq data sets for H1-hESC.** We show the number of sequences in each subset of the input data and the corresponding labels. The ChIP-seq positive sequences in H^+ are split at a ratio of 2:1 into training and test data.

	positives	negatives	total
training	$\mathrm{H}^+_{\mathrm{train}}(2,176)$	$\mathrm{H}^-_{\mathrm{train}}(4,352)$	$\mathrm{H}_{\mathrm{train}}(6,528)$
test	$\mathrm{H}^+_{\mathrm{test}}(1,088)$	$\mathrm{H}^-_{\mathrm{test}}(2,176)$	$\mathrm{H}_{\mathrm{test}}(3,264)$
total	$\mathrm{H}^+(3,264)$	$\mathrm{H}^-(6,528)$	

of different models in finding the inserted binding sites, such an artificial or semi-artificial data set is not satisfying. Since binding sites found in databases have been often obtained using a prediction tool such as MEME [31], which may use a particular statistical model such as the PWM model, the results are likely to be biased towards such a simple model and thus not being generalizable to real world data.

5.2.1. Fragment-based classification

We thus propose an evaluation solely based on ChIP-seq positive and negative data, where the classification problem is to classify long sequence fragments into those that contain an instance of the motif, and are thus ChIP-seq positives, and those that do not, thus being ChIP-seq negatives. We call this indirect approach of evaluating different motif models *fragment-based classification*, and discuss the detailed procedure and the rationale behind it in the next two paragraphs.

After dividing positive and negative sequences into training and test data sets at a ratio of 2:1 (Table 5.1), we train the parameters Θ_{f} of a homogeneous Markov model on $\mathrm{H}_{\mathrm{train}}$, which is the union of both positive and negative training data set, and we denote this model as *background model*. Next, we utilize $\mathrm{H}^+_{\mathrm{train}}$ for training a ZOOPS model (Section 4.2), using a PMM with structure hyperparameter κ as motif model. We use a homogeneous Markov model with parameters Θ_{f} as flanking model, and we refer to the complete ZOOPS model as *foreground model*. Hence, Θ_{f} serves (i) as parameter of the background model and (ii) as parameter of the submodel for the flanking sequences within the foreground model. We estimate the parameters Θ_{m} and Θ_{c} of the ZOOPS model, here by using the EM algorithm described in Section 4.3.3. Subsequently, we classify all test sequences utilizing the foreground model and the background model. Utilizing the true class labels, we can compute different measures of accuracy, such as the sensitivity for a fixed specificity of 99%.

The rationale behind this classification approach is the following. Since the model and model parameters for the negative sequences is identical to the flanking model for the positives, which we achieve by estimating its parameters Θ_{f} from the union of positive and negative training data, the only difference between the foreground model and the background model is the capability of the former to include a binding site in a sequence. If positive test sequences are predominantly classified to be generated by the foreground model, it can be caused only by the existence of at least one binding site that fits well to the motif. Comparing different motif models via this experiment, we may conclude that the model that yields highest classification performance contains the most realistic motif model among the investigated candidates.

5.2.2. Finding optimal model complexity

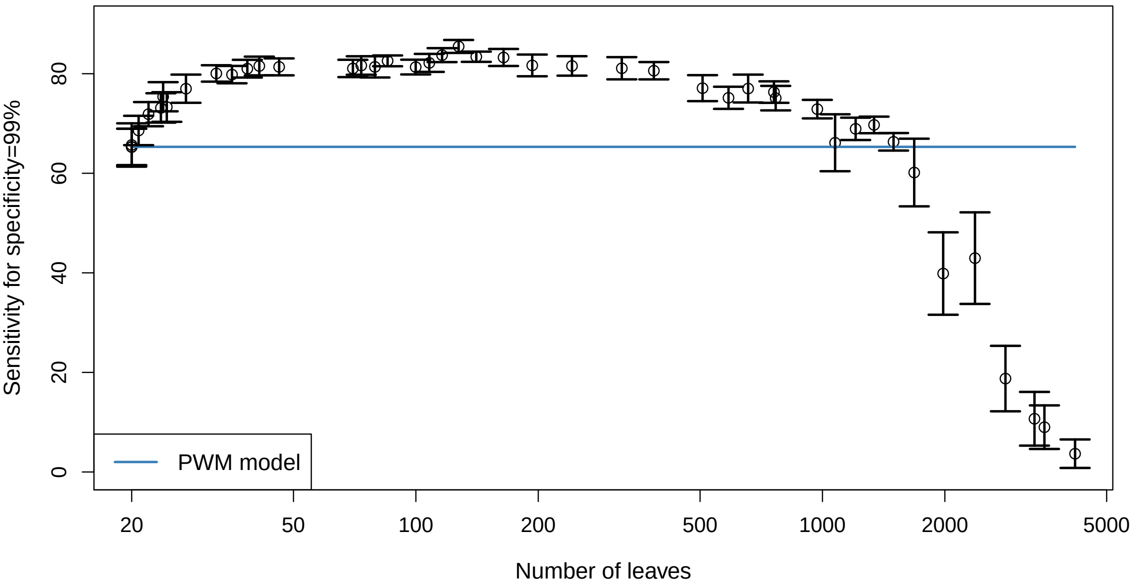


Figure 5.1.: **Cross-validation classification results on the H1-hESC training data set.** We show the averaged results of a 10-fold cross-validation experiment on $\mathrm{H}_{\mathrm{train}}$. For each κ, we plot the sensitivity (for a specificity of 99%) against the number of leaves. Error bars show double standard error.

We discussed in Section 4.3.2 that structure learning in the presence of latent variables using a modified EM algorithm requires – when the dimensionality of the parameter space is variable – tuning of the structure prior, which for PMMs reduces to tuning of the structure prior hyperparameter κ. In order to determine which value of κ yields an optimal model complexity, i.e., an optimal tradeoff between modeling dependencies and avoiding of overfitting, we perform in a first study a 10-fold cross-validation of the aforementioned classification experiment on the training data set $\mathrm{H}_{\mathrm{train}}$. We use a PMM of $W = 20$ and $d = 4$, and vary

κ to obtain models of different complexity.

For each value of κ we measure the classification performance by the sensitivity for a fixed specificity of 99% and average it over the ten cross-validation iterations. For visualizing the results, we plot the sensitivity against the average number of leaves that we obtain with a particular choice of κ (Figure 5.1).

Twenty leaves – one at each position in the motif model – corresponds to a PWM model. It yields an average sensitivity of 65.6% with a standard error of 4.3%. With increasing model complexity, we observe an steep increase in sensitivity until an average complexity of 40 leaves. With further increasing complexity, the sensitivity varies only slightly, indicating that models on the one hand do not yield substantial improvements, but on the other hand do not cause overfitting yet. This changes when the model has approximately 500 leaves where we observe a slightly decreased sensitivity compared to less complex models of 40-400 leaves. Nevertheless, the sensitivity is still higher than that of the PWM model, indicating that taking into account complex dependencies still outweights overfitting effects. This finally changes when the model complexity exceeds 1000 leaves, as the corresponding models perform worse than a simple PWM model, which is in agreement with the expectation that complex models are prone to overfitting.

Among all values of κ that have been used to interpolate between PWM model and full-order Markov model, we now pick the optimal value. The PMM with $\kappa = e^{-4.5}$ contains on average 127.4 leaves and yields the highest average sensitivity (85.5% for a fixed specificity of 99%). Hence, this κ yields – on average – the best tradeoff between capturing meaningful dependencies and avoiding overfitting effects. In the following, we denote a parsimonious Markov model trained with $\kappa = e^{-4.5}$ as *optimal PMM* for the H1-hESC training data set. Interestingly, the sensitivity of $\kappa = e^{-4.5}$ shows a standard error of only 1.3%, which is less than a third of the standard error of the PWM model. Hence, the optimal PMM yields an improved average motif discovery capability and the results are also more stable.

5.2.3. Test on independent data and comparison with alternative models

In a second study, we investigate how the optimal PMM classifies independent test data. Now, we utilize all sequences in $\mathrm{H}_{\mathrm{train}}$ for training the models (optimal PMM and PWM model) and $\mathrm{H}_{\mathrm{test}}$ for evaluating the classification performance. We expect the classification results to differ from the cross-validation experiments, yet we still observe a dramatic improvement. The optimal PMM yields a sensitivity of 85.2%, whereas the PWM model yields a sensitivity of only 71.1% (Figure 5.2).

In addition, we test alternative models that also take into account intra-motif dependencies and that can be easily incorporated into the used EM algorithm for motif discovery. The weight array model (WAM) [57], which takes into account nearest-neighbor dependencies only, achieves a sensitivity of 82.3%. We also test a first-order permuted Markov model

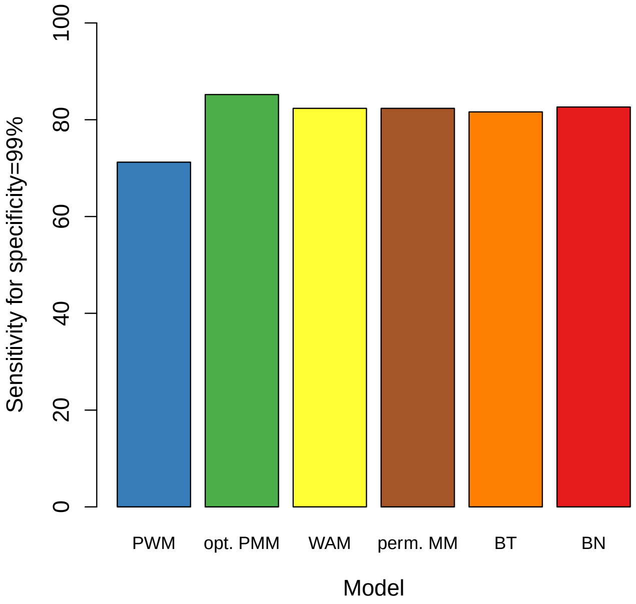


Figure 5.2: **Classification results on the H1-hESC test data set.** We display the sensitivity (for specificity of 99%) on the independent test data set H_{test}. Here we pick the optimal model from the cross-validation experiment (green) and compare it with the PWM model (blue). In addition, we display the sensitivity of alternative models that take into account statistical dependencies.

[76], but it turns out that the optimal permutation is the actual sequential ordering of the random variables as they appear in the sequence, so it yields exactly the same sensitivity as the WAM. A Bayesian tree [61], which is also limited to first-order dependencies but allows dependencies among non-adjacent positions achieves a sensitivity of only 81.6%. This shows that dependencies among adjacent positions are dominant as the additional flexibility of selecting an appropriate Bayesian tree even leads to a decreased classification of independent test data and the structure learning of the permuted Markov models yields essentially a WAM. A strictly second-order Bayesian network, i.e., a BN(2) [61] achieves a sensitivity of 82.6%. This is slightly better compared to the WAM, but the improvement is only small given the much higher complexity of the model class, which also involves finding the optimal BN structure. These results demonstrate that (i) modeling statistical dependencies among adjacent nucleotides in the binding sites improves de novo discovery of the CTCF binding motif and that (ii) PMMs might be a promising alternative to other models that also take into account intra-motif dependencies.

5.2.4. Different cell lines

In a third study, we further validate this result. To this end, we repeat the same analysis for ChIP-seq data from different cell lines in order verify that the H1-hESC data set is a reasonable representative for all cell lines. The data processing is in all cases identical to that of the H1-hESC data and we also apply the identical procedure of model selection via cross-validation (Table 5.2) and subsequent test on independent data. The final results are shown in Figure 5.3 and confirm the findings from the study on the H1-hESC cell line. The achieved sensitivities vary from cell line to cell line, and so does the optimal κ (Table 5.2). This is not surprising, since the size of the data sets also varies to a great extent, and larger data sets generally require a stronger prior for obtaining a certain model complexity. However, the optimal PMM always yields an improvement in sensitivity compared to the PWM model, stating that taking into account dependencies among adjacent nucleotides instead of neglecting them improves de novo discovery of the CTCF motif in all cell lines.

Table 5.2.: **Overview of different cell lines**. The table shows training sample size, estimated value of κ, and average estimated model complexity during cross-validation.

Cell line	N	$\ln(\hat{\kappa})$	avg #leaves
GM12878	7,068	-10.0	118.8
H1-hESC	2,176	-4.5	127.4
HeLa-S3	6,646	-9.5	119.0
HepG2	1,574	-5.5	96.4
HUVEC	8,162	-7.5	155.2
K562	8,142	-9.5	136.0
MCF7	3,082	-10.0	74.0
NHEK	5,076	-7.5	124.2
ProgFib	4,220	-8.0	105.2

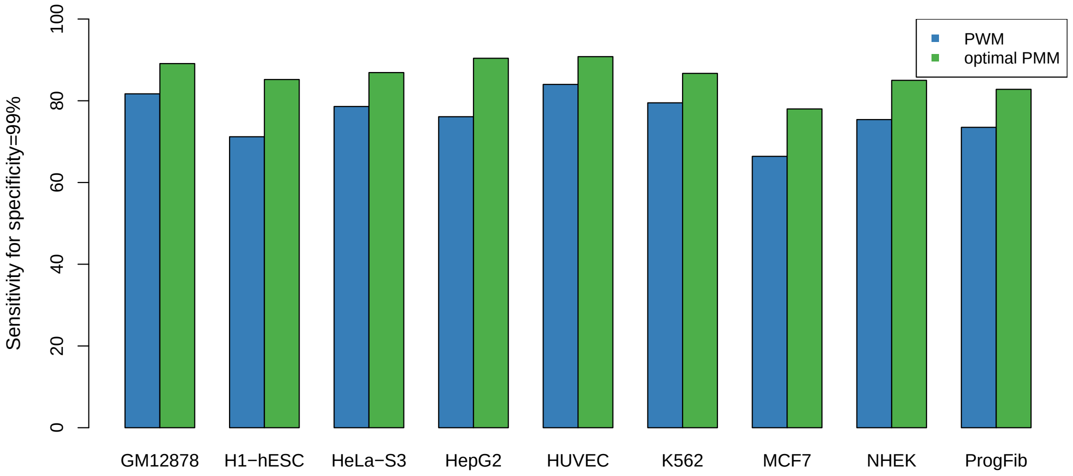


Figure 5.3.: **Classification results for different cell lines.** We show the sensitivity (for specificity of 99%) on data sets for nine different cell lines in analogy to Figure 5.2.

We repeat the experiment with the area under the ROC curve as performance measure and find that the results are qualitatively similar (Appendix Figure C.1). We also repeated the same studies with negative data sampled from the whole genome with the same length distribution as the positive peaks. The results are shown in Appendix Figure C.2(a) and Appendix Figure C.2(b). The classification performance generally increases, but the relative improvement gained by taking into account intra-motif dependencies remains qualitatively identical.

5.3. Motif analysis

In the previous section, we observed that taking into account statistical dependencies among adjacent nucleotides in the binding site yields a more accurate motif discovery and thus a more accurate sequence motif. Next, we use this motif model for binding site prediction in the H1-hESC data set. Utilizing the optimal PMM trained on $\mathrm{H}_{\text{train}}$, we predict binding sites in H^{+} by a threshold-based approach.

We use the learned parameters $\hat{\boldsymbol{\Theta}}$ of the ZOOPS model and compute the likelihood

$$\begin{aligned} P(\vec{x}_i, u_i = 1, v_i = \ell | \hat{\boldsymbol{\Theta}}) = & P(\vec{x}_i, u_i = 1, v_i = j, s_i = 1 | \hat{\boldsymbol{\Theta}}) \\ & + P(\vec{x}_i, u_i = 1, v_i = j, s_i = 0 | \hat{\boldsymbol{\Theta}}) \end{aligned} \tag{5.1}$$

for each possible start position j in each sequence $\vec{x}_i$ in the negative data set $\mathrm{H}^{-}_{\text{train}}$. We thus obtain a list of likelihood values, compute the empirical probability distribution of this list, and determine a threshold T as the likelihood corresponding to the lowest (10^{-4})-quantile. Using this threshold, we predict all subsequences of length W in each sequence $\vec{x}_i$ beginning at position j in the positive data set $\mathrm{H}^{+}_{\text{train}}$ satisfying

$$P(\vec{x}_i, u_i = 1, v_i = j | \hat{\boldsymbol{\Theta}}) > T \tag{5.2}$$

as binding sites. As the predictions of each position of a sequence are made independently, the ZOOPS assumption pertains only to the training algorithm, but for given model parameters $\hat{\boldsymbol{\Theta}}$ this method is capable of predicting multiple binding sites per sequence.

Using a significance level that corresponds to finding a false positive prediction every 10^4 nucleotides in control data set H^{-}, we predict 3,451 binding sites.

5.3.1. Sequence logo

The sequence logo corresponding to these binding sites is shown in Figure 5.4a. We find several positions that are dominated by a single nucleotide. In the context of motif analysis, these are often called *conserved* nucleotides, which is unrelated to the concept of evolutionary

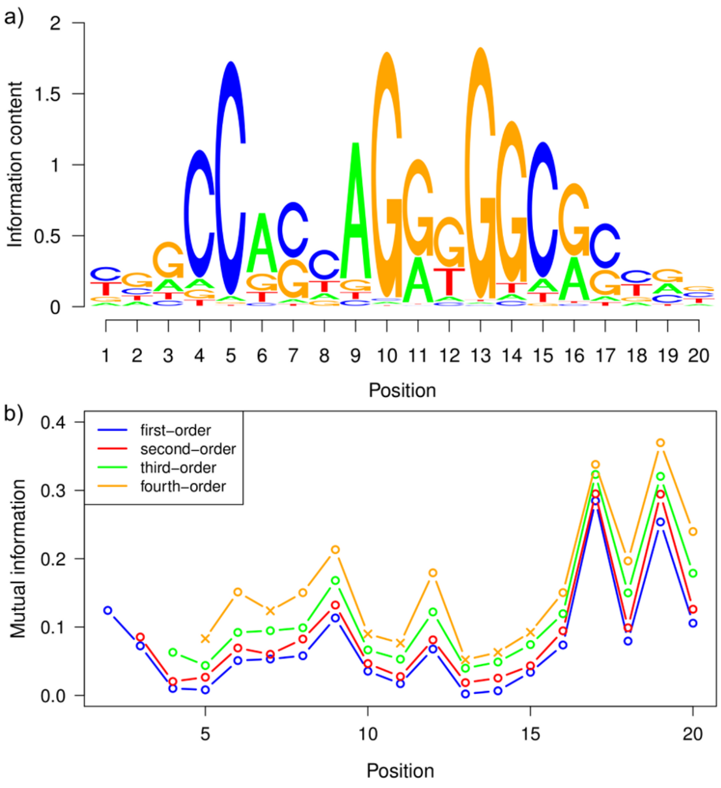


Figure 5.4: **Sequence logo and mutual information.** Figure a) depicts the sequence logo of CTCF binding sites predicted by the optimal PMM model. We find a high similarity to the previously known CTCF sequence logo [112]. Figure b) depicts the MI of different order between adjacent positions. MI values with a p-value above 0.05 are considered to be insignificant and displayed by the symbol $\times$. All MI values of first, second, and third order are significant, and the MI values of fourth order show significance only at some positions. We find that the amount of statistical dependencies varies within the motif to a great extent.

conservation. Especially at both ends of the motif, the nucleotides are *unconserved*, i.e., there is no dominating nucleotide at positions 1-3 and 16-20. Comparing the sequence logo with a prediction based on a PWM model and the same significance level, which yields 3,123 binding sites only, we observe a high similarity of both sequence logos, resembling a previously identified CTCF sequence logo [112]. Despite the fact that the majority of binding sites in each set is not contained in the other one, the position-wise nucleotide frequencies, which are the statistics visualized by a sequence logo, of both sets are almost identical. However, a sequence logo may be insufficient for fully characterizing a set of binding sites. Being a visualization of a PWM, a sequence logo is not capable of representing statistical dependencies.

5.3.2. Mutual information

Thus, we compute the mutual information (MI) between adjacent positions, which is a standard measure for quantifying statistical dependencies. We use a slightly extended definition by computing the mutual information $I(X_i, Y_i^{(d)})$, where X_i is the random variable of the nucleotide at position i of the motif and $Y_i^{(d)} = (X_{i-d}, \dots, X_{i-1})$. Hence $I(X_i, Y_i^{(d)})$, which can assume values between 0 and 2 bits, is the MI between the i-th symbol in the motif and the preceding d-mer. The MI for different orders d is shown in Figure 5.4b. It ranges

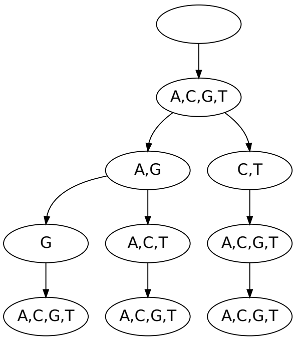


Figure 5.5: **PCT at position 13.** This PCT is an example of a small tree from a position in the motif that shows little statistical dependencies. We observe that the first layer of the tree is completely fused, which means that information about the nucleotide of position 12 is neglected.

from 0.001 bit (first-order MI at position 13) to 0.37 bit (fourth-order MI at position 19). In addition, we calculated the p-value of each MI value based on the fact that $2NI\ln 2$ is χ^2-distributed with $(|\mathcal{A}|^d - 1)(|\mathcal{A}| - 1)$ degrees of freedom.

The MI at any given position monotonically increases with increasing order. However, high-order MIs can become insignificant. We observe significant MIs of first, second, and third order for all positions in the motif. Considering MIs of fourth order, we find the MI at some positions to be insignificant. This is in agreement with the fact that the maximal order of the underlying PMM, which has been used for the prediction of the binding site studied here, is four, and that each position has its own parsimonious context tree, which may – in some cases – neglect fourth-order dependencies completely.

Comparing the MIs with the sequence logo (Figure 5.4), we find high MIs at positions that are relatively unconserved. We observe particularly high MIs at positions 17 and 19, indicating the presence of strong statistical dependencies to the preceding nucleotides. Conversely, the MI is generally low at positions that contain highly conserved nucleotides, such as position 5, 10, and 13. This can be explained by the fact that there is only little room for additional information at highly conserved positions. An extreme example is an absolutely conserved position for which preceding nucleotides can not contribute any additional information.

5.3.3. Optimal PCTs

After having quantified the statistical dependencies within the CTCF binding sites, we next investigate them qualitatively. The learning algorithm yields a set of PCTs that maximize the posterior of the PMM under a given prior. The PCTs differ at each position in the model not only in structure, but also in complexity, which is measured by the number of leaves. We observe a total number of 132 leaves for the entire motif, which equals 6.6 leaves per PCT on average. The smallest tree is the tree at the first motif position. It has only one leaf, since there are no predecessors in the sequence. We do not observe other positions with a single leaf, so each position takes into account its predecessors to some extent. All other PCTs have at least three leaves. One example is shown in Figure 5.5.

A representative of intermediate complexity with 7 leaves is the tree at position 17 (Figure 5.6a). The largest tree, which has 11 leaves, is located at position 19 (Figure 5.7a).

5.3.4. PCTs and MI values

The MI of different orders at a specific position and the structure of the corresponding PCT are not unrelated. Considering position 13 for instance, we find that the nodes on the first layer of the PCT are completely fused, i.e., there is only one node, representing all nucleotides. The first-order MI is almost zero, while the second- and third-order MIs are significant and the nodes on the second and third layer of the PCT show a certain diversity. On the fourth layer of the PCT, all nodes are completely fused, which is in accordance to the non-significant MI of order four. Further interesting positions are 17 and 19, which yield the two highest MIs among all twenty positions in the motif. However, they differ in one important aspect: For position 17, the first-order MI is already very high, and using higher-order MIs leads to an only small increase of MI. In contrast, position 19 yields a substantially lower first-order MI, but a much larger increase for longer contexts. The fourth-order MI of position 19 is finally above the corresponding MI of position 17. The ratio of fourth-order and first-order MIs differs considerably between both positions. This is reflected by the corresponding PCTs. The tree at position 19 contains 11 leaves, dominated by the subtree of first layer node `C`, whereas the tree at position 17 contains only 7 leaves with a comparatively low number of splits below the first layer. For position 19 on the one hand, the higher-order context plays an important role if the nucleotide at position 18 is either `C` or `T`, which are the two dominant nucleotides at this position. For position 17 on the other hand, it does not seem to be of importance which nucleotides are observed at position 13-15 if the nucleotide at position 16 is known.

5.3.5. Conditional sequence logos

Considering these findings, we further investigate the nature of statistical dependencies found in the binding sites of CTCF. We focus here on positions 17 and 19, since they show the highest mutual information. We compute the conditional relative nucleotide frequencies in the set of predicted binding sites given all possible contexts of the PCT at this position.

Since there is no established method of visualizing conditional probability distribution of a (parsimonious) context trees to date, we propose a visualization of these conditional nucleotide frequencies in a way that resembles sequence logos [51], dubbing it thus *conditional sequence logo* (CSL). A traditional sequence logo depicts the position-wise nucleotide frequencies along a sequence, whereas a CSL considers only one fixed position in the sequence

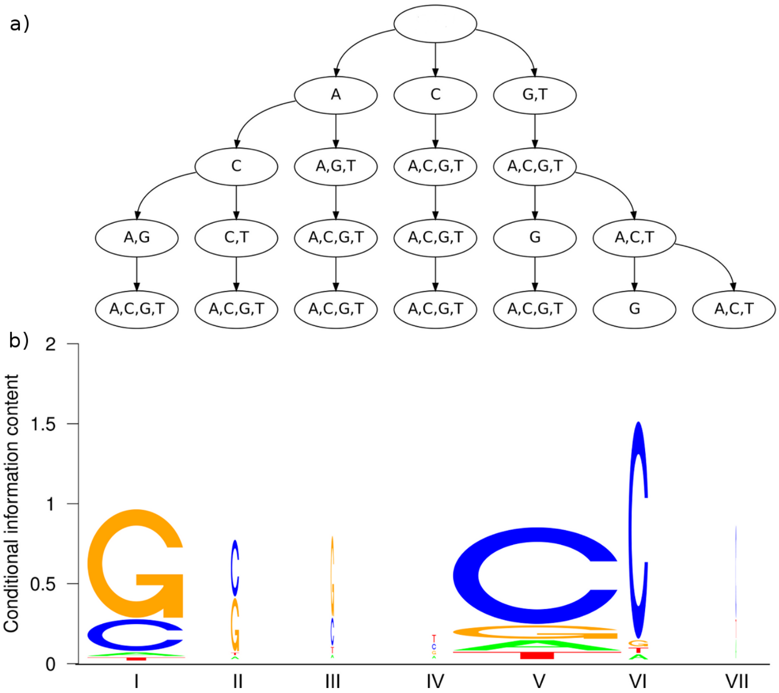


Figure 5.6.: **PCT and conditional sequence logo at position 17.** The PCT at position 17 (Figure a) is aligned with the corresponding conditional sequence logo (Figure b) Each stack of nucleotides represents the relative conditional nucleotide frequency given the context represented by the corresponding leaf. The width of the stack is scaled by the number of sequences that are represented by the leaf. We observe two dominating contexts, which yield either a `G` (context **I**) or a `C` (context **V**) as dominating nucleotide.

and plots the conditional nucleotide frequencies of each context. The stack of the nucleotide frequencies is aligned to the leaf that is representing the particular context. In order to point out the difference to a traditional sequence logo, we label the contexts with Roman numerals.

However, not all contexts at a position are equally important, since the number of sequences matching a particular context in the data set may differ to a great extent. It can be even misleading to focus on the conditional nucleotide frequencies of a context that represents only very few sequences. In order to take into account the importance of each context in the visualization, we scale the width of the nucleotide stack of a context linearly by the number of sequences in the predicted binding sites that are actually represented by that context. We exemplify the visualizations by CSLs using position 17 (Figure 5.6b) and position 19 (Figure 5.7b).

At position 17, we observe a case in which more than 90% of the predicted sequences fall upon two of seven contexts (**II** and **V**). The original sequence logo (Figure 5.4a) indicates that `C` and `G` occur at position 17 with similar probability. We find that the context determines which of the two alternatives is observed with high probability. Observing `ACA` or `GCA` at

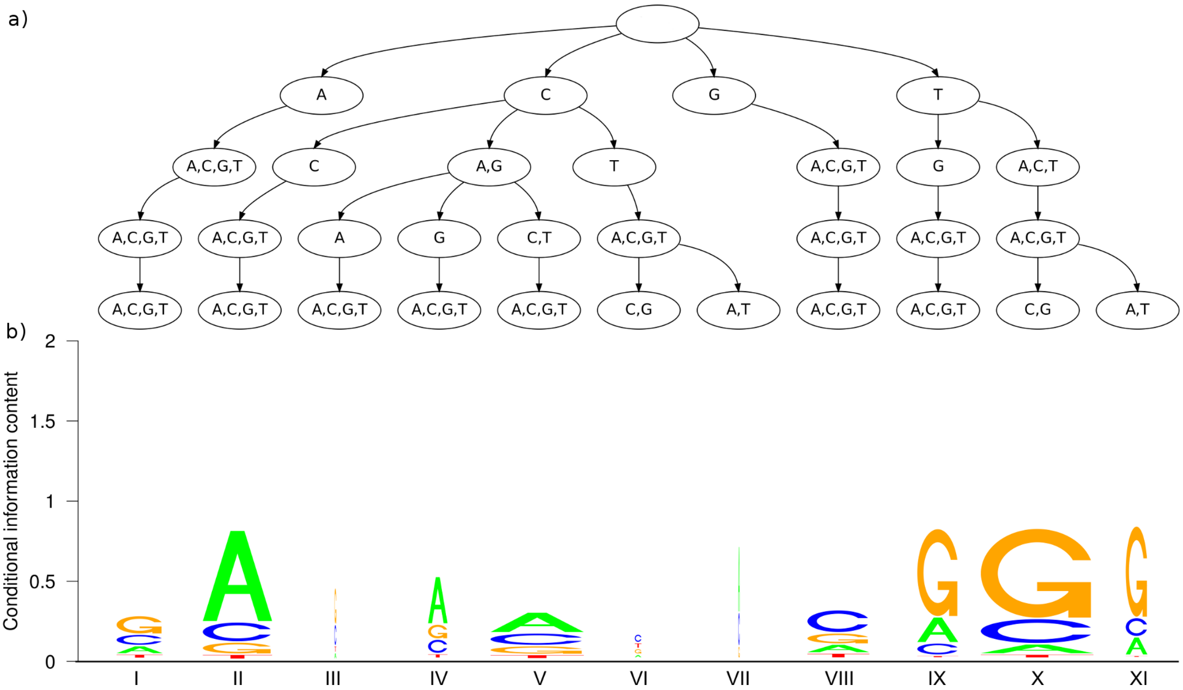


Figure 5.7.: **PCT and conditional sequence logo at position 19.** The structure of this figure is identical to that of Figure 5.6. Here, we observe more contexts representing a considerable amount of realizations, since only three context (**III**, **VI**, **VII**) represent so few sequences that they can be neglected. Among the other eight contexts, we observe two main types: If the preceding nucleotide (at position 18) is a `C` (context **II-VII**), there is a high probability of observing an `A` at position 19. If the predecessor is a `T` (context **IX-XI**), there is a high probability of observing a `G`. The differences among context within those two main types are smaller, yet not negligible, since they further refine the conditional probability distribution. One example are contexts **IX** and **X**, which differ in the nucleotide at the second and fourth predecessor. Context **IX** yields an `A` as second most probable nucleotide for position 19, whereas context **X** yields a `C`.

positions 14-16 (context **I**) increases the probability of finding a `G` at position 17, whereas observing `GNG` or `GNT` (context **V**) increases the probability of finding a `C`. The remaining five contexts represent less than 10% of the binding sites, thus the corresponding probability distributions should be judged with caution. This is represented by the horizontal scaling of the CSL: the smaller the width of a CSL, the fewer sequences contributed to its estimation. Context **VI**, which is similar to context **V** but differs at the third and fourth predecessor nucleotide, yields an even more increased probability of the dominating nucleotide `C`. For context **II**, which differs from context **I** at the third and fourth predecessor nucleotide, `G` and `C` are almost equally likely, whereas context **I** yields a clear preference towards `C`. Interestingly, the probability for finding a particular nucleotide at position 17 is mainly determined by the nucleotide at position 16, as it determines whether a `G` or a `C` is predominantly observed. This is a further explanation for the small ratio between fourth- and first-order mutual information at position 17 in Figure 5.4b.

At position 19 (Figure 5.7), the situation is more diverse. From the eleven contexts, only three are comparatively unimportant (**III**, **VI**, and **VII**), and the remaining eight contexts represent a substantial number of sequences each. By considering the (unconditional) sequence logo (Figure 5.4a), we find that position 19 is relatively unconserved, since we observe similar frequencies for `A`, `C`, and `G`, while only `T` rarely occurs. The conditional sequence logo indicates that the nucleotides `A` and `G` are conserved rather strongly if they are preceded by a particular context (**II** and **IX-XI** respectively). The first predecessor (position 18) determines again which nucleotide is predominantly observed, but in contrast to position 17, the remaining predecessors play a more important role, as we can see by considering contexts **IX-XI** and the corresponding CSL. All three contexts require a `T` at position 18, and all of them yield the same high probability of finding a `G` at position 19.

However, the second-most probable nucleotide strongly depends on the second and fourth predecessor. Observing a `G` at position 17 yields an `A` as second-most probable nucleotide at position 19. Observing no `G` at position 17 and `C` or `G` at position 15 yields a `C` as second-most probable nucleotide at position 19. In all other cases (`G` at position 17 and `A` or `T` at position 15), `A` and `C` are equally probable at position 19. These higher-order effects are the cause of the comparatively large ratio between fourth- and first-order mutual information for position 19 (Figure 5.4b).

Having analyzed the CSLs at positions 17 and 19, we may conclude that the first predecessor nucleotide is predominantly responsible for statistical dependencies, but taking into account second, third, and fourth predecessors may further refine the probability distribution. This explains with hindsight the results of the first classification experiment (Figure 5.1), where we observe the steepest ascent of sensitivity for models that are much less complex than the optimal model, and that further increase in complexity yields a smaller ascent towards the sensitivity value of the optimal model. For both position 17 and position 19 we finally observe that the maximal conditional information content of the CSL is much higher than the information content in the (unconditional) sequence logo at the corresponding position, which further explains the high mutual information in Figure 5.4b. These findings suggest that considering a sequence motif as a set of independent nucleotide frequencies is – at least in case of the binding sites of CTCF – not justified.

Bottom line

In this chapter, we studied intra-motif dependencies within the binding sites of human insulator protein CTCF and the effect of their utilization through PMMs on motif discovery performance in order to contribute to main objective II (Figure 1.11). Using a fragment-based classification approach for evaluating different motif models, we observed that taking into account intra-motif dependencies yields a 10% increase in sensitivity compared to a PWM model, which neglects intra-motif dependencies. Analyzing the predicted CTCF binding sites, we found significant mutual information between a nucleotide and its preceding oligomer at all sequence positions, although varying in strength considerably, with the strongest dependencies being located at the 3' end of the motif, where nucleotides are relatively unconserved. Finally, we investigated the nature of these dependencies by utilizing a new visualization of CPDs in PMMs, which we dubbed conditional sequence logo. We observed that some positions in the motif are not as uninformative as a traditional sequence logo, which neglects statistical dependencies, suggests. We also found that the strongest dependencies exist among two directly adjacent nucleotides, which is biophysically plausible, even though in some cases also higher-order dependencies play a significant role.

Considering these findings for insulator protein CTCF, it might be worthwhile to take into account intra-motif dependencies via parsimonious Markov models in the de novo motif discovery for different DNA binding proteins as well. However, a large-scale application is not feasible with the EM algorithm as learning method, which requires massive hyperparameter-tuning via cross-validation. In the next chapters, we thus studies alternative learning methods for PMMs in the presence of latent variables with the incentive of obtaining a method that allows a large-scale investigation of putative intra-motif dependencies for many different transcription factors.

In the first place, the best way to convey to the experimenter what the data tell him about is to show him a picture of the posterior distribution.

George E. P. Box & George C. Tiao

6

Model averaging with latent variables

In Chapter 4, we introduced latent variable models and discussed the theoretical difficulties of structure learning under latent variables, where model selection must be carried out within the M-step of an EM algorithm. Whereas we learned in Chapter 5 that in practice these difficulties can be coped with by sheer brute force approaches, i.e., massive cross-validation for hyperparameter-tuning, such a solution remains dissatisfactory for several reasons.

First, in contrast to fully observable data, model selection is here not carried out according to a well-defined and theoretically justified score, e.g., comparable to those of Section 3.2.1. Instead, model selection is basically carried out via cross-validation and the entire dynamic programming algorithm for finding optimal PCTs is only needed for narrowing down the space of candidates as input for cross-validation, since testing all possible candidates would be infeasible.

Second, cross-validation is already an expensive, time-consuming procedure for fully observable data (Section 3.3). This multiplies now, as within each cross-validation iteration not only a single model has to be learned, but an entire EM algorithm with possibly multiple restarts has to be run. Hence, we quickly obtain situation where even on a high-performance cluster the analysis of more than a few data sets becomes infeasible.

An alternative to model selection under latent variables is model averaging, which is – from a purely Bayesian point of view – the preferable method in the first place, since the belief in one single optimal model might be overly confident, especially when the sample size is small and the posterior distribution cannot be expected to have one sharp peak at its maximum. In addition to avoiding the aforementioned obstacles, a fully Bayesian prediction, which takes into account the full posterior distribution, is often superior to a prediction based on point estimators for practical applications on limited data. In fact, for arbitrary training data y and test data x, model structure ξ and probability parameter θ_ξ, the predictive distribution

$$P_1(x|y) = P(x|\hat{\theta}_{\hat{\xi}(y)}(y)), \tag{6.1}$$

with $\hat{\xi}(y)$ being the model structure learned from data y and $\hat{\theta}_{\hat{\xi}(y)}(y)$ being the corresponding parameter estimate, can be seen as an approximation of

$$P(x|y) = \sum_{\xi} P(\xi) \int P(x|\theta_\xi) P(\theta_\xi|y) d\theta_\xi, \tag{6.2}$$

which is the fully Bayesian prediction that sums over all discrete model structures and integrates over the probability parameter space of each structure.

In the presence of latent variables, neither the model selection tasks needed for computation of Equation 6.1 nor model averaging for computation of Equation 6.2 can be done analytically for more but the most simple cases, as it involves a summation of the exponential number of latent variable realizations. Whereas point estimates can be approximated using an EM algorithm, the approximation of the full posterior distribution $P(\xi, \theta_\xi|Y)$ requires a different strategy.

6.1. Model averaging for mixture models

In this section, we focus on a mixture model parameterized by $\mathbf{\Theta}$ as defined in Section 4.1 as example of a latent variable model. The mixture components are PMMs as defined in Section 2.2. Since analytic computation of the posterior, and thus an analytic computation of the fully Bayesian prediction of Equation 6.2, is intractable, the key idea is to approximate the integral by using parameter samples from the posterior.

We thus intend to generate a sample from the distribution $P(\mathbf{\Theta}, \vec{u}|\mathbf{x})$, from which $P(\mathbf{\Theta}|\mathbf{x})$ for use in Equation 6.2 can be further derived by marginalization. However, sampling from $P(\mathbf{\Theta}, \vec{u}|\mathbf{x})$ directly is also intractable.

Here we recruit the technique of Gibbs sampling [116, 32], which is a general method for generating samples from a joint distribution that is not directly available. Let $P(x, y, z)$ denote an arbitrary joint distribution that cannot be computed directly. If the conditional distributions $P(x|y, z)$, $P(y|x, z)$, and $P(z|x, y)$, we can iteratively sampling from these distributions, for instance

$$\begin{aligned} X^{t+1} &\sim P(x|y^t, z^t) \\ Y^{t+1} &\sim P(y|x^{t+1}, z^t) \\ Z^{t+1} &\sim P(z|x^{t+1}, y^{t+1}). \end{aligned} \tag{6.3}$$

This strategy can be seen as a Markov chain that generates a series of realizations $(x, y, z)^t$ for $t \in \mathbb{N}$. It can be shown that $P((x, y, z)^t)$ converges to $P(x, y, z)$ for $t \to \infty$. Whereas convergence is guaranteed only in the limit, it is in practice sufficient to find a step T where $P((x, y, z)^T)$ is sufficiently close to $P(x, y, z)$. The time points $t < T$ are typically called *burn-in phase*, and the time points $t > T$ are called *stationary phase*. Estimating the length of the burn-in phase is critical for Gibbs sampling applications, as only samples from the stationary phase should be used for approximating the target distribution.

6.1.1. Gibbs sampling for mixtures of PMMs

In analogy to the abstract sampling scheme as described above, we derive here a Gibbs sampler for a mixture model with PMMs as component models.

We sample in the t-th iteration $\mathbf{\Theta}^{(t)}$ from $P(\mathbf{\Theta}|\vec{u}^{(t-1)}, \mathbf{x})$ and $\vec{u}^{(t)}$ from $P(\vec{u}|\mathbf{\Theta}^{(t)}, \mathbf{x})$. The conditional probability of the parameters given the latent variables decomposes to

$$P(\mathbf{\Theta}|\vec{u}^{(t-1)}, \mathbf{x}) = \prod_{k=1}^{K} \prod_{\ell=1}^{W} P(\tau_{k\ell}, \vec{\theta}_{k\ell}|\vec{u}^{(t-1)}, \mathbf{x}). \tag{6.4}$$

For each component k and each position ℓ (and thus omitting here both indices for the sake of convenience), we use the idea of variable grouping [117] and sample $(\tau, \vec{\theta}_\tau)^{(t)}$ jointly from $P(\tau, \vec{\theta}_\tau|\vec{u}^{(t-1)}, \mathbf{x})$ instead of sampling from each single conditional distribution separately. This is achieved in a hierarchical manner, by decomposing the joint conditional probability distribution into $P(\tau|\vec{u}^{(t-1)}, \mathbf{x})P(\vec{\theta}_\tau|\tau, \vec{u}^{(t-1)}, \mathbf{x})$. We first sample $\tau^{(t)}$ w.r.t. its conditional distribution given $(\vec{u}^{(t-1)}, \mathbf{x})$, and afterwards $\vec{\theta}_\tau^{(t)}$ from its full conditional distribution, $P(\vec{\theta}_\tau|\tau^{(t)}, \vec{u}^{(t-1)}, \mathbf{x})$, under which the components of $\vec{\theta}_\tau$ are independent. Similarly, the components of $\vec{u}$ are conditionally independent given the other parameters and the data. As a result, we obtain the following sampling scheme.

Algorithm 2 Gibbs sampling for mixtures of PMMs.

```
initialize $\vec{u}^{(0)}$
for $t = 1, \ldots, T$ do
  for $k = 1, \ldots, K$ do
    for $\ell = 1, \ldots, W$ do
      sample $\tau_{k\ell}^{(t)}$ from $P(\tau_{k\ell}|\vec{u}^{(t-1)}, \mathbf{x})$
      for all $c \in \tau_{k\ell}$ do
        sample $\vec{\theta}_{k\ell c}^{(t)}$ from $P(\vec{\theta}_{k\ell c}|\tau_{k\ell}^{(t)}, \vec{u}^{(t-1)}, \mathbf{x})$
      end for
    end for
  end for
  for $i = 1, \ldots, N$ do
    sample $u_i^{(t)}$ from $P(u_i|\mathbf{\Theta}^{(t)}, \vec{x}_i)$
  end for
end for
```

The initialization of $\vec{u}$ can be implemented in different ways, and a straightforward approach is to sample a realization of each $u_i^{(0)}$ according to the component probabilities $\vec{\pi}$.

The remaining task is to efficiently sample from the specified conditional distributions, which we discuss in the following sections. Whereas both the sampling of latent variables and the sampling probability parameters are comparatively simple tasks, the particular challenge lies in the sampling of the PCTs.

Structure sampling

In this section, we focus on the sampling of a PCT structure. Let $\vec{N}_{k\ell c}$ denote the vector of nucleotide counts obtained for the ℓ-th position given context c in mixture component k, and let $\vec{\alpha}_{k\ell c}$ denote the corresponding hyperparameters of the local Dirichlet prior. The probability of a particular tree structure given data is

$$P(\tau_{k\ell}|\vec{u}, \mathbf{x}) \propto \prod_{c \in \tau_{k\ell}} \kappa \frac{\mathcal{B}(\vec{N}_{k\ell c} + \vec{\alpha}_{k\ell c})}{\mathcal{B}(\vec{\alpha}_{k\ell c})}. \tag{6.5}$$

Hence, the probability decomposes into a product of scores for each context, i.e., leaf scores for each leaf in the PCT. Each leaf score is itself a marginal likelihood for the particular context multiplied with the structure prior hyperparameter κ. While the probability for a given tree can be computed easily, the challenge lies in sampling one out of a super-exponential number (with respect to model order and alphabet size) of possible PCTs, without computing the probability for every single tree explicitly.

To this end, we propose a dynamic programming algorithm (Algorithm 3), which is inspired by the maximization algorithm for PCTs (Algorithm 1 in Section 1.4.2), and which thus uses the same data structure, the extended PCT (Figure 1.9), as well as the same notation.

In analogy to the maximization algorithm, we now sample a regular PCT using the following algorithm.

(i) If n is a leaf (representing context c), we compute the score $\kappa \frac{\mathcal{B}(\vec{N}_{k\ell c} + \vec{\alpha}_{k\ell c})}{\mathcal{B}(\vec{\alpha}_{k\ell c})}$, and assign it to n.

(ii) If n is an inner node, we first compute the probability of each *valid choice* of children of n, where a valid choice is a set of children, whose labels form a partition of $\mathcal{A}$ and the score of a valid choice is simply the product of the scores of the children contained in it. Next, one valid choice is sampled according to the computed probability distribution. The probability of this sampled set of children becomes the score of n. The remaining children of n, which do not belong to the sampled set, and all subtrees below are discarded. Hence n becomes the root of a subtree that satisfies the characteristics of a PCT and has a score assigned to it.

We obtain a complete PCT, once we have sampled a valid choice of children of the root of the extended tree.

The algorithm samples correctly from the posterior distribution for the following reasons: First, when sampling the children of a particular node, the scores of all potential children

Algorithm 3 Dynamic programming sampling PCT subtrees

```
samplePCTSubtree($n$)
  if $n \in \mathcal{L}(\mathcal{T}_d^A)$ then
    compute $s(n) = \kappa \frac{\mathcal{B}(\vec{N}_{k\ell c}+\vec{\alpha}_{k\ell c})}{\mathcal{B}(\vec{\alpha}_{k\ell c})}$
  end if
  if $n \in \mathcal{I}(\mathcal{T}_d^A)$ then
    for all $m \in \mathcal{C}(n)$ do
      samplePCTSubtree($m$)
    end for
    for all $v \in \mathcal{V}(\mathcal{C}(n))$ do
      $s(v) := \prod_{m \in v} s(m)$
    end for
    sample $v^\star$ from $\frac{s(v)}{\sum_{v' \in \mathcal{V}(\mathcal{C}(n))} s(v')}$
    $s(n) := \frac{s(v^\star)}{\sum_{v' \in \mathcal{V}(\mathcal{C}(n))} s(v')}$
    for all $m \in \mathcal{C}(n) \setminus v^\star$ do
      remove $m$ and subtree below
    end for
  end if
```

are already available. In step (ii), we can safely assume that for each inner node n, each child (Figure 1.9) is either a leaf, in which case we have obtained its score by step (i), or the root of a subtree that already satisfies the characteristics of a PCT and has a score assigned to it. Second, the subtree rooted at an arbitrary node n and subtrees rooted at the siblings of n are conditionally independent given the labels on the path from n to the global root of the PCT. This information is available at any time due to the top-down construction of the extended tree.

The time complexity of the algorithm is identical to that of the maximization algorithm, as it is given by the number of valid choices of child nodes multiplied by the number of inner nodes in the extended tree, plus the total number of score computations in the leaves.

Parameter sampling

The probability of the conditional probability parameters of a given context of a given PCT structure is a Dirichlet distribution with the hyperparameters being a sum of counts and pseudocounts, that is,

$$P(\theta_{k\ell c}|\tau_{k\ell}, \vec{u}, \mathbf{x}) = \mathrm{Di}(\theta_{k\ell c}|\vec{N}_{k\ell c} + \vec{\alpha}_{k\ell c}). \tag{6.6}$$

Samples from a Dirichlet distribution can be obtained by standard libraries, here we use the Jstacs [118] implementation.

Latent variable sampling

The conditional probability distribution of the latent variables is merely a fraction of likelihood values, that is,

$$P(u_i|\mathbf{\Theta}, \vec{x}_i) = \frac{\pi_{u_i} P(\vec{x}_i|\Theta_{u_i})}{\sum_{k=1}^{K} \pi_k P(\vec{x}_i|\Theta_k)}. \tag{6.7}$$

It is noteworthy that given the parameters and the data, the components of $\vec{u}$ are mutually independent. In addition, the conditional distribution of each component u_i depends on the data only through the sequence $\vec{x}_i$.

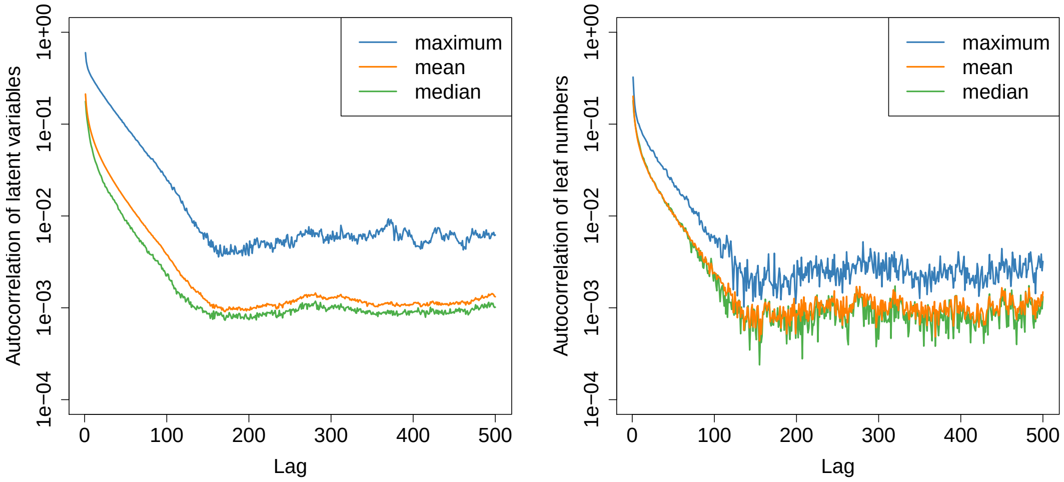
6.1.2. Case studies

In this section, we empirically evaluate the performance of the Gibbs sampler for mixtures of PMMs. We study its convergence behavior (Section 6.1.2) in order to determine the length of the burn-in phase, and compare the fully Bayesian prediction that makes use of the Gibbs sampling output with the prediction based on point-estimates resulting from the corresponding EM algorithm (Section 6.1.2). To this end, we use the same splice donor site data of Yeo and Burge [99] as used for benchmarking in Section 3.3.

Convergence

Gibbs sampling, like any MCMC method, samples from the target distribution only asymptotically. It is therefore important to study the convergence rate of the algorithm in order to control the quality of the approximation brought by the sample. In practice, a number of first samples is generated but discarded, thereby skipping the burn-in phase of the sampler. We investigate the convergence for a data set of size $N = 500$, since we use data sets of the same size for a prediction study in section 6.1.2. After initializing each latent variable by sampling from the uniform distribution $(0.5, 0.5)$, we perform 10^4 iterations of the Gibbs sampler. In each iteration step, we store the sampled value of each of the 500 latent variables. In absence of a simple notion of correlation among PCTs, we focus here on the number of leaves of the PCTs. The tree of the first position of both components can be neglected, since it always consists of one leaf. So we store only the number of leaves of each PCT at position 2-7 in both components, yielding 12 additional variables per iteration, which makes 512 variables in total. Since the space of actively used probability parameters changes from iteration step to iteration step, we do not measure their convergence behavior.

Next, we compute the autocorrelation function for each of the 512 variables with lags of 1 to 500 from the 10^4 iterations. We repeat the process 10^2 times with different initializations and average the autocorrelation coefficients for each variable. We project the results to six curves in the following way. First, we investigate the latent variables and the PCT complexity separately. Second, we plot for each lag the (i) maximum, (ii) mean, and (iii) median of



(a) Hidden variables

(b) Number of Leaves

Figure 6.1.: **Convergence behavior of sampled latent variables and PCTs.** We show autocorrelation coefficients of latent variables and number of leaves of the parsimonious context trees in logarithmic scale. Figure 6.1(a) depicts for each lag the mean, median, and maximum of the autocorrelation over the 500 latent variables. Figure 6.1(a) depicts the same statistics of the autocorrelation of the PCT leaf number over the 12 nontrivial PCTs in the mixture model. In all cases, the autocorrelation function shows a nearly exponential decay with a decay constant of approximately 1.7 until it remains stable at low values at lag 150-200.

the absolute autocorrelation, which is shown in Figure 6.1. In both cases, we observe an approximately exponential decay of the autocorrelation up to a lag of approximately 150. For larger lags, it differs only slightly, even though there is a small increase between lag 250 and 350. We finally conclude that 200 iteration steps is the minimal length of the burn-in phase, since all samples before that are still correlated to the random initialization.

Prediction

After having evaluated convergence behavior of the Gibbs sampling algorithm, we study the predictive performance compared to the EM algorithm. In analogy to the convergence study, we use a mixture of two second-order PMMs. To this end, we use a sub-sample of $N = 500$ sequences from the training data set of Yeo and Burge.

For the Gibbs sampler, we consider the first 400 iterations as burn-in phase and use the following 500 samples from the stationary phase to approximate the posterior for a Bayesian prediction according to Equation 6.2. We restart the Gibbs sampler for different values of structure hyperparameter κ. From each iteration step and each value of κ, we compute the sum of leaves in all PCTs (both mixture components), and average this value, for each

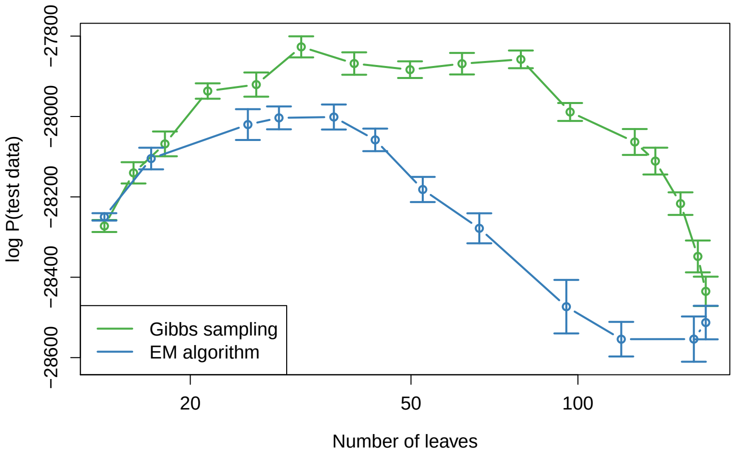


Figure 6.2: **Prediction performance on splice site data.** We show the mean log prediction of a repeated hold-out experiment for the Gibbs sampler and the EM algorithm for different model complexities. The Bayesian prediction using the Gibbs sampler outperforms the EM algorithm for all possible model complexities.

κ separately, over all iteration steps in the stationary phase. Next, we compute the log predictive probability on the test data set of Yeo and Burge.

For the EM algorithm, we use a log posterior difference of 10^{-6} as termination condition and restart the algorithm 10 times with different initializations. Here, we also use different values of κ, even though the selection of values differs dramatically from the Gibbs sampling experiment. For each value of κ, we report the learned model complexity and compute the log predictive probability on the test data according to Equation 6.1.

We repeat the entire procedure for both algorithms with 10 different samples of 500 sequences from the training data set, and average model complexities and log predictive probabilities. Finally, we plot the average log predictive probability against the average model complexity (Figure 6.2).

We observe that for all model complexities, the Bayesian prediction using the Gibbs sampler yields a better prediction than the plug-in prediction using the EM algorithm. In addition, the standard errors for the Gibbs sampler are smaller. Moreover, the Gibbs sampler yields an acceptable, though not optimal, result when a uniform structure prior is used. We obtain 63 leaves on average and a mean log prediction of -27928.8, whereas the EM algorithm learns for a uniform prior almost the full model, namely 161 leaves and a mean log prediction of -28553.9.

This indicates that, for a fully Bayesian prediction, a uniform prior might be already a reasonable choice when the maximal PCT depth is not too large, and that extensive cross-validation for hyperparameter-tuning could be avoided in those cases. A downside of the Gibbs sampler, however, is the slow prediction. Whereas the sampling algorithm is w.r.t. running time in the same order of magnitude compared to the EM algorithm, the running time of the Bayesian prediction increases by a factor of M, with M being the number of sampled parameter sets.

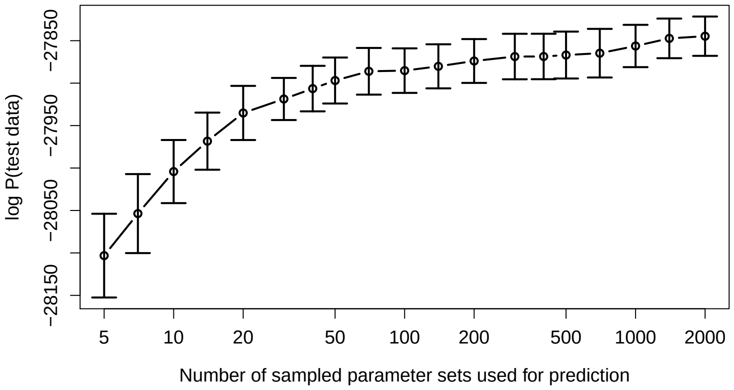


Figure 6.3: **Influence of the number of sampled parameter sets** on the Bayesian prediction performance. We observe that at least 100 parameter samples are necessary for obtaining a reliable approximation of the posterior.

An open question is which value for M is sufficient to obtain a reliable approximation of the posterior. We expect an increasing prediction performance with increasing M, but decreasing relative prediction improvement. To study this effect, we repeat the previous study for the uniform structure prior ($\kappa = 1$) with varying values for M and show the results in Figure 6.3.

We observe an increasing prediction performance with increasing value of M until it reaches a plateau and remains at a similar level for $M > 100$. Hence, for a Bayesian prediction of splice sites using a two-component mixture of second-order PMMs, at least 100 parameter samples should be used to approximate the posterior distribution. However, these values may differ from data set to data set. Typically larger data sets yield a sharply peaked posterior distribution, which then requires less parameter samples for approximation, whereas for smaller data sets typically the opposite holds. However, data set size is not the only influencing factor, as the diversity of the observed data points plays an important role w.r.t. the shape of the posterior as well.

6.2. Model averaging for motif discovery

In the previous section, we observed that in the case of learning mixture models with PMM components from splice site data, model averaging based on Gibbs sampling constitutes a promising alternative to the model selection approach using the modified EM algorithm. In this section, we now apply the same concept to the problem of motif discovery using PMMs in order to obtain an alternative to the EM algorithm used in Chapter 5.

6.2.1. Algorithm

We here use the ZOOPS model as promoter model, as specified in Section 4.2, with the alternative definition of latent variables, which combines occurrence, position, and strand orientation of a binding site in $\vec{x}_i$ in a single latent variable h_i. We denote the set of all binding sites that are identified by a current state of latent variables $\vec{h}$ in data set $\mathbf{x}$ as $\mathbf{x}_{\vec{h}}$. The Gibbs sampler for sampling structure and parameters of the PMM motif model is shown in Algorithm 4.

Algorithm 4 Gibbs sampling for motif discovery using PMMs.

```
initialize $\vec{h}^{(0)}$
for $t = 1, \ldots, T$ do
  for $i = 1, \ldots, N$ do
    sample $h_i^{(t)}$ from $P(h_i|\vec{x}_i, \mathbf{\Theta}^{(t-1)})$
  end for
  for $\ell = 1, \ldots, W$ do
    sample $\tau_\ell^{(t)}$ from $P(\tau_\ell|\mathbf{x}_{\vec{h}^{(t)}})$
    for $c \in \tau_\ell^{(t)}$ do
      sample $\theta_{\ell c}^{(t)}$ from $P(\theta_{\ell c}^{(t)}|\tau_\ell^{(t)}, \mathbf{x}_{\vec{h}^{(t)}})$
    end for
  end for
end for
```

Both structure sampling and parameter sampling for the motif model use the same procedures as in the case of one component of a mixture model (Section 6.1.1). The difference to mixture models is the sampling of each latent variable v_i from its conditional distribution given model structures and parameters, which writes here as

$$P(h_i|\vec{x}_i, \mathbf{\Theta}) = P(u_i = \delta_{h_i,0}, v_i = \left\lfloor \frac{h_i}{2} \right\rfloor, s_i = h_i \bmod 2|\vec{x}_i, \mathbf{\Theta}). \tag{6.8}$$

6.2.2. Case studies

For the evaluation of the Gibbs sampler for motif discovery, we use the CTCF data set for H1-hESC cell line from Chapter 5. We use a ZOOPS model with a PMM of width $W = 20$ as motif model and a second-order homogeneous Markov model as flanking model. The parameters of the flanking model parameters are estimated from the union of training and test data set and kept fixed during the entire learning procedure. Since we observed in Section 6.1.2 that a uniform structure prior is a reasonable choice for model averaging, we use here also $\kappa = 1$.

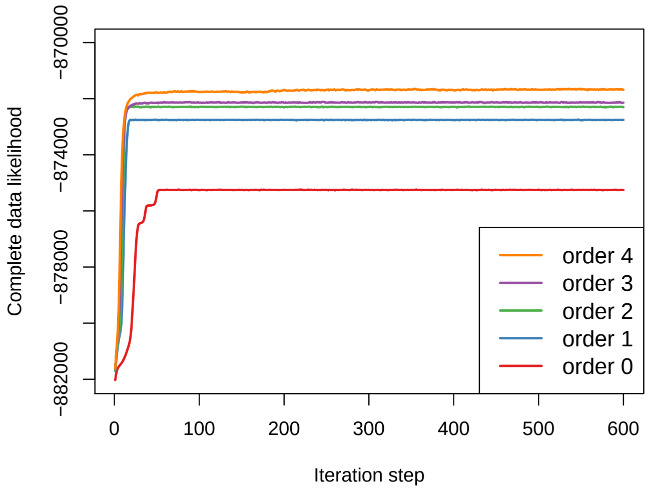
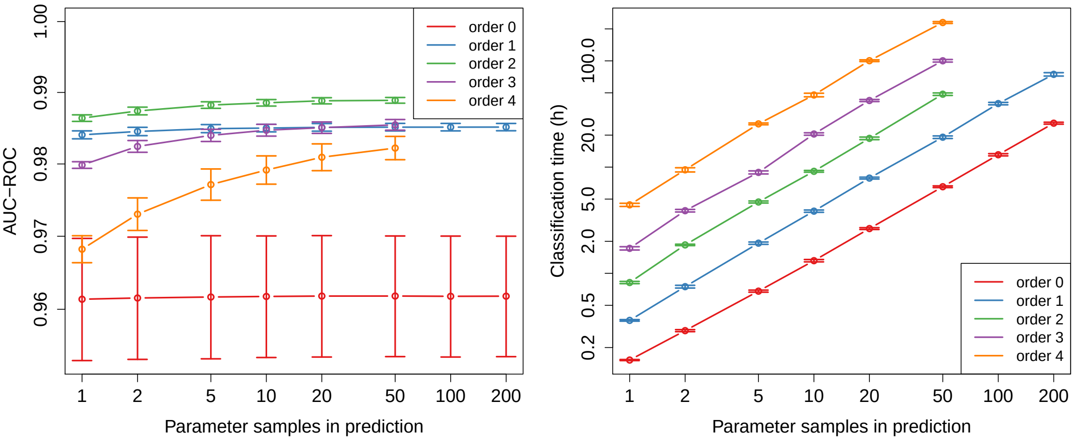


Figure 6.4: **Convergence of the Gibbs sampler for motif discovery using PMMs.** The complete data likelihood is averaged over ten independent restarts. After approximately 60 iteration steps, all algorithms have converged to a stationary phase.

Convergence

In a first study we investigate the convergence behavior of the algorithm. For motif discovery it is inappropriate to measure convergence by autocorrelation, since identifying a correct motif results in sampling very similar latent variable configurations in subsequent iteration steps, which yields very high autocorrelation values for each latent variable. We thus measure convergence by the completely data likelihood $P(\mathbf{x}|\vec{h}^{(t)}, \mathbf{\Theta}^{(t)})$ (CDL), which we plot for increasing iteration steps $t = 1, \ldots, 600$ and for different motif model orders $d = 0, \ldots, 4$, averaged over ten independent restarts with different initializations, in Figure 6.4. We observe that 30 to 60 iteration steps are required before the algorithm convergences. Convergence is here for motif discovery on CTCF data generally more favorable than in the case of learning mixture models on splice site data, which can be explained by the fact that most possible latent variable assignments have a very low probability and once the algorithm has identified the motif, the latent variable assignment changes only slightly.

The only exception to that general statement is the PWM model (PMM of order $d = 0$), which has the slowest convergence, since there are small steps in the likelihood function, which can be explained as follows. If the motif model neglects dependencies, the algorithm is more prone to phase shifts, that is, different restarts identify the same motif, but it may be shifted by some positions. In case that some positions that are only informative through dependencies, such as at the 3' end of the CTCF motif, are lost, such a shifted motif yields a substantially lower complete data likelihood in relation to the optimal shift. This effect causes (i) a generally low average CDL, as suboptimal shifts are included in the average, and (ii) small steps in the first part of the CDL, as not all shifts are identified within exactly the same number of iteration steps. Here, we thus observe directly that dependencies in the data does not only exist, but that ignoring them through an oversimplified model may make the learning procedure more difficult.



(a) Performance (b) Running time

Figure 6.5.: **Bayesian fragment-based classification on CTCF data.** Figure a) shows the classification performance measured by the area under the ROC curve. Figure b) shows the required time on a 2.4 GHz server. The x-axis shows for both plots the number of parameter samples the posterior is approximated with in the Bayesian prediction of binding site occurrence, which is a subtask of the total classification problem.

Classification

In a second study we investigate the performance of the fully Bayesian approach for the task of fragment-based classification as introduced in Section 5.2.1. Learning from the results of the convergence study for motif discovery on CTCF data (Figure 6.4), we consider the first 100 iteration steps as burn-in phase, and use subsequent parameter samples for approximating the posterior to carry out a fully Bayesian fragment-based classification. For different initial model order $d = 0, \dots, 4$, we show the area under the ROC curve in Figure 6.5(a).

We observe that the classification performance depends on model order d and taking into account intra-motif dependencies through PMMs improves the simple PWM model, which is well in accordance with the findings of Chapter 5. For $d = 0$ (PWM model) the classification performance remains almost constant with increasing number of parameter samples, which is not surprising as there is no model averaging, and the posterior of the probability parameters of a PWM model can be approximated well by a single point estimate when the input data is as large as the CTCF data set, which contains several thousands of data points. For larger values of d, the classification performance depends on the number M of used parameter samples, and generally the improvement gained by taking into account more parameter samples grows with d. This can be explained by the fact that the number of possible PCTs grows exponentially with d, hence it becomes more unlikely that one (or a few) samples are representative candidates for approximating the posterior. In fact, we may speculate that the observation of $d = 2$ yielding the overall best performance, which is in slight contrast to Chapter 5, can be simply explained by the fact that $d = 3$ and $d = 4$ need to take into account more parameter samples in order to fully benefit from the Bayesian approach. In contrast to $d = 1$ and $d = 2$ both prediction curves have not yet reached a plateau where further improvement with increasing M is unlikely to be expected.

However, further increasing M for larger values of d is limited by running time for performing the Bayesian prediction. While the Gibbs sampler itself is not significantly slower than a corresponding EM algorithm, computing the average over all sampled parameters for each tested sequences grows linearly in M (Figure 6.5(b)). This increase in running time w.r.t. M is inherent to the Bayesian prediction in comparison to the prediction based on a single point estimate and thus cannot be avoided. For the problem of binding site prediction based on de-novo motif discovery, the factor M quickly becomes the limiting factor, i.e., the running time required for prediction based on M samples quickly dominates the time the algorithm needs for generating those M samples. For evaluating the quality of a de novo discovered motif, prediction probabilities for a large number of potential binding sites have to be computed, which involves every position in both strand orientations in every sequence in both the positive and the negative data set. In addition, complete genomic scans for the occurrence of a motif also become quickly intractable when being based on a Bayesian prediction with sufficiently large M.

Bottom line

In this chapter, we investigated an alternative to the EM algorithm for learning PMMs in a setting with latent variables in order to contribute to main objective I (Figure 1.11). Instead of seeking an optimal model for a point estimate-prediction, we generated a series of structure and parameter samples in order to approximate the posterior for a fully Bayesian prediction. We applied the Gibbs sampler to mixture models and promoter models and found in both cases convergence to be fast. Moreover, the Gibbs sampler does not share the theoretical problems of the EM algorithm, as the uniform structure prior does lead to sampled PCTs of appropriate complexity. While the fully Bayesian prediction via Gibbs sampling avoids one problem of the EM algorithm, namely the conceptual necessity of hyperparameter-tuning via internal cross-validation, it has the disadvantage of requiring substantially more time for prediction, so the time gained in learning may be then again lost in prediction or classification. Hence, a robust alternative that avoids both problems, the need for hyperparameter-tuning and a computationally expensive prediction, is required, and such a method is the topic of the next chapter.

Speed is the essence of war.

Sun Tzu

7

Model selection with latent variables

While conducting scientific research has certainly a different incentive than conducting warfare, it shares the property that speed is of crucial importance for being successful. A statistical algorithm for scientific data analysis that yields good predictions at the cost of being extremely slow may be literally useless when there are alternatives that are much faster while making only small compromises w.r.t. to the quality of the prediction.

While there are certainly situations where extremely accurate solutions are of importance, many scientific projects rather benefit from doing more computational analyses, especially when predictions are experimentally verified in any case. Judging by the empirical results collected so far, learning mixtures of PMMs or applying PMMs to the problem of motif discovery may be such an instance where the gain in prediction performance in relation to simpler approaches may not fully justify the additionally invested amount of computational resources.

In the previous chapters, we studied different approaches for learning model structures in the presence latent variables that partition the data set in several subsets, either for learning a mixture component model for each subset, or for identifying binding sites to estimate a sequence motif from. Both approaches discussed so far are limited by being overly costly with respect to the computational resources that are required, but the reasons for that fundamentally differ from each other in both cases.

Model selection according to the MAP principle via EM algorithm essentially requires hyperparameter-tuning via cross-validation (Chapter 5). Since an EM algorithm with possibly multiple restarts has to be run in each cross-validation iteration, the entire structure learning is either extremely slow if cross-validation is carried out sequentially, or it requires a high performance cluster that enables massive parallelization. Once an optimal structure is estimated, subsequent prediction of unseen data is comparatively fast, though, as only one set of model structures and parameters needs to be stored and a single likelihood needs to be computed for each test sequence.

Model averaging based on Gibbs sampling as discussed in Chapter 6 does not necessarily require hyperparameter-tuning for sampling reasonable model structures (though it may improve prediction performance), and the Gibbs sampling algorithm itself is comparable in running time to a single run of the EM algorithm. However, here the prediction step is costly, as for each sequence to be tested, M parameter samples have to be loaded, parsed, and be used for computing the likelihood. A linear increase of the running time by a factor M does becomes a practical problem if the data sets that have to be tested are large, and while this problem could be coped with by massive parallelization, the total amount of invested resources is still substantial.

Hence, neither model selection via the EM algorithm nor model averaging via Gibbs sampling are completely satisfying solutions for dealing with models of variable structure in the presence of latent variables. In order to keep the prediction feasible, we need to approximate the full posterior over model structures by a single structure, but since we intend to avoid hyperparameter-tuning, we cannot use a standard EM framework.

In this chapter, we propose a solution for this problem by studying a method for performing fast and robust model selection in a setting with latent variables, which then also enables a fast prediction.

Previous approaches [104, 107] that tackled a comparable problem are based on the attempt to incorporate the structure learning step into a framework for latent variables, i.e., into an EM algorithm for maximizing the likelihood or posterior of the complete latent variable model. This approach has the crude consequence of performing model selection according to the ML or MAP principle, which may be reasonable when the parameter space of all candidate models has the same dimensionality, but is inappropriate when this is not the case.

Here, we approach the problem of combining latent variables with model selection from the opposite point of view, by generalizing the structure learning problem for fully observable data to learning structures for multiple models in the presence of latent variables. Even though the incentive for combining structure learning with latent variables arises from PMMs and their applicability in computational biology, we here state the problem from general viewpoint that is widely independent of model class and their particular application, provided that structure learning is feasible for fully observable data. We thus assume that the structure learning problem

$$\hat{\xi} = \operatorname*{argmax}_{\xi} = S(\xi|\mathbf{x}) \tag{7.1}$$

for model structure ξ and data $\mathbf{x}$ can be solved efficiently, which requires that the structure score S can be computed quickly. This is holds, among others, for the Bayesian marginal likelihood, factorized NML, BIC, or AIC, which we investigated for PMMs in Chapter 3. Moreover, the number of possible realizations of ξ has to be either small enough to enumerate all of them explicitly (such as different orders of Markov models), or there has to be an algorithm that finds provably the optimum by searching through the space of all model structures without explicit enumeration (PMMs, VOMMs, BNs).

If both conditions are fulfilled, we can approximately solve the structure learning problem by a stochastic algorithm that reduces the latent variable problem to structure learning for fully observable data. In the remainder of this chapter, we discuss the idea for two different settings, namely learning mixture models in Section 7.1 and motif discovery in Section 7.2.

7.1. Model selection for mixture model components

In this section, we focus on a mixture of K component models, each of which may have a variable structure. In contrast to previous work [104], we here explicitly allow the candidate structures for each component to be of different size, i.e., representing a parameter space of different dimensionality.

7.1.1. Optimization problem and algorithm

We attempt to solve a structure learning problem, i.e., finding the optimal model structures of each component, and a clustering problem, i.e., finding the optimal latent variable configuration, simultaneously. We denote the structure of the k-th mixture component by ξ_k.

When the realizations of latent variables are given, we define the structure score of the full mixture model, which we denote by S^{mix}, simply as the sum

$$S^{\mathrm{mix}}(\vec{\xi}|\mathbf{x},\vec{u}) = \sum_{k=1}^{K} S(\xi_k|\mathbf{x}_{\vec{u}=k}), \tag{7.2}$$

and we refer to this quantity as *joint structure score*. Here we assume w.l.o.g. the individual structure scores to have a logarithmic interpretation, e.g., as penalized log-likelihood such as BIC or AIC. For cases, were structure scores have an interpretation as probabilities (such as the Bayesian marginal likelihood), we take the logarithms of the original scores for each mixture component.

Computing the joint structure score for given latent variables is trivial, as it reduces to computing individual scores for each component for fully observable data. From this follows that model selection can be carried out individually for each component, as the joint structure score is additive w.r.t. mixture components.

In absence of a realization of the latent variables, we seek the score of Equation 7.2 for the best possible configuration of latent variables, that is,

$$S^{\mathrm{mix}}(\vec{\xi}|\mathbf{x}) = \max_{\vec{u}} S^{\mathrm{mix}}(\vec{\xi}|\mathbf{x},\vec{u}), \tag{7.3}$$

which we call *marginal structure score*. Here, the computation of the score is not straightforward anymore, as it involves enumerating all possible realizations of $\vec{u}$, which grows exponentially in N, in order to find the realization that yields the highest joint structure score. This is the same problem that renders exact parameter learning for mixture models infeasible. Whereas for parameter learning the EM algorithm provably approximates at least a local maximum or a saddle point of the likelihood [33], there is no efficient algorithm known that achieves the same for the score of Equation 7.3.

In order to cope with these difficulties, we propose an approach that is inspired by the Gibbs sampler of Section 6.1.1 and display it as pseudo-code in Algorithm 5. It relies on the observation that once samples of the latent variables are given, we do have a realization that reduces the structure learning problem for latent variables to structure learning for fully observable data.

Algorithm 5 Iterative algorithm for finding the optimal model structures in a setting with latent variables.

```
for $t = 1, \dots, T$ do
    for $i = 1, \dots, N$ do
        sample $u_i^{(t)}$ from $\begin{cases} \left(\frac{1}{K}, \dots, \frac{1}{K}\right) & \text{if } t = 1 \\ P(u_i|\vec{x}_i, \mathbf{\Theta}^{(t-1)}) & \text{if } t > 1 \end{cases}$
    end for
    for $k = 1, \dots, K$ do
        compute $\xi_k^{(t)} = \underset{\xi_k}{\operatorname{argmax}}\, S(\xi_k|\mathbf{x}_{\vec{u}^{(t)}=k})$
        estimate $\theta_{k\xi_k}^{(t)}$ from $\mathbf{x}_{\vec{u}^{(t)}=k}$
    end for
end for
```

The algorithm samples the latent variables from their conditional distributions given model structures and probability parameters, and is for this subtask identical to the Gibbs sampler of Section 6.1.1. However, it does not sample model structures, but solves the maximization problem according to the individual structure scores for each component. With given structures, the estimation of the corresponding probability parameters can be performed according to one of the criteria from Section 3.2, depending on whether prior knowledge should be incorporated or not.

In analogy to the Gibbs sampler, this algorithm also generates a series of $(\vec{u}, \mathbf{\Theta})^{(t)}$ for $t = 1, \dots, T$. However, there are some key differences, which we discuss in the following. From the perspective of Bayesian model averaging, the parameter samples $\mathbf{\Theta}^{(t)}$ are of primary interest in order to approximate the full posterior, whereas the sampled latent variables are little more than a byproduct. Here, we are mainly interested in finding the optimal configuration of latent variables, which maximizes the joint structure score (Equation 7.3). Since $\mathbf{\Theta}^{(t)}$ is – in contrast to the Gibbs sampler – deterministically obtained when $\vec{u}^{(t)}$ is known, it is sufficient to store, apart from the current latent variable assignment $\vec{u}^{(t)}$, only one additional realization of $\vec{u}$, namely the one that did yield the highest joint structure score in course of the current algorithm run and thus represents the currently optimal solution.

For a series of sampled latent variable configurations $\vec{u}^{(1)}, \ldots, \vec{u}^{(T)}$, we thus select $\hat{\vec{u}} = \vec{u}^{(\hat{t})}$ with

$$\hat{t} = \operatorname*{argmax}_{t \in (1,\ldots,T)} S^{\text{mix}}(\vec{\xi}|\mathbf{x}, \vec{u}^{(t)}), \tag{7.4}$$

which can be subsequently used to find optimal model structures and probability parameter estimates according to the scores of interest.

The algorithm can be interpreted as a variant of a stochastic EM algorithm [119], which also replaces the expected sufficient statistics for the missing data by parameter samples in the so called E step, but maximizes the parameters in the M step in accordance with a standard EM algorithm, which we here explicitly intend to avoid. The approach is at least partially motivated by the desire to be capable of using a structure score such as fNML, which requires discrete data to compute the multinomial normalizing constant, making it unsuitable for structure learning from expected sufficient statistics.

For structure scores that can cope with weighted data, the algorithm could be modified to use expected sufficient statistics without a sampling step, thus resulting in an EM algorithm with modified M step. However, such an algorithm would be deterministic and thus could not benefit from a sampling step for avoiding getting trapped in local optima, which often is an incentive to use a stochastic EM algorithm for parameter learning.

One drawback of a stochastic algorithm is the fact that it is heuristic by nature. There is no guarantee $\hat{\vec{u}}$ obtained through 7.4 is even close to the maximum required for Equation 7.3. Hence, an empirical investigation of the performance of the algorithm is inevitable, and we fill this gap in Section 7.1.2.

Nevertheless, there are two sanity checks of the entire approach from theoretical point of view, which we obtain by investigating two important special cases. First, we consider $K = 1$, i.e., there is only one single mixture component. In that case, the entire approach is reduced to a selection of one model structure for fully observable data by construction, as there is only one possible latent variable assignment: $\vec{u} = \vec{1}$. Second, we consider the case where each of the K components allows only a single model structure, which may differ among components, though. Now, Equation 7.2 is still a sum of penalized log-likelihoods, but all penalty terms remain constant during the algorithm and are thus are neglected in maximization. Hence, the optimal latent variable configuration is then computed with the attempt to maximize the likelihood of the mixture model, which may be a reasonable approach as long as prior knowledge is not explicitly taken into account.

7.1.2. Case studies

The purpose of the following case studies is to investigate the behavior of the proposed algorithm for model selection within mixture components. To this end, we apply the method to learning mixtures of PMMs from the splice site data of Yeo and Burge [99] from Section 3.3,

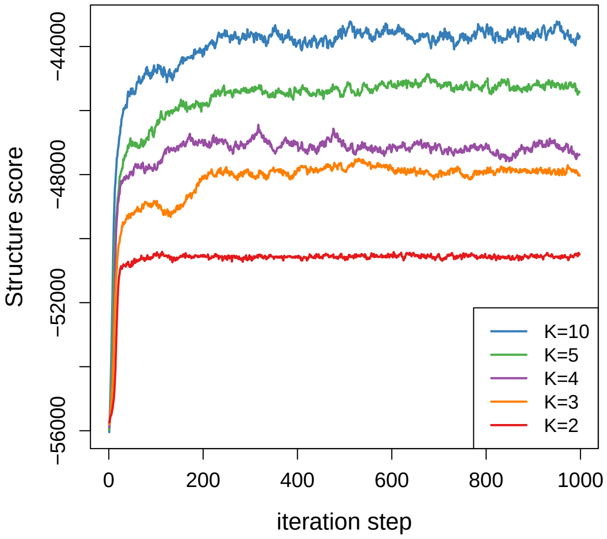


Figure 7.1: **Development of the joint structure scores.** We plot the score of Equation 7.2 during the run of the stochastic algorithm for learning mixture models with $K \in (2, \ldots, 5, 10)$ components of second-order PMMs on splice site data against the iteration steps.

using the original partitioning into training and test data at the ratio of 2:1.

Convergence

In a first study, we investigate the convergence behavior of the algorithm for the given model class and data. We learn several mixture models with different number of components, i.e., $K \in \{2, 3, 4, 5, 10\}$ on the training data set. As component models, we use second-order PMMs. We choose fNML as structure score S and fsNML for estimating the probability parameters.

We show the joint structure score $S^{\text{mix}}(\vec{\xi}|\mathbf{x}, \vec{u}^{(t)})$ for $T = 10^3$ iteration steps in Figure 7.1. Due to the stochastic nature of the algorithm, this target function does not monotonically increase, but it quickly converges to comparatively stable levels. For a smaller number of mixture components the convergence is faster, especially the two-component mixture converges within less then 100 iteration steps. In addition, the smaller the number of mixture components, the smaller is the fluctuation in what could be called “stationary phase” in an informal analogy to the Gibbs sampler.

It should be noted that the algorithm can be also restarted multiple times with different initializations. Whereas for a traditional EM algorithm restarts are absolutely necessary as it may get trapped in local optima, the situation is less critical here since the stochastic behavior of the algorithm attempts to avoid that. However, the success in leaving suboptima depends on the shape of the target function: if local optima are sharp peaks, even a stochastic algorithm may get trapped, as the probability of sampling substantially different latent variable assignments gets close to zero. Multiple restarts may thus be beneficial in order to

Table 7.1.: **Prediction performance of mixtures of PMMs**. The rows represent different numbers of mixture components and the columns represent the maximal order of the PMM. The table entries are the logarithmic predictive probabilities of the test data set. The optimal number of mixture component for each maximal model order is emphasized.

K	PMM(0)	PMM(1)	PMM(2)	PMM(3)
1	-28,883.7	-28,103.8	-27,775.4	-27,509.1
2	-27,978.0	-27,480.6	-27,196.2	-27,183.2
3	-27,681.1	-27,231.7	-27,147.8	**-27,152.7**
4	-27,527.0	-27,189.5	-27,143.5	-27,193.0
5	-27,383.2	-27,157.3	**-27,130.3**	-27,184.7
10	**-27,234.8**	**-27,123.4**	-27,182.8	-27,245.4

search through a larger fraction of the space of latent variables, but at least for the splice site data empirical tests have shown that restarting the algorithm R times while allowing each restart to run T iteration steps each does not yield a significantly better score than TR iteration steps for a single start.

Prediction performance

In a second study, we focus on the prediction performance of mixtures of PMMs using the combination of fNML-based joint structure score and fsNML parameter estimates. We learn mixture models of different number of components K on the splice site training data based on $T = 1000$ iteration steps for finding the optimal latent variable configuration and thus optimal model structures and probability parameters. Based on the optimal model, we then compute the logarithmic probability of the test data.

Here, we also include $K = 1$ for comparison, which represents the non-mixture case, where learning can be done analytically. In addition, we also compare to mixtures of PMMs of maximal order 0, 1, and 3. The maximal order of the PMM defines the search space of the structure learning algorithm, so there is no structure learning for PMM(0), which is equivalent to a simple independence model. We thus obtain a matrix of prediction values where the rows represent the number of mixture components and the columns determine the maximal order of the component PMMs (Table 7.1).

We observe that a simple independence model has the worst log prediction of $-28,883.7$, and increasing the degree of statistical dependence that is taken into account by increasing the maximal model order to 3 gradually improves prediction performance (first row). Since PMM(1)-PMM(3) infer the optimal PCT structures from data, overfitting is avoided. Using a mixture of simple independence models also gradually improves prediction performance up to $K = 10$ (first column).

The combination of using a mixture model and inferring the optimal structure within each component, yields even better performances, though. The largest improvement is – for all maximal model orders – achieved by using a two component mixture instead of a single distribution. Further improvements are generally small, with the optimum being a ten-component mixture of first-order PMMs.

The combination of many mixture components and complex component models decreases in some cases prediction slightly compared to the simpler alternatives, which might indicate that here overfitting w.r.t. the number of mixture components occurs. Another explanation for the poor performance of a large-component mixture of complex PMMs is the fact that the algorithm has more difficulties to find the optimal latent variable configuration, or at least a configuration that yields a joint structure score close to the optimum. Since we used $T = 1000$ iteration steps in all cases, combinations of many mixture components with putative complex models make the optimization problem more difficult.

7.2. Model selection within motif discovery

Promoter models for performing motif discovery are closely related to mixture models, so the same algorithmic idea as discussed in Section 7.1 can be applied as well. However, there are several small pitfalls that need to be considered and make a slight adaption of the algorithm necessary.

7.2.1. Algorithm

First, a standard ZOOPS model contains only one component model – the motif model – that is to be dynamically inferred via an iterative algorithm, as it is common to estimate the parameters of the flanking model from the entire data set and to keep them fixed during the iterations in order to save a substantial amount of computation time. We thus may consider it as a special case of a two-component mixture model, where structure and parameters of the second component are fixed and thus apply Algorithm 5.

Second, a ZOOPS model allows a sequence not to contain a binding site at all. This may cause a problem if the structure score of the motif model is used for evaluation of the latent variable configuration: Structure scores that assume the form of a penalized log-likelihood become larger when sample size is reduced. Hence, we obtain the best possible score when there is no data at all, which is a problem w.r.t. Equation 7.3. In the case of mixture models, there is always the same number of data points involved, and the latent variables determine the distribution of those data points among mixture components. For the ZOOPS model, however, we encounter the situation that the number of data points that influence the structure score may vary according to the realization of the binding site occurrence latent variables $\vec{u}$.

For that reason, we restrict the ZOOPS model by externally setting $\nu = 1$, resulting in an OOPS model. It may appear to be a severe restriction at a first glance, but there is little practical justification for estimating ν from data. In most practical cases, this parameter converges to $\nu = 1$ during the iteration anyway, since there are in each sequence of length $L \gg W$ at least a few subsequences of width W which have a higher likelihood for the motif model than for the flanking model, even though this likelihood may be orders of magnitude lower than that of "true" binding sites. On the other hand, not enforcing OOPS conceptually could result in paradox behavior. For example, if $\vec{u}$ are initialized using a uniform distribution for each variable, the very first iteration step would probably result in the highest joint structure score (Equation 7.2), provided that in the course of the algorithm ν converges to 1.

There is another difference from practical point of view. Motif discovery has – due to possible phase shifts of the motif – a highly multi-modal optimization landscape, i.e., the complete data likelihood and penalized variants thereof have several sharply peaked local optima. Hence, multiple restarts are here crucial for finding a latent variable configuration that contains the optimal phase shift of the motif.

For robust motif discovery using an OOPS model with a PMM as motif model, we thus propose Algorithm 6. Here $\mathbf{x}_{\vec{v},\vec{s}}$ denotes the set of binding sites from $\mathbf{x}$ that are given by latent variable configuration $(\vec{v}, \vec{s})$. The algorithm is general in the sense that arbitrary structure scores and parameter estimates, such as those from Chapter 3, can be plugged in.

Algorithm 6 Robust motif discovery algorithm for PMMs

```
for $r = 1, \ldots, R$ do
  for $t = 1, \ldots, T$ do
    for $i = 1, \ldots, N$ do
      sample $v_i^{(t)}$ from $\begin{cases} \left(\frac{1}{L_i-W+1}, \ldots, \frac{1}{L_i-W+1}\right) & \text{if } t = 1 \\ P(v_i|\vec{x}_i, \mathbf{\Theta}^{(t-1)}) & \text{if } t > 1 \end{cases}$
      sample $s_i^{(t)}$ from $\begin{cases} (\sigma, 1-\sigma) & \text{if } t = 1 \\ P(s_i|\vec{x}_i, \mathbf{\Theta}^{(t-1)}) & \text{if } t > 1 \end{cases}$
    end for
    for $\ell = 1, \ldots, W$ do
      compute $\tau_\ell^{(t)} = \underset{\tau_\ell}{\text{argmax}}\, S(\tau_\ell|\mathbf{x}_{\vec{v}^{(t)},\vec{s}^{(t)}})$
      for $c \in \tau_\ell$ do
        estimate $\theta_{\ell c}^{(t)}$ from $\mathbf{x}_{\vec{v}^{(t)},\vec{s}^{(t)}}$
      end for
    end for
  end for
end for
```

For each restart, the algorithm terminates depending on two parameters, T and T', when the expression

$$t \geq T \wedge \left(\max_{t' \in (t-T'+1 \ldots t)} S_{t'} \right) \leq \left(\max_{t' \in (1,\ldots,t-T')} S_{t'} \right) \tag{7.5}$$

is true, where S_t is a short notation for $S(\tau_{\ell}^{(t)} | \mathbf{x}_{\vec{v}^{(t)},\vec{s}^{(t)}})$. In other words, we run the algorithm at least T iteration steps, but require in addition that we did not observe an improvement of the score during the last T' iteration steps before termination. Using that strategy, we are capable of limiting the running time of the algorithm in most cases, while still ensuring that a reasonably stable score and thus motif is found.

7.2.2. Case study

In this section, we study the performance of the robust motif discovery algorithm using a PMM as motif model with BIC as structure score and fsNML for estimating the probability parameters in order to obtain a robust algorithm by eliminating any need for specifying prior hyperparameters. We evaluate the performance the algorithm on the CTCF data from Chapter 5, which we also used in Section 6.2 for evaluating the Bayesian model averaging via Gibbs sampling for the problem of motif discovery. We use $T = 50$ as minimal number of iteration steps per run of the algorithm, $T' = 10$ as the minimal number of non-maximal iteration steps before termination, and $R = 10$ restarts.

We perform a fragment-based classification in resonance with Section 5.2 and Section 6.2, and show the results in terms of sensitivity for specificity of 99% as well as area under the ROC curve in in Table 7.2. Both performance measures are here relatively independent of the initial model order, which is a consequence of using BIC as structure score, which does not learn substantially larger trees just because the initial model order is increased.

In comparison with the Bayesian model averaging approach from Chapter 6, and the extensive hyperparameter-tuning of Chapter 5, the algorithm yields a slightly decreased classification performance, but in most cases the difference between the different methods negligible compared to the gain that all methods yield in comparison to the simple PWM model that neglects intra-motif dependencies. We observe the only larger difference among all PMM methods when considering the sensitivity of the hyperparameter-tuned EM algorithm, which is about two percentage points higher compared to the other algorithms. This effect may be explained by the fact that the hyperparameter-tuning for the EM algorithm was performed by optimizing the sensitivity, hence it is not surprising that for this performance measure the hyperparameter-tuned method is superior to all other alternatives, whereas for the area under the ROC curve this difference is less pronounced.

While it may appear that the robust algorithm from this section is slightly inferior to the other variants, it should be noted that its results have been obtained within several minutes on a standard desktop computer. In contrast, the Bayesian prediction using the Gibbs sampling output required – depending of number of parameter samples used – hours to days for evaluating the classifier, whereas the EM algorithm for maximizing the posterior distribution required a high-performance cluster that is capable of running hundreds of jobs in parallel in order to perform hyperparameter-tuning via cross-validation.

Table 7.2.: **Classification performance of the of the robust motif discovery algorithm.** We show two performance measures for the fragment-based classification on CTCF data for three different orders of the PMM motif model. For comparison, we also show the results of the EM algorithm with hyperparameter-tuning from Section 5.2, the Gibbs sampler from Section 6.2, and a simple PWM model learned via EM, which already served as base for comparison in Section 5.2.

Measure	$d = 2$	$d = 3$	$d = 4$	EM($d = 4$)	GS($d = 2$)	PWM
Sensitivity	82.90%	83.00%	82.90%	85.20%	82.93%	71.20%
AUC-ROC	98.74%	98.86%	98.86%	98.94%	98.89%	97.45%

Bottom line

In this chapter, we discussed an approach to perform model selection in the setting with latent variables in order to complete the algorithmic work in order to achieve main objective I (Figure 1.11). The key idea is to combine sampling of latent variables, as already used in the Gibbs sampler, with a structure learning step that is – with given latent variable samples – then identical to the fully observable case. Hence, we obtain a strategy for dealing with latent variables that is widely independent of model classes and learning principles, but is heuristic in nature. Applying it to PMM motif discovery with BIC as robust structure score, we found that convergence of the algorithm is very fast. Even though the classification performance does not reach the levels of the heavily hyperparameter-tuned EM algorithm, such a slight decrease may be well acceptable, if it yields a dramatic speedup that makes a study for many different data sets possible. Such a large-study of intra-motif dependencies for a broad variety of different transcription factors, which was infeasible with the previously presented approached from Chapter 5 and Chapter 6, is the final topic of this thesis.

There is one thing even more vital to science than intelligent methods; and that is, the sincere desire to find out the truth, whatever it may be.

Charles S. Peirce

8

ENCODE ChIP-seq data analysis

In Chapter 5, we studied the phenomenon of intra-motif dependencies for the single, albeit very important, transcription factor CTCF. We found substantial evidence for intra-motif dependencies, but repeating such a study for a larger number of transcription factors was infeasible due to an overly resource-demanding EM algorithm, which required hyperparameter-tuning via cross-validation.

In the previous chapter we discussed an alternative learning algorithm for learning optimal PMMs under latent variables in order to replace the EM algorithm used in the CTCF study. This alternative algorithm has the advantage that arbitrary scoring criteria that exist for fully observable data can be applied, including the robust MDL-inspired methods from Chapter 3.

In this chapter, we now apply this algorithm for investigating intra-motif dependencies in the spirit of Chapter 5 for a large set of transcription factors. The main purpose of these case studies is to investigate (i) how prevalent intra-motif dependencies within binding sites are, and (ii) which order of dependency may be necessary and sufficient to take into account for classifying ChIP-seq fragments, both contributing to main objective II. Moreover, we intend to study (iii) which of the different scoring criteria is optimal for robustly learning PMMs within motif discovery, in order to complete main objective I.

8.1. Methods

In this section, we describe origin of the data we intend to analyze and the design of the performed case studies. We subsequently present and discuss the results in Section 8.2.

8.1.1. ENCODE ChIP-seq data

For the studies in this chapter, we use various ChIP-seq data set from the Uniform TFBS track of the ENCODE project [115, 120]. The individual data sets in this track were produced by different labs using different tools for mapping and peak-calling, but all data sets were processed based on the hg19 version of the human genome.

We use all data sets in the Uniform TFBS track that are available for the H1-hESC cell line, which comprises 50 different transcription factors. Those data sets consists of a list of ChIP-seq peaks, i.e., genomic fragments, identified by chromosome, start position, and end position, each being associated with an enrichment score, which indicates how strong (how likely) the binding of the transcription factor to that particular fragment is.

However, these enrichment scores are difficult to interpret quantitatively, and especially the comparison among different experiments for different transcription factors is hardly possible. As a consequence, the definition of a threshold for selecting a high-quality subset of all available fragments for motif discovery that is generally applicable to all data sets is difficult.

In order to cope with that problem, we reduce the information provided by the enrichment scores to a qualitative ranking. For each TF, we pick the top 20% of the available peaks and extract the corresponding sequences from the human genome in order to build a positive data set. Next, we extract for each positive fragment two sequences from randomly sampled locations on the same chromosome, provided they do not overlap with a positive sequence or contain ambiguous nucleotides, and compile them into a negative data set.

8.1.2. Classification study

In a first study, we perform for each transcription factor a fragment-based classification comparable to Chapter 5. For the sake of robust evaluation, we perform a ten-fold cross-validation, i.e., we divide positive and negative data set into ten subsets, and each of the ten subsets serves as test data set once, whereas union of the remaining nine subsets are the corresponding training data set.

We estimate Θ_{f}, the parameters of the flanking model of the OOPS model, which is here once more a second-order homogeneous Markov model, from the union of positive and negative training data set. Subsequently, we estimate Θ_{m}, the parameters of the motif model of the OOPS model, on the training data set by applying the stochastic algorithm from Section 7.2. We here use the same parameter values for setting the duration of the optimization, namely $T = 50$, $T' = 10$, and $R = 10$.

In this study, we compare different initial model orders and learning methods for the motif. We study PMMs of second, third and fourth order, and learn these models once with BIC and once with fNML as structure score. In addition, we include several models that do not require structure learning, such as the PWM model, WAM, and fixed order Markov models of order two and three. For all models we estimate the conditional probability parameters using fsNML in order to eliminate the necessity for setting external hyperparameters.

For all ten methods under consideration, we build a classifier that consists of an OOPS model for class one and a second-order homogeneous Markov model for class two, where the parameters of the homogeneous Markov model are identical to Θ_{f}. We classify all sequences in the test data sets and compute the area under the ROC curve given the true class labels.

In contrast to Chapter 5, we have now eliminated the necessity for hyperparameter-tuning, thus there is no internal cross-validation on the training data sets.

8.1.3. Binding site analysis

For predicting binding sites, we do not use any of the learned models from the cross-validation experiment, but learn a new model from the entire data set of ChIP-seq positives. The models and learning methods under consideration as well as the optimization parameter settings are identical to the classification study. However, another difference to the classification study is that the flanking parameters Θ_{f} are estimated only from the positive data, not from the union of positives and negatives. Hence, training a model and predicting binding sites utilizes only the positive data, which resembles applications where pure motif discovery with subsequent prediction of binding sites is of interest, and alternative models or learning approaches need not to be compared.

After having trained the motif model, we predict binding sites using the same threshold-based approach based on negative data as described in Section 5.3, and we also use the threshold of 10^{-4}.

We subsequently use these predicted binding sites for plotting a sequence logo and the mutual information of different orders among adjacent positions in the motif as exemplified in Figure 5.4 for the case of CTCF.

8.2. Results and discussion

In this section, we present and discuss the results for the previously described studies using ChIP-seq data sets for 50 different transcription factors. We compare PWM model, WAM, and fixed-order Markov models of second and third order with PMMs. For the rest of this section, we slightly deviate from standard notation by denoting a PMM of order d learned by BIC as BICd , and a PMM of order d learned by fNML as fNMLd .

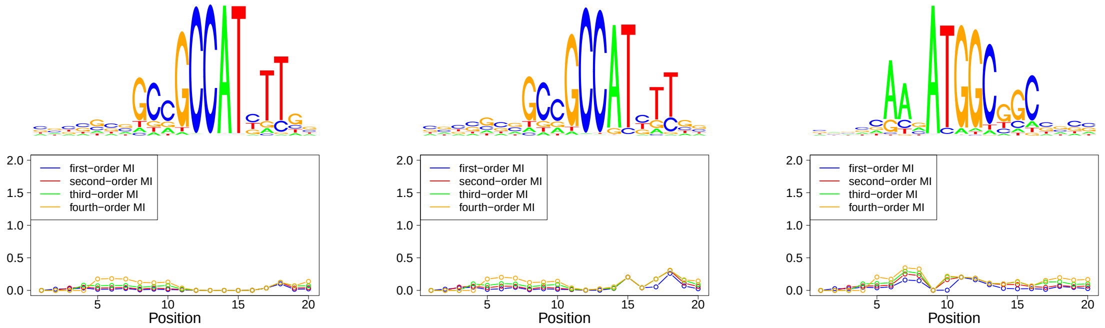
8.2.1. Categorization of data sets

ChIP-seq experiments produce a set of sequences, each of which has a binding affinity of possible different strength towards a specific TF of interest. However, this does not necessarily imply that there is a specific binding motif for this TF within these sequences. There may be cases, in which a TF is only indirectly connected to the DNA molecule due to binding other DNA-binding proteins. Moreover, there are basal TFs that bind unspecifically but are required to form the transcription initiation complex. In these cases, we may observe a ChIP-seq data set that does not contain a clear sequence motif, and it would be questionable to include these data sets into a systematic comparison of models and learning approaches.

For that reason, we do not begin to compare all approaches by classification performance, but investigate the predicted binding sites and to analyze the predicted motifs first. We use both sequence logos and mutual information plots in order to categorize each data set according to the quality of motif discovery.

Category A: Distinct motif identified

The first category contains data sets, where all approaches that can be expected not to be completed overfitted (PWM, WAM, BIC2, BIC3, BIC4, and fNML2) predict a similar consensus sequence in the sequence logo, subject to possible phase shifts or reverse complementary strand orientation. These sequence logos may still vary among approaches, but a visual inspection and comparison with known sequence logos from Factorbook [1] or Jaspar [28] confirms that the correct motif was identified through motif discovery.



(a) PWM model (b) BIC2 (c) BIC4

Figure 8.1.: **Sequence logos and mutual information plots of a typical category A data set (YY1) for different motif models.** The sequence logos are qualitatively similar (the BIC4 approach has learned the motif in reverse complement strand orientation), yet the mutual information indicates the presence of additional intra-motif dependencies.

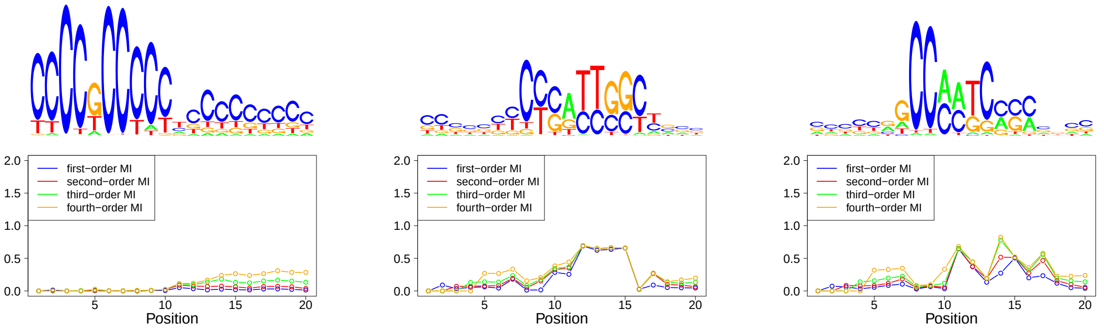
One example for a category A data set is YY1, for which we plot sequence logos and mutual information plots for three different approaches in Figure 8.1. In total, 30 of 50 data sets fall into category A, and these are thus perfectly suitable for comparing different models that have the capability of further refining the – on the first glance – similar motif.

Category B: Evidence of multiple motifs

The second category comprises data sets that appear to contain multiple motifs. The main indications are substantially different predicted sequence logos for i) the PWM based prediction and ii) predictions using intra-motif dependencies, in combination with high mutual information values compared to category A data sets.

---

[1] http://www.factorbook.org



(a) PWM model (b) BIC2 (c) BIC4

Figure 8.2.: **Sequence logos and mutual information plots of a typical category B data set (SP1) for different motif models.** There are indications for the presence of multiple sequence motifs in the data.

One example for a category B data set is SP1 (Figure 8.2). Here, we find a `CCCCGCCCC`-consensus sequence by prediction using the PWM model, which is typical for all TFs of the SP-family. However, when dependencies are taken into account, the predicted sequence logos look substantially different. The PWM-based prediction yields only 529 predicted binding sites (from 3009 ChIP-seq positive sequences), whereas the BIC2, for instance, yields 1430 predicted binding sites. Moreover, the mutual information reaches values of almost 1 bit at several positions.

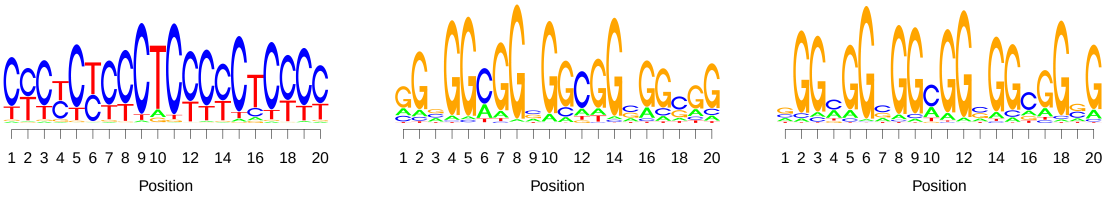
These observations indicate that at least two different sequence motifs have been pressed into one motif model. In these cases, it is no surprise that taking into account intra-motif dependencies improves classification performance, but these improvements could be considered as artifacts, resulting from a overly simple promoter model. In total, 6 of the 50 investigated data sets show a similar behavior and are thus not suited for a fair investigation of intra-motif dependencies.

Category C: Repetitive structure identified

The third category contains data sets where motif discovery identified only repetitive structures, but no reasonable sequence motifs in the traditional sense. Examples for such data sets are shown in Figure 8.3. Whereas these logos result from a PWM-based prediction, the results look similar for all other approaches. In total, we find 14 of 50 data sets that show comparable results.

While there may be transcription factors that actually do bind a long repetitive structure, the identification of a repeat is also a typically result that is obtained when the data set contains no other specific motif. Since we use a generative learning approach, the algorithm then simply identifies an overrepresented feature that is not directly covered by the flanking model. Here, we often find third-order repeats, which are not covered by the second-order flanking model.

In many cases, it is not surprising that the algorithm did not identify a distinct motif if we consider the functionality of the corresponding TF. POLR2A, for instance, is simply the RNA polymerase that is involved in every transcriptional activity, and the same applies to TBP, the TATA-binding protein. CTBP2, on the other hand, is a CT-binding-protein, and we find its name to be quite accurate indeed, as we identify a CT-rich region as overrepresented feature in the data. Nevertheless, unspecific CT-rich regions would probably rarely be considered as sequence motif.



(a) CTBP2 (b) TBP (c) POLR2A

Figure 8.3.: **Sequence logos of several category C data sets.** We find repetitive structures that can hardly be considered as TFBS.

8.2.2. Classification results

In the previous section, we observed that among the 50 data sets three types of results can be qualitatively distinguished. Now, we conduct a quantitative analysis by investigating the results of fragment-based classification. The complete results, i.e., the AUC values for each learning approach and each data set, separated among categories, are shown in Appendix Table C.1 for category A, in Appendix Table C.2 for category B, and in Appendix Table C.3 for category C. The corresponding standard errors for all data sets are shown in Table C.4. In this section, we discuss some aggregate statistics of these results in order to draw some general conclusions.

Absolute averaged AUC values

First, we plot the averaged AUC values over all data sets, and over all data sets within each category (Table 8.4). For most approaches, we observe the category A data sets to achieve the highest average AUC. An exception are the approaches that use very complex models, namely third-order Markov model, and fNML-learned PMMs of third and fourth order. Here, category B data sets have the highest average AUC, which can be explained by the capability of complex models to accurately take into account the features of more than

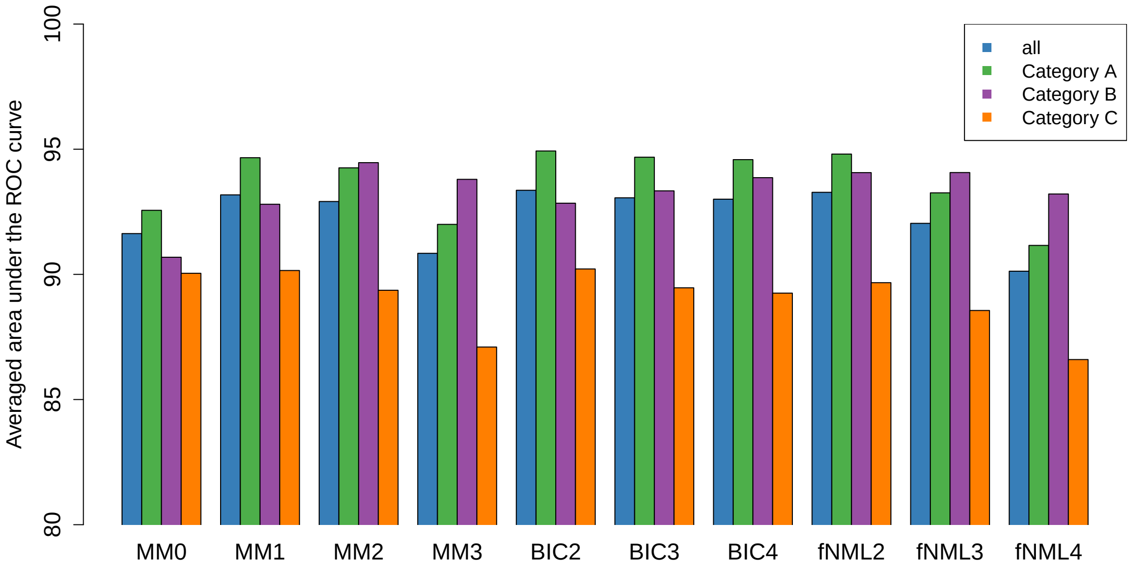


Figure 8.4.: **Averaged AUC values**. We plot the average AUC over all data sets and subgroups thereof according to the categorization of Section 8.2.1.

one motif at once. On the other hand, in case of only one dominant motif (Category A), these three approaches appear to yield overfitted motifs, since the classification performance decreases here.

Category C data sets always yield the worst classification performance, which is not surprising since we observed that in these datasets apparently no specific sequence motif exists. Nevertheless, classification is still possible and even for these data sets we yield average AUC values above 85%, since even the identification of repetitive structures that are frequent in promoter regions, may already make a classification possible.

For category A, BIC2 yields the highest average AUC, followed by MM1, BIC3, BIC4, and fNML4, and the difference among these methods is fairly small. The fact that BIC2, BIC3, and BIC4 yield almost identical average AUC values indicates that BIC is once again a robust choice, which yields good classification results almost independent of initial model order. However, the fact that MM1 is also in the same range as these models also indicates that nearest-neighbor dependencies might be dominant in the data.

Relative improvements to PWM model

The previous study has one drawback, namely AUC values of different data sets are not directly comparable, as the classification problem is different for each data set. Different data sets are of different size, and the putative motif may be of different length. With increasing data set size and increasing motif length, we expect the classification performance to increase, and CTCF (Chapter 5) is an example where both data set size and motif length are comparatively large, leading to a good classification with AUC values reaching almost 99%. For other transcription factors, however, this value may be significantly lower, even if the correct motif has been identified. Hence, averaging over all AUC values for one model may be heavily influenced by these data sets, which yield comparatively low AUC values.

We use the classification experiment for comparing different models, so we are mainly interested in the relative difference between two models. Let AUC_0 denote the area under the ROC curve for model 0, and AUC_1 denote the area under the area ROC curve for model 1. We compute

$$\Psi = \log_2 \left(\frac{1 - \mathrm{AUC}_0}{1 - \mathrm{AUC}_1} \right), \tag{8.1}$$

which measures the improvement of model 1 in relation to model 0. The idea behind this score is not to compare absolute values of performance measures, since they are not comparable from data set to data set anyway. Instead, we compute the difference to a perfect classification for both approaches, with subsequently computing their ratio. The binary logarithm is then taken to ensure that positive values indicate an improvement of model 1 over model 0, and negative values indicate the opposite. Obtaining $\Psi = 1$ thus indicates that using model 1 reduces, in relation to model 0, the distance to a perfect classification by a factor of 2.

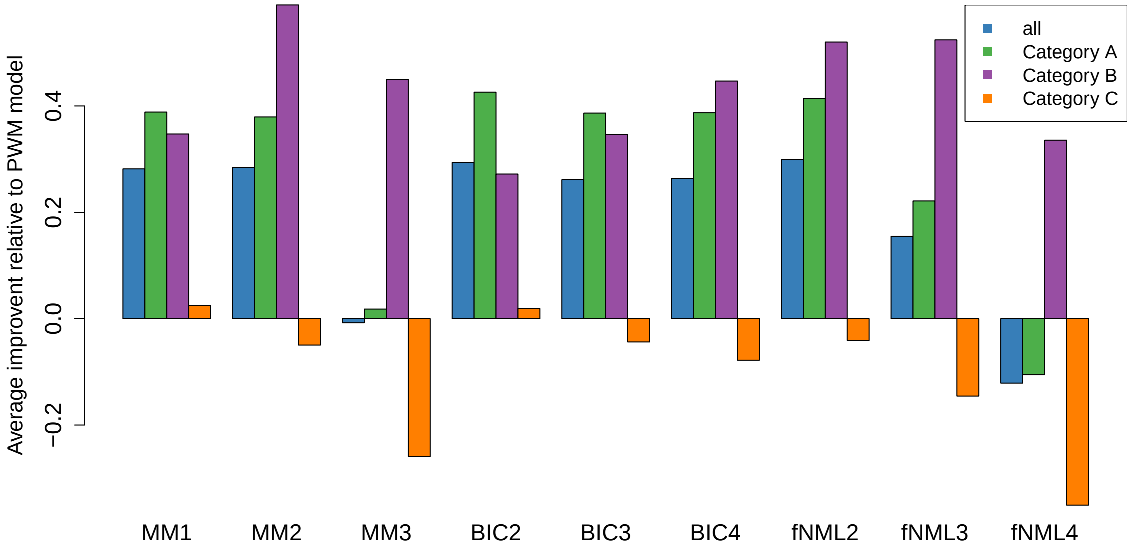


Figure 8.5.: **Averaged Ψ values for different approaches**. Here, reference model 0 is the PWM model.

We compute Ψ for all approaches and all data sets with the PWM model being the reference model 0, and we display the averaged results in Figure 8.5. We observe the results to be generally in accordance with the previous study. Apart from category B data sets, for which we know that strong dependencies as multiple-motif artifact exist, the approaches using MM3 and fNML4 yield a clearly overfitted model, since we observe no improvement and mostly even a decrease in classification accuracy in relation to the PWM model. For BIC4 we do not observe this behavior, so BIC is the only approach that is comparatively robust against the choice of the initial order model order. We also observe that for the relevant category A, the BIC2 method is slightly superior to all other approaches in comparison, which is also in accordance with the absolute averaged AUC values shown in Figure 8.4.

Data set specific improvements

In the previous section, we studied the average improvement of different approaches in relation to the PWM model. Now we take a more fine-grained view and investigate these improvements separately for all data sets of category A.

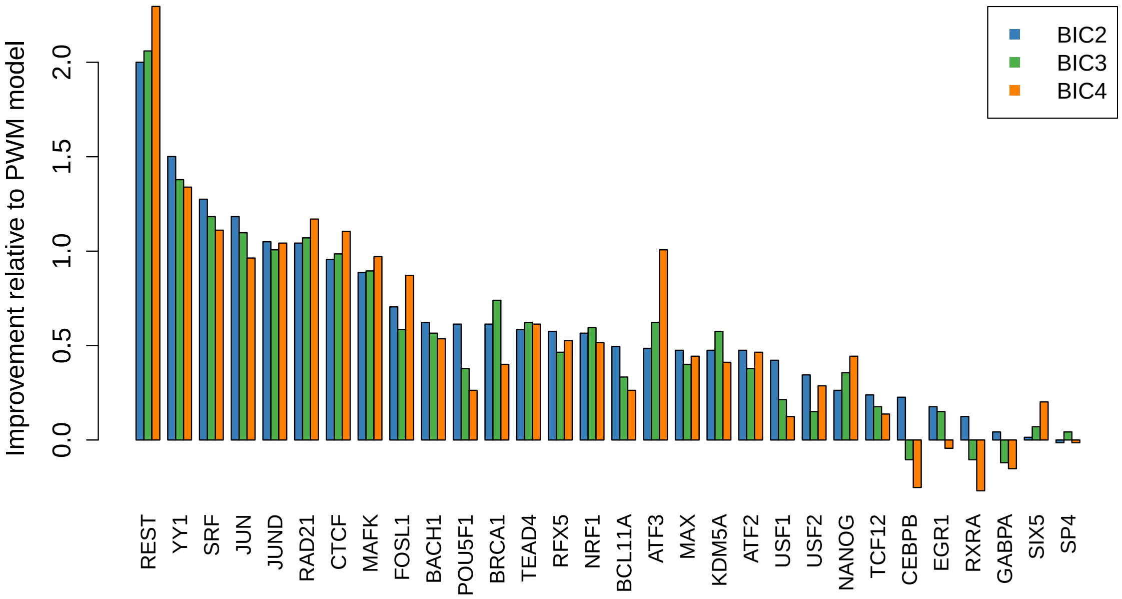


Figure 8.6.: **Data set specific Ψ values based on PWM model reference.** We show the relative improvements of modeling dependencies with PMMs of different order, which use BIC as model selection criterion, compared to PWM model.

In Figure 8.6, we plot Ψ for the three BIC methods in relation to the PWM model. The results are ordered according to the BIC2 performance, which is the best performing approach on average. We observe that BIC2 yields for all but one data set (SP4) a classification that is superior to the PWM model, and for SP4 the decrease in classification is almost negligible. Hence, modeling dependencies with second-order PMMs learned with BIC is always at least as good as using a PWM model and in most cases it yields a substantial improvement.

In addition, there are many cases in which PMMs of higher order increase classification performance. For BRCA1, NRF1, and KDM5A, a third-order PMM is the best choice, even though the difference to the second-order PMM is rather small. More substantial improvements can be achieved for some data sets when increasing the initial model order to four. Examples are REST, RAD21, CTCF (which is in accordance with the finding from Chapter 5), MAFK, FOSL1, ATF3, NANOG, and SIX5. However, in some cases using a fourth-order PMM also decreases classification, even when it is learned using BIC. Examples are POUF1, BCL11A, and data sets where the difference to the PWM model is comparatively small, such as CEBPB, EGR1, RXRA, and GABPA.

This slight overfitting behavior may not be attributed solely to the scoring criterion, since the larger models have also a certain disadvantage in the sense that the search space for the optimal model is larger. Since we allowed all methods to use the same number of iteration steps and restarts, models that have a smaller search space may have slight advantages. It is possible that the fourth-order PMMs catch up to the second-order PMMs, once we invest more computation time, but in the spirit of this section it then is debatable whether investing substantially more resources is truly worth the effort.

We perform a second data set specific study for both fixed-order Markov models and PMMs of different order that are learned with fNML as structure score, and show the resulting plots in Appendix Figure C.3 and Appendix Figure C.4. We observe for many data sets a sharp decrease in classification performance when the model orders increases, which is a clear sign that overfitting occurred. This effect is particularly evident when comparing MM2 to MM3 and fNML3 to fNML4, where attempting to model third- and fourth-order dependencies with MM3 and fNML4 often even leads to a decreased performance in relation to the PWM model. Apart from the transition from MM1 to MM2, there is almost no case in which an increase of the model order leads to increased classification performance. While for fixed-order MM this effect is to be expected at a certain order, it may have not been totally obvious for fNML, which only yields acceptable classification when the sample size is very large (REST, RAD21). Here, the problem cannot be dealt with just by investing a larger number of iteration steps, since the estimated model structures on small data sets are simply overly complex.

Finally, we compare BIC-learned PMMs to the WAM, since that model also appears to be a quite robust choice, taking into account first-order dependencies among adjacent nucleotides, yet not using too many parameters. We thus plot Ψ for BIC2, BIC3, and BIC4 in relation to the WAM for all data sets of category A in Figure 8.7. The data sets are here ordered according to the Ψ value of BIC4, for which we observe the data sets to partition in two groups of similar size: For the first group, using BIC4 is clearly better than using the WAM, which indicates that modeling higher-order dependencies exist for the corresponding TFs, and utilizing them improves the fragment-based classification. For the second group, BIC4 decreases classification performance in relation to the WAM, which indicates that either no higher-order dependencies exist, or the amount of available data is not sufficient to effectively utilize them. In case of BIC2 and BIC3, the discrepancy is somewhat smaller, yet there are eight data sets for which none of the BIC-methods outperforms a WAM, and in some cases (KDM5A, GABPA, JUN, CEBPB) there is strong evidence that taking into account higher-order dependencies does decrease classification performance.

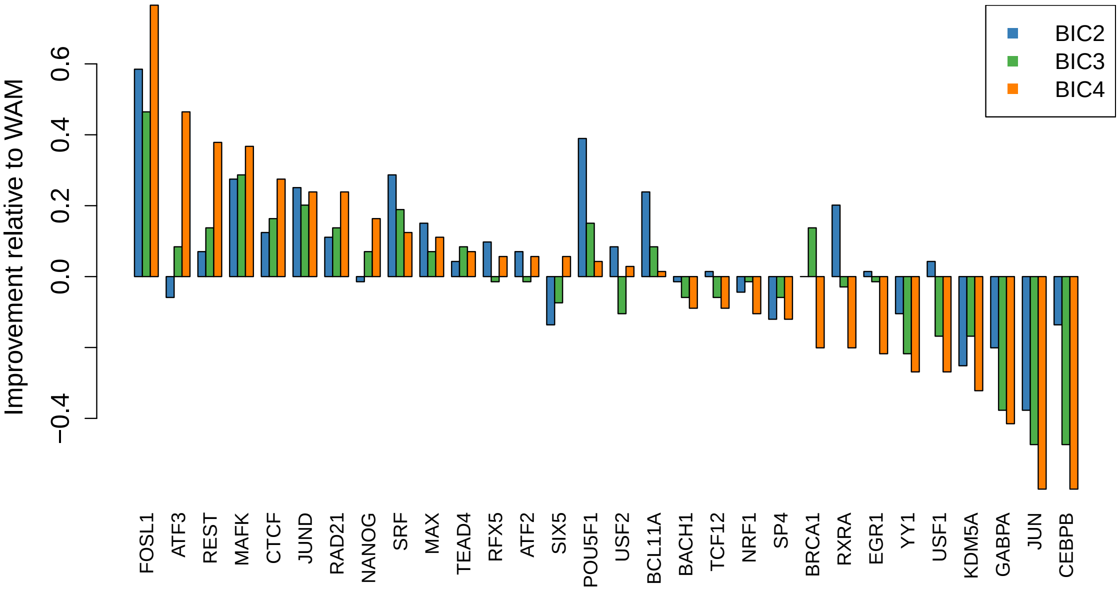


Figure 8.7.: **Data set specific Ψ values based on WAM reference.** We show the relative improvements of modeling dependencies with PMMs of different order, which use BIC as model selection criterion, compared to a WAM.

Significance tests

In the previous sections we learned that the different methods yield different AUC values for individual data sets. Here, we study whether the mean AUC values of two methods within the category A data sets are significantly different. To this end, we apply the Wilcoxon signed-rank test [121], which is a paired difference test.

We show p-values for comparing different methods with the null hypothesis of their means AUC values being identical in Table 8.1. Assuming a significance level of 1%, we observe that the results from the previous sections are generally confirmed: Most methods that take

Table 8.1.: **p-values of Wilcoxon signed-rank test comparing the distribution of AUC values of category A data sets among different methods.** When two methods yield not a significantly different distribution, the corresponding p-value is displayed in bold.

	WAM	MM2	MM3	BIC2	BIC3	BIC4	fNML2	fNML3	fNML4
PWM	1.3E-7	3.1E-5	**0.557**	2.5E-6	1.6E-5	1.2E-5	1.5E-5	**0.054**	**0.046**
WAM		**0.583**	4.7E-7	**0.163**	**0.703**	**0.926**	**0.411**	2.5E-4	8.0E-8
MM2			1.8E-9	**0.096**	**0.634**	**0.674**	**0.041**	1.2E-4	1.3E-8
MM3				3.8E-7	6.1E-8	9.9E-7	2.4E-6	2.3E-4	1.4E-3
BIC2					**0.031**	**0.034**	**0.472**	1.8E-4	1.8E-4
BIC3						**0.405**	**0.210**	5.1E-4	5.1E-4
BIC4							**0.073**	8.6E-4	1.3E-7
fNML2								5.7E-6	4.6E-8
fNML3									1.1E-5

into account dependencies while not using overly large models (WAM, MM2, BIC2, BIC3, BIC4, fNML2) show a significant difference to the PWM model, and from Figure 8.4, we know that this difference is actually in favor of these methods. Among the methods that yield a significantly better classification than the PWM model, however, the differences are generally not significant, which we could expect already from the data set specific comparison of BIC2-4 with the WAM model in Figure 8.7. However, in some cases the difference is close to be significant, and with choosing a more liberal significance level, such as 5%, we would conclude that BIC2 yields significantly higher mean AUC values than BIC3 and BIC4, for instance. One explanation for the slightly decreased performance of third- and fourth order models despite being learned with BIC is the following. Increasing the initial model order has the consequence that the total space of possible candidates increases. While BIC behaves well in the sense that it learns sparse PCTs, there is still the danger of learning a structure based on some random features present in the training data that are not prevalent in the test data. The smaller the data set and the larger the space of possible PCT structures to choose from, the more likely is this phenomenon to occur, and in some sense this behavior might be inevitable. The desired behavior of a more complex procedure, such as an increased maximal depth of PCTs, is to perform in all cases at least as good as the simpler alternative. The apparent deviation from this expectation can be explained by the fact that structure learning is more sensitive to random noise in the data when the initial order of PCTs, and thus the sheer number of possible structures, increases. Since the probability of choosing a suboptimal PCT grows with the number of possible PCTs to choose from, it might be very difficult to overcome this problem and devise a robust method that yields for all data sets optimal results.

Bottom line

Summarizing all results, we may now answer the three questions raised at the beginning of this chapter and thus achieve both main objective I and main objective II (Figure 1.11). Considering the investigated TFs, intra-motif dependencies are not an exception but rather the rule, which justifies the use of statistical models that are more complex than a simple PWM model. This finding is interesting from biological point of view, as it suggests that either binding energies of nucleotides to the protein of interest are either not completely additive or that other features than pure protein-nucleotide binding affinities are of importance. However, first-order dependencies among directly adjacent nucleotides are dominating, whereas higher-order dependencies are less pronounced, and at least considering average statistics over all transcription factors, there is no significant difference between models of higher order and reasonable complexity. Nevertheless, higher-order dependencies do exist for several transcription factors, and if these features are to be utilized, then a PMM learned with BIC as scoring criterion is the preferable choice. Using BIC, the choice of the initial model order d is, in comparison to fNML, not crucial w.r.t. to performance, even though there is a slightly negative effect, which is on the edge of being not significant, though. While it thus may appear to be a good advice to choose d as large as possible in order not to miss any possible dependencies in the data, it does have an effect on the time complexity of the structure learning algorithm. For practical applications involving TFBS, $d \leq 4$ should be sufficient, and learning PCTs of that depth over a four-letter alphabet is completely feasible, i.e., requiring less than a second on standard desktop computers.

Life is the art of drawing sufficient conclusions from insufficient premises.

Samuel Butler

9

Conclusions

In this thesis, we discussed methods for statistical modeling of functional oligonucleotides, such as transcription factor binding sites or splice sites.

In this chapter, we summarize the key results of the presented work (Section 9.1), and discuss several new questions that did arise in the course of this work and might be addressed in future projects (Section 9.2).

9.1. Summary and lessons learned

This work was mainly motivated by and thus based on two hypotheses.

On the one hand, we assumed that independent relative nucleotide frequencies, as represented by a PWM model and graphically displayed as sequence logo, are not a sufficient representation of sequence motifs. We speculated from biological point of view that substantial statistical dependencies among adjacent nucleotides within binding sites do exist, and that these features can be helpful for recognizing binding sites, enhancing prediction and classification performance.

On the other hand, we assumed existing approaches, i.e., combinations of models and learning principles, that into account statistical dependencies to overparameterized, suffering from overfitting. We speculated from a computational point of view that recently proposed parsimonious context trees could serve as an utility to design a model class that is capable of taking into account intra-motif dependencies and that it is possible to learn such models efficiently from data.

At the end of Chapter 1, we formulated both hypotheses as questions in order to motivate both main objectives of the thesis. In the course of this thesis, we were able to answer those questions, i.e., we found substantial evidence that supports both hypotheses with only slight reservations. We summarize the main biological and computational results in following two sections, beginning with the latter.

9.1.1. Computational aspects

By combining inhomogeneous Markov models with parsimonious context trees, we introduced inhomogeneous PMMs as a model for DNA sequence patterns and proposed an appropriate prior distribution for that model class that enables a Bayesian learning approach. Using Bayesian model selection and parameter estimation via the mean posterior principle on an exemplary data set, we demonstrated that PMMs are capable of yielding – due to their higher structural flexibility – a better prediction than special cases such as fixed order and variable order Markov models. However, we found evidence that a uniform structure prior might not an optimal choice for obtaining models of appropriate complexity, making it necessary to apply an additional tuning step for the structure hyperparameter κ, which we accomplished by internal leave-one-out cross-validation on the training data. As a consequence, the learning approach was not truly Bayesian model selection, but rather model selection via cross-validation, even though the Bayesian approach narrowed down the superexponentially large search space of possible PCTs to a comparatively small number of possible values for κ.

Since this hyperparameter-tuning is not an entirely satisfying approach, we investigated robust alternatives to the Bayesian approach that are mainly motivated by the MDL principle and require no explicit prior specification. Approximations of the NML distribution, which have been original proposed for Bayesian networks, namely fNML as structure score and fsNML as parameter estimate appeared to be reasonable alternatives at first glance, and we demonstrated that the combination of both methods is indeed a robust alternative for the Bayesian approach that eliminates the necessity of specifying a parameter prior. However, fNML as structure score shows a similar behavior to the Bayesian marginal likelihood in the sense that for small data overly complex models are chosen, which is an undesired behavior for DNA sequence patterns, where, in the case of doubt, a simple model that has not substantially more parameters than a PWM model should be favored. We observed that the so called Bayesian information criterion, which also has an interpretation as MDL two-part code, does not show this behavior, which makes it the superior structure score when maximal model complexity is large in relation to the sample size. Here, the typical underfitting behavior of BIC actually becomes an advantage, as it is – in the case of PMMs – more important to learn sparse models that yield an adequate prediction, rather than attempting to identify a true, data-generating model, which is the incentive behind the Bayesian approach with a uniform structure prior. For fully observable data we may thus answer the question of how PMMs should be learned by the simple advice: Provided there is no prior knowledge available, BIC as structure score and fsNML for estimating the probability parameters is the preferable choice for learning a statistical representation from a pre-aligned set of training sequences of identical length.

When data is not fully observable, as in the case of mixture models or de novo motif discovery, the learning task is more challenging. Parameter learning in the presence of latent variables cannot be carried out analytically, so iterative algorithms such as the EM algorithm have to be resorted to. Structure learning is here a little explored field, and while the EM technique can be generalized to models with variable structure in principle, it yields reasonable results only in the special case where all candidate models have the same dimensionality. In the case of PMMs, however, the candidate structures do represent parameter spaces of different size, and we discussed that applying an EM algorithm here is problematic. Seeking a maximum a posteriori estimation of the combination of both structure and probability parameters corresponds asymptotically to model selection via maximum likelihood, which always picks the maximal model structure, provided the candidates are nested. The MAP structure score is thus not consistent, so observing more data does not guide us closer, but actually farther away from any true model. As a result, hyperparameter-tuning via cross-validation is here not just needed for counter-balancing small sample sizes as in the case of fully observably data, but it has to be an integral part of the learning scheme. Since each holdout of the cross-validation requires running an EM algorithm, this procedure becomes extremely time-consuming and cannot be carried out without massive parallel computing. As a consequence, we studied in alternatives for dealing with variable structures in the presence of latent variables.

One alternative to model selection, and thus relying for the task of prediction on one single model that was estimated from training data, is model averaging, i.e., performing a fully Bayesian prediction by integrating over the entire posterior distribution of all members of the model class given the training data. In the case of latent variables this cannot be achieved analytically, but the technique of Gibbs sampling provides an elegant method of generating samples from the posterior, which does not have the theoretical problems that seeking the maximum of the posterior entails. We thus derived a Gibbs sampling algorithm for mixture models and promoter models using PMMs and evaluated the approach in comparison to the EM algorithm. We observed that a fully Bayesian prediction, which uses a series of parameter samples from the posterior, can be superior to the maximizing approach when the number of observed data points is limited. Moreover, a uniform structure prior is here among the best possible choices, which eliminates in cases when maximal model order is small in relation to sample size the need for cross-validation for hyperparameter-tuning. However, the limiting factor for the Bayesian approach is the number of parameter samples that the integral over the posterior for prediction is approximated with. While generating those parameter samples can be achieved in a reasonable amount of time, the Bayesian prediction itself is limiting its own applicability.

We thus considered the problem of model selection in the presence of latent variables again, with the goal of performing robust structure learning without relying on cross-validation. Inspired by the Gibbs sampler and the results for robust learning for fully observable data, we proposed a stochastic algorithm in order to reduce the latent variable problem to the fully observable case. Thereby, we generalized the structure learning problem to models involving latent variables, independent of learning principles, whether it may be a Bayesian or an MDL approach. As such, we also discussed the first attempt to carry out NML-style model selection in the presence of latent variables, which has never been studied before. A drawback of the proposed approach is that such a stochastic algorithm is heuristic by nature and does not even guarantee local optimality. Due to that fact, we empirically investigated the convergence and prediction performance of the algorithm for learning mixtures of PMMs from splice site data and for motif discovery for TFBS data. In both cases, we observed quick convergence, and prediction and classification performance that is almost as good as for the very time-consuming approaches discussed in the previous chapters, which are outperformed w.r.t. speed by several orders of magnitude.

9.1.2. Biological aspects

From biological point of view, we did intend to answer the question about the prevalence and nature of dependencies within DNA binding sites. In order to address this question, we relied on the assumption that the existence of dependencies should yield an improved prediction performance, provided statistical methods are capable of utilizing them properly. Hence, we interpreted an improved prediction or classification performance of a method that attempts to utilizes intra-motif dependencies over a method that neglects them as indirect evidence for the existence of intra-motif dependencies.

In an initial study, we investigated the binding motif of the important human enhancer-binding insulator protein CTCF by performing a motif discovery on ChIP-seq data using a PMM as motif model. We evaluated the performance by a fragment-based classification approach and found that taking into account intra-motif dependencies up to order four improves motif finding capability substantially. Quantifying the intra-motif dependencies in a position-specific manner by measuring the mutual information between a position and its predecessors in the sequence revealed that dependencies are not equally strong along the sequence. Instead, the strongest statistical dependencies are located at two positions at the 3' end of the CTCF motif. We also proposed conditional sequence logos as a method for visualizing the conditional probability distributions the PMM is based on, and we exemplified this visualization on the two positions identified to show the strongest dependencies according to the mutual information. From the case studies on CTCF, we concluded that modeling of intra-motif dependencies using PMMs can be worthwhile indeed and that the view of a PWM model on the data can be overly simple, neglecting important information.

In a second study, we used the fast and robust stochastic algorithm developed in this thesis, which makes it possible to conduct a large-scale study of intra-motif dependencies inferred by PMMs, in order to investigate whether the dependencies found for CTCF are a biological curiosity, or a common feature of TFBS. We studied 50 ChIP-seq data sets of human DNA-binding proteins from the ENCODE project and found that those data sets can be can be categorized according to motif discovery results into three groups, for which only one is useful for an unbiased and fair investigation of intra-motif dependencies. For this group, which comprises the majority of the transcription factors under consideration, we consistently identified one motif and found intra-motif dependencies to be abundant. However, first-order dependencies appear to be the dominant feature, and higher-order dependencies can be observed to be beneficial in motif discovery only for about half of those data sets.

9.2. Possible future projects

While we answered the initial questions that were the main motivation for the topic of this thesis, several new questions did arise that could be addressed in future work.

9.2.1. More sequence analysis

From the bioinformatics point of view, or rather DNA sequence analysis to be more specific, there are several directions that could be pursued.

First, we observed that the uniform structure prior is often not the optimal choice when the task is prediction rather than finding a true model structure. In the course of this work, we found that eliminating the structure prior by making use of the simple BIC structure score instead of the Bayesian marginal likelihood does somewhat solve this problem for model selection. However, for Bayesian model averaging (Chapter 6) the dependency on the structure prior remains, and while we could cope with the problem by testing some candidates for the structure prior hyperparameter κ, this approach is unsatisfactory. As a result, studying different functional forms of the structure prior, that give more weight to the few existing sparse tree structures in expense of the vast number of large structures could be fruitful, although it may be far from trivial to obtain a robust prior that mimics the behavior of BIC.

Second, we found evidence for the presence of multiple motifs in ChIP-seq data of some transcription factors (Chapter 8). Since the high intra-motif dependencies observed in those cases appear to be an artifact of the presence of multiple motifs, we here excluded these data sets from further analysis. However, these cases are an incentive to extend the promoter model from the simple OOPS model that allows a single motif to more complex variants. A first step could be to allow a mixture of PMMs to be motif model, while still keeping the OOPS assumption, but further generalization leading to a complex promoter model that allows multiple binding sites of multiple motifs per sequence [38] would be of great interest. Using such a complex promoter model would enable to study which dependencies are caused by the mixture of two substantially different motifs, i.e., substantially different consensus sequences as in the case of SP1, and which are just position-specific variants as in the case of the CTCF motif.

Finally, all real data investigated in this work originate from the human organism. While human biology is certainly of great interest due to the close connection to medicine, the application of the methods presented in this work to data of other organisms would be interesting as well, in order to obtain a broader picture. The methodology as such is general and makes no assumptions that are specific to humans, but whether intra-motif dependencies exist for other organisms as diverse as yeasts, plants, or insects is still to be investigated.

9.2.2. Machine learning

While this thesis is mainly concerned with bioinformatics, it did benefit a lot from input from the machine learning community. So it is only consequent that several ideas for future work did arise that do not directly pertain any biological application but rather tackle more general machine learning problems.

In Chapter 2, we introduced inhomogeneous PMMs as a statistical model for sequence patterns that allows position-specific CPDs and thus makes position-specific use of PCTs. As we know that inhomogeneous Markov models can be perceived as special case of Bayesian networks, the utilization of PCTs to define *parsimonious Bayesian networks* is somewhat obvious. As we learned in Chapter 8 that for most TFBS data sets first-order dependencies among next neighboring nucleotides are dominant, the application of parsimonious Bayesian networks as a motif model for DNA binding sites is probably not overly fruitful, but as a general refinement of Bayesian networks for applications that suffer from limited data such a model class could be of great interest. In a similar spirit, *permuted parsimonious Markov models* could serve as an alternative to permuted variable length Markov chains [76].

We presented the idea to perform model selection in the presence of latent variables in Chapter 7 in an abstract setting for arbitrary model classes and learning principles. While the case studies in this thesis pertain only to PMMs, it might be interesting to consider different applications of the proposed stochastic algorithm. A possible future project could involve learning mixtures of arbitrary models with variable structure of different dimensionality outside of sequence analysis, and a possible starting point could be to relax the idea of mixture of Bayesian trees [104] towards mixtures of Bayesian forests.

In addition, it could be worthwhile to investigate other possibilities to solve the optimization problem of Equation 7.3. In this work, we used a stochastic algorithm that is inspired by the Gibbs sampler and thus iteratively samples in the space of latent variables in order to find a good latent variable configuration. A possible alternative is to apply generic algorithms that perform a (local) optimization in a discrete state space, i.e., in the space of latent variable configurations, such as hill climbing or simulated annealing. While the convergence of the stochastic algorithm is quick enough for practical application PMMs on the data sets studied in this thesis, other learning approaches could be beneficial for different models or entirely different types of data.

One problem that we did not address in this work at all is the algorithmic problem of finding optimal PCTs. While the dynamic programming algorithm [71] is feasible for the four-letter DNA alphabet and relatively small tree-depths, which are completely sufficient for modeling sequence patterns, the application of PCTs to different problems is still hampered by the unfavorable time complexity for learning PCTs w.r.t. alphabet size and tree depth. Here, significant algorithmic advancements are required in order to achieve a broader applicability of all possible model classes that make use of PCTs.

Bibliography

[1] R. Eggeling, A. Gohr, P.-Y. Bourguignon, E. Wingender, and I. Grosse, “Inhomogeneous parsimonious Markov models,” in *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013*, vol. 1, pp. 321–336, Springer, 2013.

[2] R. Eggeling, T. Roos, P. Myllymäki, and I. Grosse, “Robust learning of inhomogeneous PMMs,” in *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 33 of *JMLR Workshop and Conference Proceedings*, 2014.

[3] R. Eggeling, A. Gohr, J. Keilwagen, M. Mohr, S. Posch, A. Smith, and I. Grosse, “On the value of intra-motif dependencies of human insulator protein CTCF,” *PLoS ONE*, vol. 9, no. 1, p. e85629, 2014.

[4] R. Eggeling, P.-Y. Bourguignon, A. Gohr, and I. Grosse, “Gibbs sampling for parsimonious Markov models with latent variables,” in *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models (PGM)*, 2012.

[5] R. Eggeling, T. Roos, P. Myllymäki, and I. Grosse, “Model selection in a setting with latent variables,” in *Proc. 6th Workshop on Information Theoretic Methods in Science and Engineering (WITMSE-13)*, 2013.

[6] D. Reinberg, C. Allis, and T. Jenuwein, *Epigenetics.* Plainview, N.Y: Cold Spring Harbor Laboratory Press, 2007.

[7] W. Filipowicz, L. Jaskiewicz, F. Kolb, and R. Pillai, “Post-transcriptional gene silencing by siRNAs and miRNAs,” *Current Opinion in Structural Biology*, vol. 15, pp. 331–341, 2005.

[8] N. Sonenberg and A. Hinnebusch, “Regulation of Translation Initiation in Eukaryotes: Mechanisms and Biological Targets,” *Cell*, vol. 134, no. 4, pp. 731–745, 2009.

[9] A. Moll, A. Hildebrandt, H. Lenhof, and O. Kohlbacher, “BALLView: A tool for research and education in molecular modeling,” *Bioinformatics*, vol. 22, no. 3, pp. 365–366, 2006.

[10] P. Ferre-D’Amare, A.R. Pognonec, R. Roeder, and S. Burley, “Structure and function of the b/HLH/Z domain of USF,” *EMBO Journal*, vol. 13, no. 1, pp. 180–189, 1994.

[11] A. Gutmanas, Y. Alhroub, G. M. Battle, J. M. Berrisford, E. Bochet, M. J. Conroy, J. M. Dana, M. A. Fernandez Montecelo, G. van Ginkel, S. P. Gore, P. Haslam, R. Hatherley, P. M. Hendrickx, M. Hirshberg, I. Lagerstedt, S. Mir, A. Mukhopadhyay, T. J. Oldfield, A. Patwardhan, L. Rinaldi, G. Sahni, E. Sanz-García, S. Sen, R. A. Slowley, S. Velankar, M. E. Wainwright, and G. J. Kleywegt, “PDBe: Protein data bank in europe,” *Nucleic Acids Research*, vol. 42, no. D1, pp. D285–D291, 2014.

[12] A. Jolma, T. Kivioja, J. Toivonen, L. Cheng, G. Wei, M. Enge, M. Taipala, J. Vaquerizas, J. Yan, Sillanpää, M. Bonke, K. Palin, S. Talukder, T. Hughes, N. Luscombe, E. Ukkonen, and J. Taipala, “Multiplexed massively parallel selex for characterization of human transcription factor binding specificities,” *Genome Research*, vol. 20, pp. 861–873, 2010.

[13] S. Mukherjee, M. Berger, G. Jona, X. Wang, D. Muzzey, R. Young, and M. Bulyk, “Rapid analysis of the DNA-binding specificities of transcription factors with DNA microarrays,” *Nature Genetics*, vol. 36, no. 12, pp. 1331–1339, 2004.

[14] C. Zhu, K. Byers, R. McCord, Z. Shi, M. Berger, D. Newburger, K. Saulrieta, Z. Smith, M. Shah, M. Radkakrishnan, A. Philippakis, Y. Hu, F. De Masi, M. Pacek, A. Rolfs, T. Murthy, J. LaBaer, and M. Bulyk, “High-resolution DNA binding specificity analysis of yeast transcription factors,” *Genome Research*, vol. 19, pp. 556–566, January 2009.

[15] D. Johnson, A. Mortazavi, R. Myers, and B. Wold, “Genome-wide mapping of in vivo protein-DNA interactions,” *Science*, vol. 316, no. 5830, pp. 1497–1502, 2007.

[16] E. Mardis, “ChIP-seq: welcome to the new frontier,” *Nature Methods*, vol. 4, no. 8, pp. 613–614, 2007.

[17] P. Park, “ChIP–seq: advantages and challenges of a maturing technology,” *Nature Reviews*, vol. 10, pp. 669–680, 2009.

[18] E. Mardis, “Next-generation DNA sequencing methods,” *Annu. Rev. Genomics Hum. Genet.*, vol. 9, pp. 387–402, 2008.

[19] M. Metzker, “Sequencing technologies – the next generation,” *Nature Reviews Genetics*, vol. 11, pp. 31–46, 2010.

[20] H. Li, J. Ruan, and R. Durbin, “Mapping short DNA sequencing reads and calling variants using mapping quality scores,” *Genome Research*, vol. 18, pp. 1851–1858, 2008.

[21] H. Li and R. Durbin, “Fast and accurate short read alignment with Burrows-Wheeler transform,” *Bioinformatics*, vol. 25, pp. 174–1760, 2009.

[22] B. Langmead, C. Trapnell, M. Pop, and S. Salzberg, “Ultrafast and memory-efficient alignment of short DNA sequences to the human genome,” *Genome Biology*, vol. 10, p. R25, 2009.

[23] Y. Zhang, T. Liu, C. Meyer, J. Eeckhoute, D. Johnson, B. Bernstein, C. Nusbaum, R. Myers, M. Brown, W. Li, and X. Liu, “Model-based analysis of ChIP-Seq (MACS),” *Genome Biology*, vol. 9, p. R137, 2008.

[24] A. Boyle, J. Guinney, G. Crawford, and T. Furey, “F-Seq: a feature density estimator for high-throughput sequence tags,” *Bioinformatics*, vol. 24, pp. 2537–2538, 2008.

[25] J. Rozowsky, G. Euskirchen, R. Auerbach, Z. Zhang, T. Gibson, R. Bjornson, N. Carriero, M. Snyder, and M. Gerstein, “Peakseq enables systematic scoring of chip-seq experiments relative to controls,” *Nat. Biotechnol.*, vol. 27, no. 1, pp. 66–75, 2009.

[26] P. A. Sharp, “On the origin of RNA splicing and introns,” *Cell*, vol. 42(2), pp. 397–400, September 1985.

[27] R. Breathnach, C. Benoist, K. O’Hare, F. Gannon, and P. Chambon, “Ovalbumin gene: evidence for a leader sequence in mRNA and DNA sequences at the exon-intron boundaries.,” *Proc Natl Acad Sci U S A*, vol. 75, pp. 4853–4857, October 1978.

[28] A. Sandelin, W. Alkema, P. Engström, W. Wasserman, and B. Lenhard, “JASPAR: an open-access database for eukaryotic transcription factor binding profiles.,” *Nucleic Acids Research*, vol. 32, pp. D91–D94, 2004.

[29] V. Matys, E. Fricke, R. Geffers, E. Gößling, M. Haubrock, R. Hehl, K. Hornischer, D. Karas, A. Kel, O. Kel-Margoulis, D. Kloos, S. Land, B. Lewicki-Potapov, H. Michael, R. Münch, I. Reuter, S. Rotert, H. Saxel, M. Scheer, S. Thiele, and E. Wingender, "TRANSFAC: transcriptional regulation, from patterns to profiles," *Nucleic Acids Research*, vol. 33, pp. 374–378, 2003.

[30] C. Lawrence, S. Altschul, M. Boguski, J. Liu, A. Neuwald, and J. Wootton, "Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment," *Science*, vol. 262, pp. 208–214, 1993.

[31] T. Bailey and C. Elkan, "Fitting a mixture model by expectation maximization to discover motifs in biopolymers," in *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, pp. 28–36, 1994.

[32] G. Casella and E. George, "Explaining the Gibbs sampler," *The American Statistician*, vol. 46, pp. 167–174, 1992.

[33] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[34] T. Bailey, N. Williams, C. Misleh, and W. Li, "MEME: discovering and analyzing DNA and protein sequence motifs," *Nucleic Acids Research*, vol. 34, pp. W369–W373, 2006.

[35] W. Thompson, E. Rouchka, and C. Lawrence, "Gibbs recursive sampler: finding transcription factor binding sites," *Nucleic Acids Research*, vol. 31, pp. 3580–3585, July 2003.

[36] W. Thompson, L. Newberg, S. Conlan, L. McCue, and C. Lawrence, "The Gibbs centroid sampler," *Nucleic Acids Research*, vol. 35, pp. W232–W237, 2007.

[37] S. Sinha, E. van Nimwegen, and E. Siggia, "A probabilistic method to detect regulatory modules," *Bioinformatics*, vol. 19:Suppl.1, pp. i292–i301, 2003.

[38] W. Thompson, M. Palumbo, W. Wasserman, J. Liu, and C. Lawrence, "Decoding human regulatory circuits," *Genome Research*, vol. 14, pp. 1967–1974, 2004.

[39] G. Pavesi, M. G., and G. Pesole, "An algorithm for finding signals of unknown length in DNA," *Bioinformatics*, vol. 17, pp. S207–214, 2001.

[40] J. Keilwagen, J. Grau, I. Paponov, S. Posch, M. Strickert, and I. Grosse, "De-novo discovery of differentially abundant transcription factor binding sites including their positional preference," *PLoS Computational Biology*, vol. 7, p. e1001070, February 2011.

[41] W. Ao, J. Gaudet, W. Kent, S. Muttumu, and S. Mango, "Environmentally induced foregut remodeling by PHA-4/FoxA and DAF-12/NHR," *Science*, vol. 305, pp. 1743–1746, 2004.

[42] N. Kim, K. Tharakaraman, L. Mariño Ramírez, and J. Spouge, "Finding sequence motifs with Bayesian models incorporating positional information: an application to transcription factor binding sites," *BMC Bioinformatics*, vol. 9, p. 262, 2008.

[43] S. Sinha, M. Blanchette, and M. Tompa, "PhyME: a probabilistic algorithm for finding motifs in sets of orthologous sequences," *BMC Bioinformatics*, vol. 5, p. 170, October 2004.

[44] R. Siddharthan, E. Siggia, and E. Nimwegen, “PhyloGibbs: A Gibbs sampling motif finder that incorporates phylogeny,” *PLoS Computational Biology*, vol. 1, no. 7, p. e67, 2005.

[45] R. Siddharthan, “PhyloGibbs-MP: Module prediction and discriminative motif-finding by Gibbs sampling,” *PLoS Computational Biology*, vol. 4, no. 8, p. e1000156, 2008.

[46] A. Smith, P. Sumazin, and M. Zhang, “Identifying tissue-selective transcription factor binding sites in vertebrate promoters,” *Proc Natl Acad Sci U S A*, vol. 102, no. 5, pp. 1560–1565, 2005.

[47] E. Redhead and T. L. Bailey, “Discriminative motif discovery in DNA and protein sequences using the DEME algorithm,” *BMC Bioinformatics*, vol. 8, p. 385, 2007.

[48] T. L. Bailey, “DREME: Motif discovery in transcription factor ChIP-seq data,” *Bioinformatics*, vol. 27, pp. 1653–1659, 2011.

[49] G. Stormo, T. Schneider, and L. Gold, “Characterization of translational initiation sites in E.coli,” *Nucleic Acids Research*, vol. 10, no. 2, pp. 2971–2996, 1982.

[50] R. Staden, “Computer methods to locate signals in nucleic acid sequences,” *Nucleic Acids Research*, vol. 12, pp. 505–519, 1984.

[51] T. Schneider and R. Stephens, “Sequence logos: A new way to display consensus sequences,” *Nucleic Acids Research*, vol. 18, no. 20, p. 6097–6100, 1990.

[52] S. Wolfe, A. Greisman, E. Ramm, and C. Pabo, “Analysis of zinc fingers optimized via phage display: Evaluating the utility of a recognition code,” *J. Mol. Biol.*, vol. 285, pp. 1917–1934, 1999.

[53] T. Man and G. Stormo, “Non-independence of Mnt repressor-operator interaction determined by a new quantitative multiple fluorescence relative affinity (QuMFRA) assay,” *Nucleic Acids Research*, vol. 29, pp. 2471–2478, 2001.

[54] M. Bulyk, X. Huang, Y. Choo, and G. Church, “Exploring the DNA-binding specificities of zinc fingers with DNA microarrays,” *Proc Natl Acad Sci U S A*, vol. 98, pp. 7158–7163, 2001.

[55] M. Bulyk, P. Johnson, and G. Church, “Nucleotides of transcription factor binding sites exert interdependent effects on the binding affinities of transcription factors,” *Nucleic Acids Research*, vol. 30, no. 5, pp. 1255–1261, 2002.

[56] I. Udalova, R. Mott, D. Field, and D. Kwiatkowski, “Quantitative prediction of NF-kappa B DNA-protein interactions,” *Proc Natl Acad Sci U S A*, vol. 99, pp. 8167–72, 2002.

[57] M. Zhang and T. Marr, “A weights array method for splicing signals analysis,” *Computational Application for Biosciences*, vol. 9, pp. 499–509, 1993.

[58] J. Keilwagen, J. Baumbach, T. Kohl, and I. Grosse, “MotifAdjuster: a tool for computational reassessment of transcription factor binding site annotations,” *Genome Biology*, vol. 10, p. R46, 2009.

[59] K. Ellrott, C. Yang, F. Sladek, and T. Jiang, “Identifying transcription factor binding sites through Markov chain optimization,” *Bioinformatics*, vol. 18, pp. s100–109, 2002.

[60] G. Heckerman, D. Geiger, and D. Chickering, “Learning Bayesian networks: The combination of knowledge and statistical data,” *Machine Learning*, vol. 20, pp. 197–243, 1995.

[61] Y. Barash, G. Elidan, N. Friedman, and T. Kaplan, “Modeling dependencies in protein-DNA binding sites,” in *Proceedings of the seventh annual international conference on Research in computational molecular biology*, pp. 28–37, 2003.

[62] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques.* MIT Press, 2009.

[63] D. Chickering, “Learning Bayesian networks is NP-complete,” in *Learning from Data: Artificial Intelligence and Statistics V*, 1996.

[64] D. Chickering, “Learning equivalence classes of Bayesian-network structures,” *Journal of Machine Learning Research*, vol. 2, pp. 445–498, 2002.

[65] M. Tessier and D. Koller, “Ordering-based search: A simple and effective algorithm for learning Bayesian networks,” in *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI-05)*, 2005.

[66] M. Koivisto and K. Sood, “Exact Bayesian structure discovery in Bayesian networks,” *Journal of Machine Learning Research*, vol. 5, pp. 549–573, May 2004.

[67] S. Ott, S. Imoto, and S. Miyano, “Finding optimal models for small gene networks,” in *Pac. Symp. Biocomput.*, pp. 557–567, 2004.

[68] A. Singh and A. Moore, “Finding optimal Bayesian networks by dynamic programming,” tech. rep., Carnegie Mellon University, 2005.

[69] T. Silander and P. Myllymäki, “A simple approach for finding the globally optimal Bayesian network structure,” in *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, 2006.

[70] C. Chow and C. Liu, “Approximating discrete probability distributions with dependence trees,” *IEEE Trans. on Info. Theory,*, vol. 14, pp. 462–467, 1968.

[71] P.-Y. Bourguignon and D. Robelin, “Modèles de Markov parcimonieux,” in *Proceedings of JOBIM*, 2004.

[72] P.-Y. Bourguignon, *Parcimonie dans les modèles markoviens et applications à l’analyse des séquences biologiques.* PhD thesis, Université Evry Val d’Essonne, 2008.

[73] M. Seifert, A. Gohr, M. Strickert, and I. Grosse, “Parsimonious higher-order hidden Markov models for improved array-CGH analysis with applications to Arabidopsis thaliana,” *PLOS Computational Biology*, vol. 8, no. 1, p. e1002286, 2012.

[74] P. Volf and F. Willems, “Context maximizing: Finding MDL decision trees.,” in *15th Symp. Inform. Theory Benelux*, pp. 192–200, May 1994.

[75] J. Rissanen, “A universal data compression system,” *IEEE Trans. Inform. Theory*, vol. 29, no. 5, pp. 656–664, 1983.

[76] X. Zhao, H. Huang, and T. Speed, “Finding short DNA motifs using permuted Markov models,” *Journal of Computational Biology*, vol. 12, no. 6, pp. 894–906, 2005.

[77] I. Ben-Gal, A. Shani, A. Gohr, J. Grau, S. Arviv, A. Shmilovici, S. Posch, and I. Grosse, “Identification of transcription factor binding sites with variable-order Bayesian networks,” *Bioinformatics*, vol. 21, pp. 2657–2666, 2005.

[78] P. Bühlmann and A. Wyner, “Variable length Markov chains,” *Annals of Statistics*, vol. 27, pp. 480–513, 1999.

[79] E. Jaynes, *Probability Theory: The Logic of Science.* Cambridge University Press, 2003.

[80] D. Ramji and P. Foka, “CCAAT/enhancer-binding proteins : structure, function and regulation,” *Biochem. J.*, vol. 365, pp. 561–575, 2002.

[81] P. Grünwald, *Advances in Minimum Description Length: Theory and Applications*, ch. A Tutorial Introduction to the Minimum Description Length Principle, p. 80. MIT Press, 2005.

[82] J. Rissanen, “Modeling by shortest data description,” *Automatica*, vol. 14, pp. 465–471, 1978.

[83] P. Grünwald, *The minimum description length principle and reasoning under uncertainty.* PhD thesis, CWI, Netherlands, 1998.

[84] L. Kraft, “A device for quantizing, grouping, and coding amplitude modulated pulses,” Master’s thesis, Electrical Engineering Department, Massachusetts Institute of Technology, 1949.

[85] B. McMillan, “Two inequalities implied by unique decipherability,” *IEEE Trans. Information Theory*, vol. 2, no. 4, pp. 115–116, 1956.

[86] J. Myung, D. Navarro, and M. Pitt, “Model selection by normalized maximum likelihood,” *Journal of Mathematical Psychology*, vol. 50, pp. 167–179, 2006.

[87] G. E. Schwarz, “Estimating the dimension of a model,” *Annals of Statistics*, vol. 2, pp. 461–464, 1978.

[88] P. Grünwald, *The Minimum Description Length Principle.* MIT Press, June 2007.

[89] Y. Shtarkov, “Universal sequential coding of single messages,” *Problems of Information Transmission*, vol. 23, pp. 3–17, 1987.

[90] P. Kontkanen and P. Myllymäki, “A linear-time algorithm for computing the multinomial stochastic complexity,” *Information Processing Letters*, vol. 103, pp. 227–233, September 2007.

[91] T. Mononen and P. Myllymäki, “Fast NML computation for naive Bayes models,” in *Proceedings of the 10th International Conference on Discovery Science*, LNAI, pp. 151–160, Springer, 2007.

[92] T. Mononen and P. Myllymäki, “Computing the NML for Bayesian forests via matrices and generating polynomials,” in *Proceedings of the 2008 IEEE Information Theory Workshop*, 2008.

[93] T. Silander, T. Roos, and P. Myllymäki, “Locally minimax optimal predictive modeling with Bayesian networks,” in *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pp. 504–511, 2009.

[94] W. Buntine, “Theory refinement on Bayesian networks,” in *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 52–60, Morgan Kaufmann, 1991.

[95] T. Silander, T. Roos, P. Kontkanen, and P. Myllymäki, “Factorized NML criterion for learning Bayesian network structures,” in *Proceedings of the 4th European Workshop on Probabilistic Graphical Models (PGM-08)*, 2008.

[96] P. Kontkanen, W. Buntine, P. Myllymäki, J. Rissanen, and H. Tirri, “Efficient computation of stochastic complexity,” in *Proceedings of the Ninth International Conference on Artificial Intelligence and Statistics*, 2003.

[97] H. Akaike, “A new look at the statistical model identification,” *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.

[98] C. Hurvich and C.-L. Tsai, “Regression and time series model selection in small samples,” *Biometrika*, vol. 76, pp. 297–307, 1989.

[99] G. Yeo and C. Burge, “Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals,” *Journal of Computational Biology*, vol. 11(2/3), pp. 377–394, 2004.

[100] T. Silander, P. Kontkanen, and P. Myllymäki, “On sensitivity of the MAP Bayesian network structure to the equivalent sample size parameter,” in *Proceedings of the The 23rd Conference on Uncertainty in Artificial Intelligence (UAI-2007)*, pp. 360–367, 2007.

[101] N. Day, “Estimating the components of a mixture of normal distribution,” *Biometrika*, vol. 56, no. 3, pp. 463–474, 1969.

[102] I. Lemnian, R. Eggeling, and I. Grosse, “Extended sunflower hidden Markov models for the recognition of homotypic cis-regulatory modules,” in *German Conference on Bioinformatics 2013*, vol. 34 of *OpenAccess Series in Informatics (OASIcs)*, pp. 101–109, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013.

[103] C. Lawrence and A. Reilly, “An expectation maximization algorithm for the identification and characterization of common sites in unaligned biopolymer sequences.,” *Proteins: Structure, Function and Genetics*, vol. 7, pp. 41–51, 1990.

[104] M. Meila and M. Jordan, “Learning with mixtures of trees,” *Journal of Machine Learning Research*, vol. 1, pp. 1–48, 2000.

[105] N. Friedman, “Learning belief networks in the presence of missing values and hidden variables,” in *Proc. Fourteenth Inter. Conf. on Machine Learning (ICML97)*, 1997.

[106] N. Friedman, “The Bayesian structural EM algorithm,” in *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 1998.

[107] A. Gohr, S. Posch, and I. Grosse, “Mixtures of parsimonious Markov models,” Tech. Rep. 2012/2, Institute of Computer Science, Martin Luther University Halle-Wittenberg, Germany, 2012.

[108] A. Bell, A. West, and G. Felsenfeld, “The protein CTCF is required for the enhancer blocking activity of vertebrate insulators,” *Cell*, vol. 98, pp. 387–396, 1999.

[109] B. Burgess-Beusse, C. Farrell, M. Gaszner, M. Litt, V. Mutskov, F. Recillas-Targa, M. Simpson, A. West, and G. Felsenfeld, "The insulation of genes from external enhancers and silencing chromatin," *Proc Natl Acad Sci U S A*, vol. 99, pp. 16433–16437, 2002.

[110] J. Phillips and V. Corces, "CTCF: master weaver of the genome," *Cell*, vol. 137, no. 7, pp. 1194–1211, 2009.

[111] X. Xie, T. Mikkelsen, A. Gnirke, K. Lindblad-Toh, M. Kellis, and E. Lander, "Systematic discovery of regulatory motifs in conserved regions of the human genome, including thousands of CTCF insulator sites," *Proc Natl Acad Sci U S A*, vol. 107, pp. 7145–7150, 2007.

[112] T. Kim, Z. Abdullaev, A. Smith, K. Ching, D. Loukinov, R. Green, M. Zhang, V. Lobanenkov, and B. Ren, "Analysis of the vertebrate insulator protein CTCF-binding sites in the human genome," *Cell*, vol. 128, pp. 1231–1245, 2007.

[113] H. Wang, M. Maurano, H. Qu, K. Varley, J. Gertz, F. Pauli, K. Lee, T. Canfield, M. Weaver, R. Sandstrom, R. Thurman, R. Kaul, R. Myers, and J. Stamatoyannopoulos, "Widespread plasticity in CTCF occupancy linked to DNA methylation," *Genome Research*, vol. 9, p. 1680–1688, 2012.

[114] A. Boyle, L. Song, B.-K. Lee, D. London, D. Keefe, E. Birney, V. Iyer, G. Crawford, and T. Furey, "High-resolution genome-wide in vivo footprinting of diverse transcription factors in human cells," *Genome Research*, vol. 21, pp. 456–464, 2011.

[115] The ENCODE Project Consortium, "Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project," *Nature*, vol. 447, p. 7146, 2007.

[116] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[117] J. S. Liu, *Monte Carlo Strategies in Scientific Computing.* Springer Series in Statistics, 2001.

[118] J. Grau, J. Keilwagen, A. Gohr, B. Haldemann, S. Posch, and I. Grosse, "Jstacs: A Java framework for statistical analysis and classification of biological sequences," *Journal of Machine Learning Research*, vol. 13, pp. 1967–1971, 2012.

[119] S. Nielsen, "The stochastic EM algorithm: Estimation and asymptotic results," *Bernoulli*, vol. 6, no. 3, pp. 457–489, 2000.

[120] The ENCODE Project Consortium, "A user's guide to the encyclopedia of DNA elements," *PLoS Biology*, vol. 9, p. e1001046, April 2011.

[121] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

A

DP algorithm for PCT maximization

Here, we show the operating mode of the dynamic programming algorithm for finding optimal PCTs (Section 1.4.2) using a small example. We consider the case of learning a PCT of depth $d = 2$ for the ternary alphabet $\mathcal{A} = \{\texttt{A,B,C}\}$. The extended PCT for this depth and alphabet size, on which the dynamic programming algorithm operates, is shown in Figure A.1. We identify a particular node in the extended PCT by the cross product of the labels of all nodes on the path from that particular node up to the root, excluding the root itself.

There are nine different context words, which corresponds to the maximal size of the PCT, but $7^2 = 49$ different contexts, which is given by the number of leaves of the extended PCT (Figure A.1). We assume each leaf node in the extended PCT to represent a local score, such as the Bayesian marginal likelihood for a particular context. The goal of the algorithm is to reduce the extended PCT to a valid PCT such that the sum of the scores of the remaining leaves is maximized.

In the first step, we traverse the extended PCT top down in a depth-first search manner. Once we reach the first leaf in the extended PCT, which is here the node $\{\texttt{A}\} \times \{\texttt{A}\}$, we compute its score. We repeat this procedure for each of its siblings, i.e., the nodes $\{\texttt{B}\} \times \{\texttt{A}\}$, $\{\texttt{C}\} \times \{\texttt{A}\}$, $\{\texttt{A,B}\} \times \{\texttt{A}\}$,$\{\texttt{A,C}\} \times \{\texttt{A}\}$,$\{\texttt{B,C}\} \times \{\texttt{A}\}$, and $\{\texttt{A,B,C}\} \times \{\texttt{A}\}$. Each of these leaf nodes is a valid subtree, in the sense that it may occur in a valid PCT and has a score assigned to it (Figure A.2). Next, we compute the score of all possible partitions of the alphabet, i.e., we compute a score for all valid combinations of child nodes for node $\{\texttt{A}\}$ (Figure A.3).

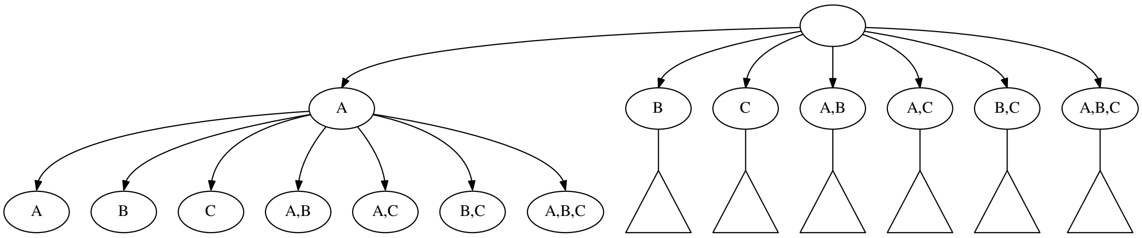


Figure A.1.: **Extended PCT of $d = 2$ for $\mathcal{A} = \{\texttt{A,B,C}\}$.** The first subtree, rooted at node $\{\texttt{A}\}$ is shown in full detail. The remaining subtrees 2–7 have the same structure, but are displayed as triangles due to limited space.

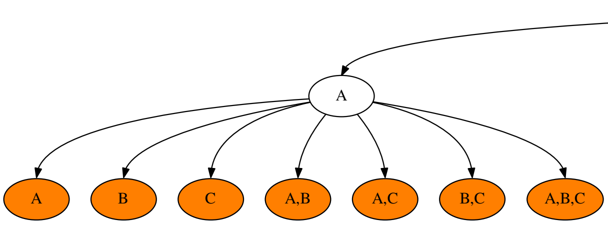


Figure A.2.: **Extended PCT with computed scores for the leaves of first subtree.** Orange color indicates that the optimal score of a particular node is computed already.

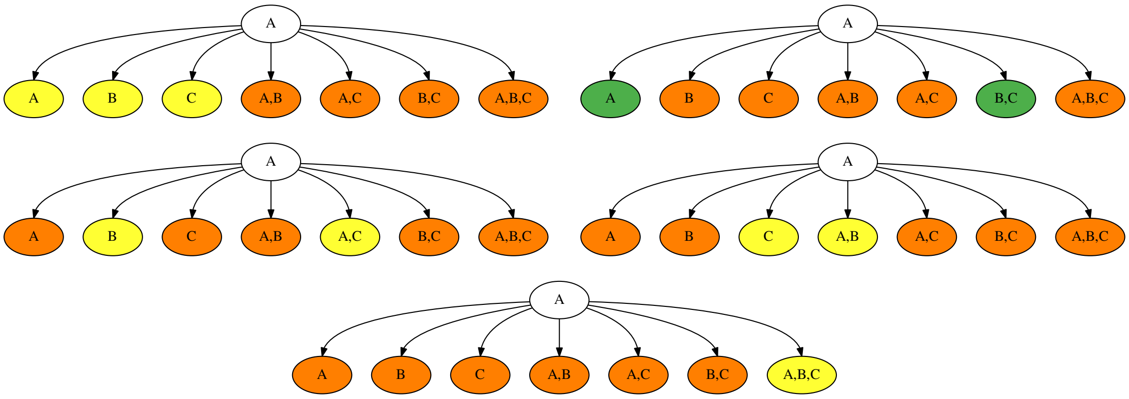


Figure A.3.: **All five valid child node combination for node {`A`}**, the members of which are colored in yellow and green. The score of a child node combination is the sum of the scores of the involved nodes, and we choose the combination with the maximal score (here colored in green), discarding the remaining nodes.

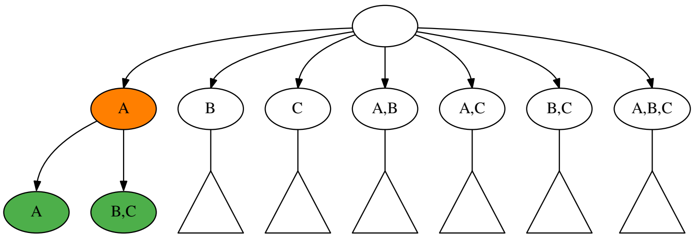


Figure A.4.: **Extended PCT with first subtree pruned.** Green color indicates that a node is member of an optimal partition, i.e., child node selection. The score of node {{`A`}} is computed as sum of the scores of its selected children.

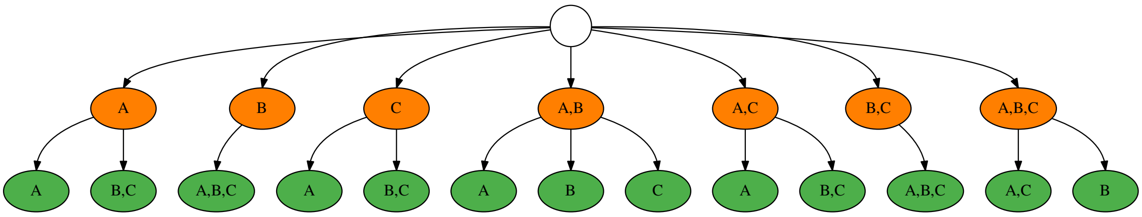


Figure A.5.: **Extended PCT with all second-level subtrees pruned.** Now all nodes on the first level have a score assigned to them, which is the same situation as for the first subtree in Figure A.2 and thus key to the recursion.

There are five different partitions for a ternary alphabet, and the score of a partition is the sum of the score of the contributing nodes. We select the partition with the maximal score and discard all children of node {`A`} in the extended PCT that are not part of that partition. In the example of Figure A.4, the optimal partition involves the two leaves {`A,B`} × {`A`} and {`C`} × {`A`}, both colored in green. We assign the score of the optimal partition, which is the sum of the scores of both leaves to parent node {`A`}.

After having locally reduced the extended PCT to a subtree that fulfills the properties of a PCT, we repeat this procedure for all siblings of {`A`} (Figure A.5). Hence, all children of the global root are themselves roots of subtrees that already fulfill the properties of a PCT, and all of them have a scored assigned to them (indicated by orange color). Now we can apply the method of selecting an optimal partition, e.g. an optimal valid combination of child nodes to the root. A possible result, which is in this example the optimal PCT resulting from an application of the DP algorithm, is shown in Figure A.6.

The example of $d = 2$ is the simplest possible example where the recursion step of the DP algorithm can be appreciated, as for $d = 1$ computing the optimal partition is equivalent to enumerating all possible PCTs. However, it should be noted that in this particular case of a ternary alphabet and $d = 2$, the explicit enumeration of all possible PCTs would not be slower than the DP algorithm, as there are only 205 possible PCTs of the same depth and alphabet size. For learning deeper PCTs, however, the effect of the recursion becomes more

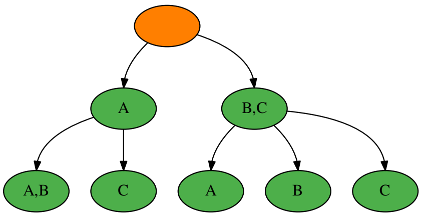


Figure A.6: **Resulting optimal PCT.** After selecting an optimal combination children for the global root, the score of this selection is assigned to the root and thus considered as the score of this PCT.

pronounced. In the case of a quarternary alphabet, for example, there are 72,465 PCTs of depth $d = 2$, and about 2.75×10^{19} PCTs of depth $d = 3$, and at least in the latter case explicit enumeration can be considered to be infeasible. The extended PCTs, however, the size of which is the limiting factor for the DP algorithm, have only 241 ($d = 2$) and 3,616 ($d = 3$) nodes, which can be quickly traversed and reduced to an optimal PCT.

B

Additional derivations

B.1. Bayesian marginal likelihood score for PMMs

We intend to perform Bayesian model selection for PMMs, and thus attempt to maximize $P(\vec{\tau}|\mathbf{x})$ w.r.t. τ, which is for given $\mathbf{x}$ equivalent to maximizing

$$\begin{aligned}
P(\vec{\tau},\mathbf{x}) &= \int P(\mathbf{x}|\Theta)P(\Theta)d\Theta_{\vec{\tau}} && \text{(B.1)}\\
&= P(\vec{\tau})\int P(\mathbf{x}|\Theta_{\vec{\tau}})P(\Theta_{\vec{\tau}})d\Theta_{\vec{\tau}} && \text{(B.2)}\\
&= P(\vec{\tau})\int \left(\prod_{\ell=1}^{W}\prod_{c\in\tau_\ell}\prod_{a\in\mathcal{A}}(\theta_{\ell ca})^{N_{\ell ca}}\right)\left(\prod_{\ell=1}^{W}\prod_{c\in\tau_\ell}\frac{1}{\mathcal{B}(\vec{\alpha}_{\ell c})}\prod_{a\in\mathcal{A}}(\theta_{\ell ca})^{\alpha_{\ell ca}-1}\right)d\Theta_{\vec{\tau}} && \text{(B.3)}\\
&= P(\vec{\tau})\prod_{\ell=1}^{W}\prod_{c\in\tau_\ell}\frac{1}{\mathcal{B}(\vec{\alpha}_{\ell c})}\int\prod_{a\in\mathcal{A}}(\theta_{\ell ca})^{N_{\ell ca}+\alpha_{\ell ca}-1}d\vec{\theta}_{\ell c} && \text{(B.4)}\\
&= P(\vec{\tau})\prod_{\ell=1}^{W}\prod_{c\in\tau_\ell}\frac{\mathcal{B}(\vec{N}_{\ell c}+\vec{\alpha}_{\ell c})}{\mathcal{B}(\vec{\alpha}_{\ell c})} && \text{(B.5)}\\
&\propto \left(\prod_{\ell=1}^{W}\kappa^{|\tau_\ell|}\right)\prod_{\ell=1}^{W}\prod_{c\in\tau_\ell}\frac{\mathcal{B}(\vec{N}_{\ell c}+\vec{\alpha}_{\ell c})}{\mathcal{B}(\vec{\alpha}_{\ell c})} && \text{(B.6)}\\
&= \prod_{\ell=1}^{W}\prod_{c\in\tau_\ell}\kappa\,\frac{\mathcal{B}(\vec{N}_{\ell c}+\vec{\alpha}_{\ell c})}{\mathcal{B}(\vec{\alpha}_{\ell c})} && \text{(B.7)}
\end{aligned}$$

Since the resulting structure score for the PMM factorizes into a product of local scores for each leaf of each PCT, it is sufficient to find

$$\forall_{\ell=1}^{W}: \hat{\tau}_\ell := \underset{\tau_\ell}{\operatorname{argmax}}\prod_{c\in\tau_\ell}\kappa\,\frac{\mathcal{B}(\vec{N}_{\ell c}+\vec{\alpha}_{\ell c})}{\mathcal{B}(\vec{\alpha}_{\ell c})}, \tag{B.8}$$

which can be achieved using the dynamic programming algorithm for finding optimal PCTs (Section 1.4.2).

B.2. Factorized Sequential NML

We consider N independent realizations $\vec{x} = (x_1, \ldots, x_N)$ from the alphabet $\mathcal{A}$. Let N_a denote the number of times symbol $a \in \mathcal{A}$ has been observed in $\vec{x}$. The factorized sequential NML (fsNML) parameter estimate $\hat{\theta}^{\text{fsNML}}$ is defined as the sequential NML predictive probability distribution $P_{\text{sNML}}(y|\vec{x})$ for the $(N+1)$-th observation [93].

$$
\begin{aligned}
P_{\text{sNML}}(y|\vec{x}) &= \frac{\hat{P}(\vec{x}, y)}{\sum_{y' \in \mathcal{A}} \hat{P}(\vec{x}, y')} && \text{(B.9)} \\
&= \frac{\prod_{a \in \mathcal{A}} \left(\frac{N_a + \delta_{a,y}}{N+1} \right)^{N_a + \delta_{a,y}}}{\sum_{y' \in \mathcal{A}} \prod_{a' \in \mathcal{A}} \left(\frac{N_{a'} + \delta_{a',y'}}{N+1} \right)^{N_{a'} + \delta_{a',y'}}} && \text{(B.10)} \\
&= \frac{\prod_{a \in \mathcal{A}} \left(N_a + \delta_{a,y} \right)^{N_a + \delta_{a,y}}}{\sum_{y' \in \mathcal{A}} \prod_{a' \in \mathcal{A}} \left(N_{a'} + \delta_{a',y'} \right)^{N_{a'} + \delta_{a',y'}}} && \text{(B.11)} \\
&= \frac{\prod_{a \in \mathcal{A}} \left(N_a^{N_a} \right) \frac{(N_y+1)^{(N_y+1)}}{{N_y}^{N_y}}}{\sum_{y' \in \mathcal{A}} \prod_{a' \in \mathcal{A}} \left(N_{a'}^{N_{a'}} \right) \frac{(N_{y'}+1)^{(N_{y'}+1)}}{{N_{y'}}^{N_{y'}}}} && \text{(B.12)} \\
&= \frac{\frac{(N_y+1)^{(N_y+1)}}{{N_y}^{N_y}}}{\sum_{y' \in \mathcal{A}} \frac{(N_{y'}+1)^{(N_{y'}+1)}}{{N_{y'}}^{N_{y'}}}} && \text{(B.13)} \\
&= \frac{e(N_y)(N_y + 1)}{\sum_{y' \in \mathcal{A}} e(N_{y'})(N_{y'} + 1)} && \text{(B.14)}
\end{aligned}
$$

B.3. EM algorithm for motif discovery

In order to learn a PMM from a set of promoter sequences, we apply the maximum a posteriori principle to the ZOOPS model (Section 4.2). However, the maximum of the posterior density of the model cannot be found analytically, but only be approximated by a modified EM algorithm [33]. For motif discovery, it is often sufficient not to update the parameters Θ_{f} of the flanking model during the EM algorithm, but estimate them in advance from the entire data set and keep them fixed during the iteration procedure. As a consequence, we consider Θ_{f} to be given, and denote from now on the tupel $(\Theta_{\mathrm{m}}, \Theta_{\mathrm{c}})$ as Θ. This EM algorithm updates iteratively weighted estimates $\gamma^{(t)}$ for the latent variables based on parameters $\Theta^{(t)}$ (E step) and parameters $\Theta^{(t+1)}$ based on weighted estimates of latent variables $\gamma^{(t)}$ (M step), which we derive in the next two sections.

E step

In the E(xpectation) step, we estimate – for each sequence – the probability of each assignment of latent variables, given the current parameters.

$$\begin{aligned}\gamma_{iklm}^{(t)} &= P(u_i = k, v_i = l, s_i = m | \vec{x}_i, \Theta^{(t)}) \\ &= \frac{P(u_i = k, v_i = l, s_i = m, \vec{x}_i | \Theta^{(t)})}{\sum_{u_i} \sum_{v_i} \sum_{s_i} P(u_i, v_i, s_i, \vec{x}_i | \Theta^{(t)})} \\ &= \frac{P(\vec{x}_i | u_i = k, v_i = l, s_i = m, \Theta^{(t)}) P(u_i = k | \nu^{(t)}) P(v_i = l) P(s_i = m | \sigma^{(t)})}{\sum_{u_i} \sum_{v_i} \sum_{s_i} P(\vec{x}_i | u_i, v_i, s_i, \Theta^{(t)}) P(u_i | \nu^{(t)}) P(v_i) P(s_i | \sigma^{(t)})}\end{aligned}$$

M step

In the M(aximization) step, we compute the parameters given the weights of the latent variables by utilizing the Q-function, defined by

$$Q(\Theta, \Theta^{(t)}) = \sum_{i=1}^{N} \sum_{u_i} \sum_{v_i} \sum_{s_i} \gamma_{iu_iv_is_i}^{(t)} \log P(\vec{x}_i, u_i, v_i, s_i | \Theta) \tag{B.15}$$

We obtain the next set of parameters $\Theta^{(t+1)}$ as set of parameters that maximize the Q-function plus logarithm the parameter prior.

$$\begin{aligned}\Theta^{(t+1)} &= \underset{\Theta}{\operatorname{argmax}} \left(Q(\Theta, \Theta^{(t)}) + \log P(\Theta) \right) \\ &= \underset{\Theta}{\operatorname{argmax}} \left(\sum_{i=1}^{N} \sum_{u_i} \sum_{v_i} \sum_{s_i} \gamma_{iu_iv_is_i}^{(t)} \log P(\vec{x}_i, u_i, v_i, s_i | \Theta) + \log P(\Theta) \right) \\ &= \underset{\Theta}{\operatorname{argmax}} \ \mathcal{F}(\Theta)\end{aligned}$$

while $\mathcal{F}(\Theta)$ is fully written as

$$\begin{aligned}
\mathcal{F}(\Theta) &= \log P(\Theta_{\mathrm{m}}) + \log P(\nu) + \log P(\sigma) \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{iu_iv_is_i}\log P_{\mathrm{f}}(\vec{x}_i|\Theta_{\mathrm{f}})^{\delta_{u_i,0}} \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{iu_iv_is_i}\log P_{\mathrm{f}}(X_{i,1},\ldots,X_{i,v_i-1}|\Theta_{\mathrm{f}})^{\delta_{u_i,1}} \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{iu_iv_is_i}\log P_{\mathrm{m}}(X_{i,v_i},\ldots,X_{i,v_i+W-1}|\Theta_{\mathrm{m}})^{\delta_{u_i,1}\delta_{s_i,1}} \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{iu_iv_is_i}\log P_{\mathrm{m}}(\mathrm{rc}(X_{i,v_i},\ldots,X_{i,v_i+W-1})|\Theta_{\mathrm{m}})^{\delta_{u_i,1}\delta_{s_i,0}} \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{iu_iv_is_i}\log(\nu^{\delta_{u_i,1}}(1-\nu)^{\delta_{u_i,0}}) \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{iu_iv_is_i}\log(\sigma^{\delta_{s_i,1}}(1-\sigma)^{\delta_{s_i,0}}) \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{iu_iv_is_i}\log\frac{1}{L_i-W+1},
\end{aligned}$$

Canceling all terms that do not pertain Θ, and simplifying the remaining terms, we obtain

$$\begin{aligned}
\mathcal{F}(\Theta) &= \log P(\Theta_{\mathrm{m}}) + \log P(\nu) + \log P(\sigma) \\
&+ \sum_{i=1}^{N}\sum_{v_i}\gamma^{(t)}_{i1v_i1}\log P_{\mathrm{m}}(X_{i,v_i},\ldots,X_{i,v_i+W-1}|\Theta_{\mathrm{m}}) \\
&+ \sum_{i=1}^{N}\sum_{v_i}\gamma^{(t)}_{i1v_i0}\log P_{\mathrm{m}}(\mathrm{rc}(X_{i,v_i},\ldots,X_{i,v_i+W-1})|\Theta_{\mathrm{m}}) \\
&+ \sum_{i=1}^{N}\sum_{v_i}\sum_{s_i}\gamma^{(t)}_{i1v_is_i}\log\nu + \gamma^{(t)}_{i0v_is_i}\log(1-\nu) \\
&+ \sum_{i=1}^{N}\sum_{u_i}\sum_{v_i}\gamma^{(t)}_{iu_iv_iF}\log\sigma + \gamma^{(t)}_{iu_iv_i0}\log(1-\sigma).
\end{aligned}$$

Maximizing $\mathcal{F}(\Theta)$ w.r.t. Θ yields

$$\begin{aligned}
\nu^{(t+1)} &= \operatorname*{argmax}_{\nu} \log P(\nu) + \sum_{i=1}^{N} \sum_{v_i} \sum_{s_i} \gamma^{(t)}_{i1v_is_i} \log \nu + \gamma^{(t)}_{i0v_is_i} \log(1-\nu) \\
&= \operatorname*{argmax}_{\nu} \, (\gamma^{(t)}_{\cdot 1 \cdot\cdot} + a_1) \log \nu + (\gamma^{(t)}_{\cdot 0 \cdot\cdot} + a_2) \log(1-\nu) \\
&= \frac{\gamma^{(t)}_{\cdot 1 \cdot\cdot} + a_1}{\gamma^{(t)}_{\cdot\cdot\cdot\cdot} + a_\cdot},
\end{aligned}$$

$$\begin{aligned}
\sigma^{(t+1)} &= \operatorname*{argmax}_{\sigma} \log P(\sigma) + \sum_{i=1}^{N} \sum_{u_i} \sum_{v_i} \gamma^{(t)}_{iu_iv_i1} \log \sigma + \gamma^{(t)}_{iu_iv_i0} \log(1-\sigma) \\
&= \operatorname*{argmax}_{\sigma} \, (\gamma^{(t)}_{\cdot\cdot\cdot 1} + b_1) \log \sigma + (\gamma^{(t)}_{\cdot\cdot\cdot 0} + b_2) \log(1-\sigma) \\
&= \frac{\gamma^{(t)}_{\cdot\cdot\cdot 1} + b_1}{\gamma^{(t)}_{\cdot\cdot\cdot\cdot} + b_\cdot},
\end{aligned}$$

and

$$\begin{aligned}
\Theta_{\mathrm{m}}^{(t+1)} = \operatorname*{argmax}_{\Theta_{\mathrm{m}}} \Bigg(& \log P(\Theta_{\mathrm{m}}) + \sum_{i=1}^{N} \sum_{v_i=1}^{L_i-W+1} \\
& + \gamma^{(t)}_{i1v_i1} \log P_{\mathrm{m}}(X_{i,v_i}, \ldots, X_{i,v_i+W-1} | \Theta_{\mathrm{m}}) \\
& + \gamma^{(t)}_{i1v_i0} \log P_{\mathrm{m}}(\mathrm{rc}(X_{i,v_i}, \ldots, X_{i,v_i+W-1}) | \Theta_{\mathrm{m}}) \Bigg)
\end{aligned}$$

Let $S = \{(i,v,s) | i \in \{1,\ldots,N\}, v \in \{1,\ldots,L_i - W + 1\}, s \in \{1,0\}\}$ denote the space of all motif occurrences of length W in the data and let $\vec{x}_j$ denote the sequence corresponding to the motif occurrence $j \in S$, we rewrite the previous formula as

$$\Theta_{\mathrm{m}}^{(t+1)} = \operatorname*{argmax}_{\Theta_{\mathrm{m}}} \left(\log P(\Theta_{\mathrm{m}}) + \sum_{j=1}^{S} \gamma_j \log P_{\mathrm{m}}(\vec{Y}_j | \Theta_{\mathrm{m}}) \right).$$

We observe this estimator to be identical to that of a single component of a mixture model, permitting to use the optimization algorithm of Gohr et al. [107].

C

Additional results

C.1. CTCF study

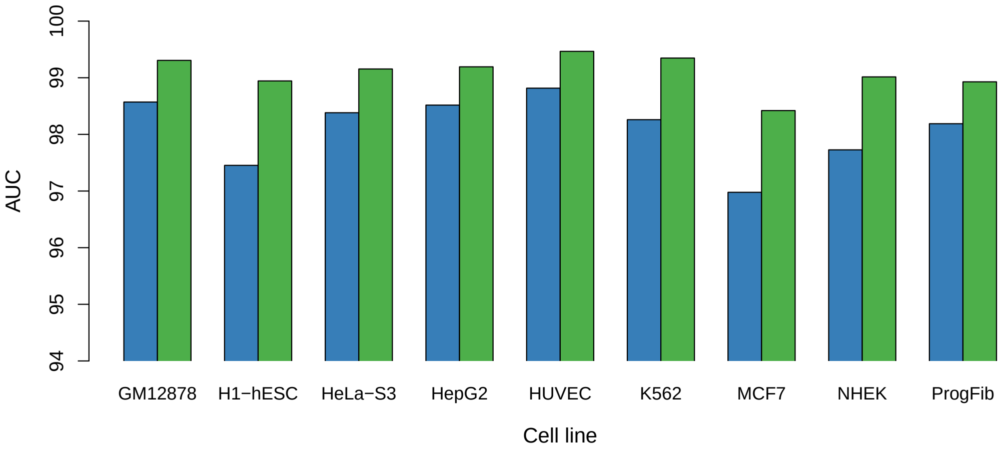
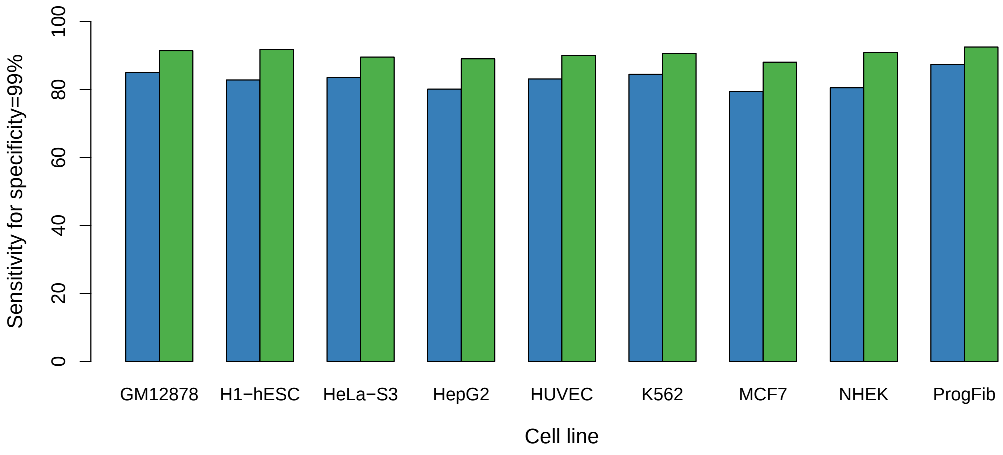
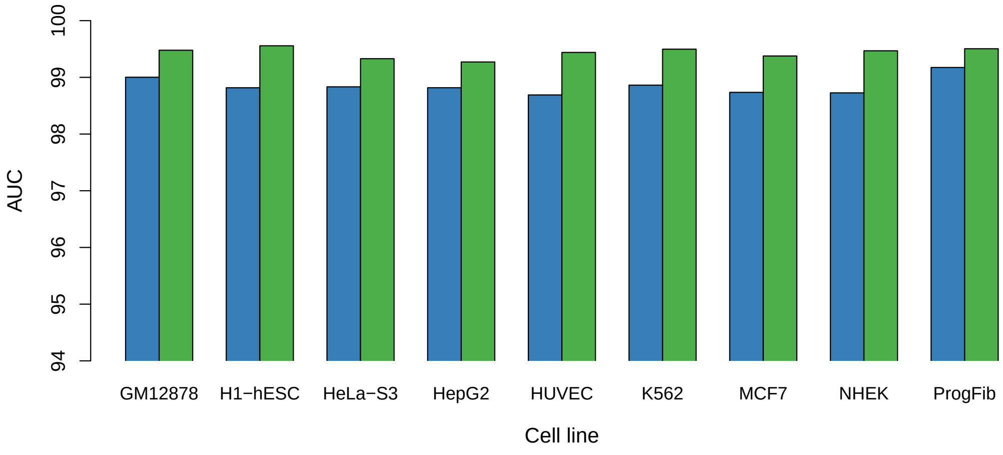


Figure C.1.: **Classification using AUC-ROC as performance measure.**



(a) Sensitivity



(b) AUC

Figure C.2.: **Classification using randomly sampled genomic background sequences.**

C.2. ENCODE ChIP-seq study

Table C.1.: **AUC values in percentage for data sets of category A.**

TF	N	MM0	MM1	MM2	MM3	BIC2	BIC3	BIC4	fNML2	fNML3	fNML4
ATF2	1193	81.57	86.04	82.00	73.77	86.71	85.87	86.60	86.42	78.07	73.67
ATF3	962	95.23	96.72	96.90	95.79	96.58	96.91	97.63	97.46	97.00	95.87
BACH1	2292	94.97	96.76	96.41	94.69	96.73	96.61	96.53	96.65	96.46	93.49
BCL11A	504	71.21	75.86	73.81	71.85	79.56	77.23	76.06	76.95	72.32	65.86
BRCA1	405	97.36	98.27	98.72	98.33	98.27	98.42	98.01	98.70	98.03	97.63
CEBPB	3112	97.08	97.74	96.75	94.10	97.51	96.86	96.55	97.22	96.13	92.42
CTCF	10822	99.24	99.57	99.64	99.52	99.61	99.62	99.64	99.63	99.61	99.42
EGR1	1749	96.11	96.53	96.68	95.91	96.56	96.49	95.99	96.67	96.11	94.69
FOSL1	223	97.91	98.07	98.78	97.31	98.72	98.61	98.86	97.87	97.88	97.01
GABPA	1130	99.44	99.53	99.37	98.86	99.46	99.39	99.37	99.43	99.24	98.83
JUN	430	79.74	93.14	91.71	81.93	91.07	90.52	89.59	92.04	91.63	79.85
JUND	1690	84.30	90.99	92.39	88.31	92.42	92.17	92.36	92.15	84.52	88.90
KDM5A	325	95.89	97.53	97.64	97.19	97.04	97.23	96.91	97.20	96.48	95.66
MAFK	2284	95.85	97.27	97.74	97.47	97.75	97.77	97.88	97.95	97.92	96.46
MAX	2225	92.81	94.27	94.66	90.52	94.83	94.54	94.70	94.93	93.62	91.30
NANOG	1095	74.73	79.16	77.17	72.35	78.88	80.19	81.39	81.82	75.16	70.13
NRF1	903	99.58	99.73	99.69	99.61	99.72	99.72	99.71	99.68	99.70	99.60
POU5F1	800	92.72	93.78	94.22	89.42	95.25	94.38	93.94	94.90	94.67	87.21
RAD21	15136	98.69	99.31	99.37	99.15	99.36	99.37	99.42	99.37	99.24	99.08
REST	2656	95.92	98.92	99.10	99.01	98.98	99.02	99.17	99.14	99.09	98.86
RFX5	338	96.51	97.49	96.61	92.81	97.65	97.48	97.58	97.35	95.78	90.68
RXRA	262	85.67	84.93	82.79	79.18	86.88	84.57	82.77	83.76	79.54	76.62
SIX5	684	97.59	97.83	98.12	98.08	97.62	97.71	97.91	97.99	98.16	98.18
SP4	1150	95.18	95.52	95.57	94.26	95.14	95.32	95.11	95.19	94.61	95.69
SRF	1020	93.57	96.76	97.02	96.03	97.35	97.16	97.02	96.96	96.60	94.85
TCF12	1567	85.50	87.63	84.65	79.58	87.70	87.16	86.81	86.60	82.13	80.62
TEAD4	3972	92.78	95.03	95.06	92.26	95.20	95.31	95.28	94.91	94.08	90.54
USF1	5208	98.48	98.83	98.85	98.42	98.86	98.69	98.60	98.77	98.56	98.08
USF2	1391	99.10	99.24	99.12	98.58	99.29	99.18	99.26	99.24	99.08	98.36
YY1	3646	92.06	97.38	97.14	95.65	97.20	96.94	96.86	97.23	96.28	95.17

Table C.2.: **AUC values in percentage for data sets of category B.**

TF	N	MM0	MM1	MM2	MM3	BIC2	BIC3	BIC4	fNML2	fNML3	fNML4
CHD2	1368	95.37	98.35	98.11	96.87	96.34	96.72	97.37	98.00	97.94	96.97
MXI1	1271	92.09	91.85	95.27	94.36	92.47	93.99	94.79	94.99	94.71	94.55
SIN3AK20	4260	90.51	90.12	91.69	92.32	91.07	90.93	92.89	91.10	91.55	91.60
SP1	3009	84.70	87.96	89.65	87.32	87.48	88.57	88.29	88.99	88.42	87.63
SP2	492	93.48	94.90	96.06	95.73	94.67	94.31	94.06	96.01	95.88	94.95
ZNF143	6134	87.96	93.63	96.00	96.18	95.03	95.50	95.78	95.30	95.91	93.57

Table C.3.: **AUC values in percent for data sets of category C.**

TF	*N*	MM0	MM1	MM2	MM3	BIC2	BIC3	BIC4	fNML2	fNML3	fNML4
SUZ12	958	94.46	94.35	94.54	93.38	94.54	94.16	93.96	93.83	93.47	93.61
CHD1	1380	80.92	78.89	76.97	72.60	80.17	77.12	77.81	78.21	74.52	74.70
MYC	244	83.50	83.29	81.03	77.00	83.63	83.74	83.45	83.18	80.95	74.94
TAF1	4008	96.88	97.09	96.90	96.69	97.21	97.02	96.32	96.98	96.39	96.01
TBP	3429	94.16	94.55	93.33	92.44	93.74	93.77	93.66	93.84	92.64	91.54
CTBP2	1418	91.98	91.90	91.45	90.07	91.68	91.31	91.43	91.03	91.43	87.33
EP300	1787	77.73	79.43	81.01	80.23	79.28	78.23	78.56	80.17	79.54	78.94
EZH2	881	97.72	97.96	97.91	97.45	97.84	97.84	97.86	97.73	97.67	93.93
GTF2F1	709	91.22	92.04	92.19	91.11	91.77	91.87	91.54	91.32	92.63	90.55
HDAC2	1129	83.13	83.71	85.17	82.41	84.44	84.24	84.14	84.78	83.43	83.14
POLR2A	4077	90.92	89.94	82.39	78.91	89.48	84.98	83.55	85.65	83.33	78.17
RBBP5	3227	95.83	95.90	95.23	88.69	95.80	95.37	95.26	95.42	92.74	91.02
SIN3A	1796	90.81	91.96	92.63	91.48	92.59	92.52	91.53	92.62	92.12	90.91
TAF7	2072	91.35	91.16	90.37	86.90	90.86	90.31	90.45	90.60	88.95	87.57

Table C.4.: **Standard errors of AUC values in percentage points.**

TF	MM0	MM1	MM2	MM3	BIC2	BIC3	BIC4	fNML2	fNML3	fNML4
ATF2	1.87	1.01	0.97	0.97	0.73	1.55	0.70	0.43	1.04	1.08
ATF3	0.40	0.19	0.33	0.41	0.30	0.59	0.26	0.23	0.30	0.53
BACH1	0.15	0.08	0.30	0.23	0.31	0.28	0.25	0.12	0.31	0.56
BCL11A	1.21	1.50	1.49	1.20	1.79	0.32	0.83	0.95	2.13	1.37
BRCA1	0.43	0.53	0.28	0.27	0.39	0.27	0.65	0.12	0.39	0.42
CEBPB	0.13	0.17	0.19	0.21	0.18	0.33	0.93	0.27	0.32	0.65
CHD1	0.80	1.08	0.73	1.11	0.90	1.55	1.24	1.53	1.46	1.05
CHD2	0.43	0.24	0.14	0.28	0.44	0.77	0.42	0.16	0.06	0.30
CTBP2	0.61	0.18	0.46	0.65	0.25	1.04	0.83	0.88	0.52	0.89
CTCF	0.02	0.02	0.02	0.02	0.02	0.02	0.12	0.02	0.03	0.07
EGR1	0.20	0.26	0.16	0.35	0.21	0.23	0.53	0.21	0.52	0.37
EP300	0.96	0.64	0.67	0.65	0.52	0.98	0.56	0.86	0.47	0.64
EZH2	0.40	0.29	0.32	0.24	0.22	0.24	0.18	0.28	0.23	1.73
FOSL1	0.63	0.62	0.46	0.66	0.27	0.56	0.28	0.21	0.32	0.69
GABPA	0.14	0.06	0.09	0.26	0.09	0.04	0.16	0.03	0.09	0.17
GTF2F1	0.58	0.71	0.43	0.65	1.02	0.58	1.19	1.51	0.49	1.00
HDAC2	0.69	0.22	0.80	0.82	1.14	0.51	0.25	1.02	0.47	0.69
JUN	3.92	0.43	1.04	1.28	0.73	1.14	0.99	0.65	0.73	1.99
JUND	0.47	0.37	0.24	0.86	0.39	0.39	0.46	0.39	0.42	0.57
KDM5A	0.40	0.35	0.49	0.31	0.65	0.23	0.22	0.68	0.67	0.60
MAFK	0.20	0.15	0.21	0.23	0.16	0.30	0.13	0.22	0.15	0.73
MAX	0.31	0.47	0.32	1.33	0.29	0.34	0.09	0.37	0.42	0.72
MXI1	0.20	0.83	0.56	0.52	0.69	0.49	0.26	0.48	0.63	0.35
MYC	0.82	1.68	1.50	1.88	1.77	1.08	2.10	1.48	1.22	2.20
NANOG	1.29	1.40	1.65	0.64	2.13	1.85	1.59	1.29	1.20	1.71
NRF1	0.18	0.09	0.12	0.14	0.17	0.13	0.09	0.16	0.13	0.09
POLR2A	0.42	0.12	1.01	0.51	0.24	0.74	1.49	0.91	0.93	0.56
POU5F1	0.34	0.61	0.55	0.86	0.68	0.73	0.67	0.75	0.59	0.37
RAD21	0.05	0.02	0.03	0.05	0.03	0.03	0.05	0.01	0.05	0.08
RBBP5	0.22	0.21	0.56	0.30	0.27	0.24	0.38	0.23	0.78	1.13
REST	0.28	0.05	0.13	0.04	0.16	0.14	0.14	0.04	0.05	0.12
RFX5	0.63	0.20	0.66	0.83	0.29	0.40	0.27	0.32	0.87	1.65
RXRA	1.28	1.35	0.97	2.73	1.07	1.57	2.16	1.20	1.96	1.84
SIN3A	0.35	0.42	0.40	0.49	0.38	0.49	0.46	0.49	0.61	0.90
SIN3AK20	0.40	0.22	0.52	0.19	0.27	0.72	0.64	0.34	0.41	0.33
SIX5	0.51	0.39	0.19	0.47	0.58	0.42	0.23	0.53	0.12	0.62
SP1	0.51	0.47	0.36	0.38	0.81	0.42	0.35	0.46	0.38	0.51
SP2	0.58	0.42	0.29	0.54	0.21	0.30	0.68	0.43	0.48	0.38
SP4	0.50	0.29	0.33	1.41	0.39	0.43	0.46	0.57	0.54	0.50
SRF	0.28	0.16	0.30	0.31	0.52	0.17	0.41	0.51	0.23	0.53
SUZ12	0.51	0.34	0.52	0.27	0.32	0.35	0.49	0.61	0.71	0.47
TAF1	0.17	0.18	0.15	0.07	0.16	0.16	0.34	0.16	0.21	0.39
TAF7	0.47	0.38	0.24	1.16	0.37	0.54	0.82	0.33	0.91	0.62
TBP	0.10	0.23	0.47	0.86	0.29	0.53	0.37	0.30	0.80	0.96
TCF12	0.80	1.01	0.31	0.62	0.22	0.66	0.80	0.53	0.53	0.48
TEAD4	0.06	0.16	0.19	0.69	0.28	0.28	0.20	0.27	0.55	1.11
USF1	0.06	0.08	0.06	0.07	0.06	0.12	0.10	0.08	0.03	0.11
USF2	0.06	0.17	0.13	0.14	0.10	0.08	0.12	0.07	0.06	0.21
YY1	1.69	0.11	0.22	0.09	0.15	0.19	0.18	0.12	0.09	0.22
ZNF143	0.25	0.36	0.11	0.11	0.42	0.22	0.63	0.20	0.37	0.54

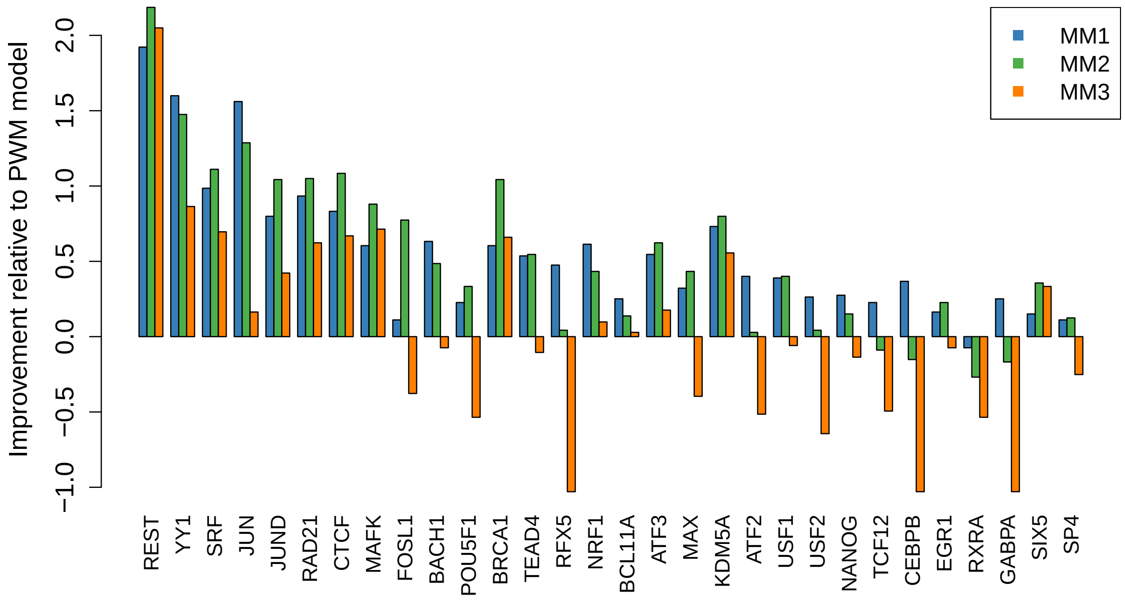


Figure C.3.: Ψ **value of modeling dependencies with fixed order Markov models of different order in relation to the PWM model.**

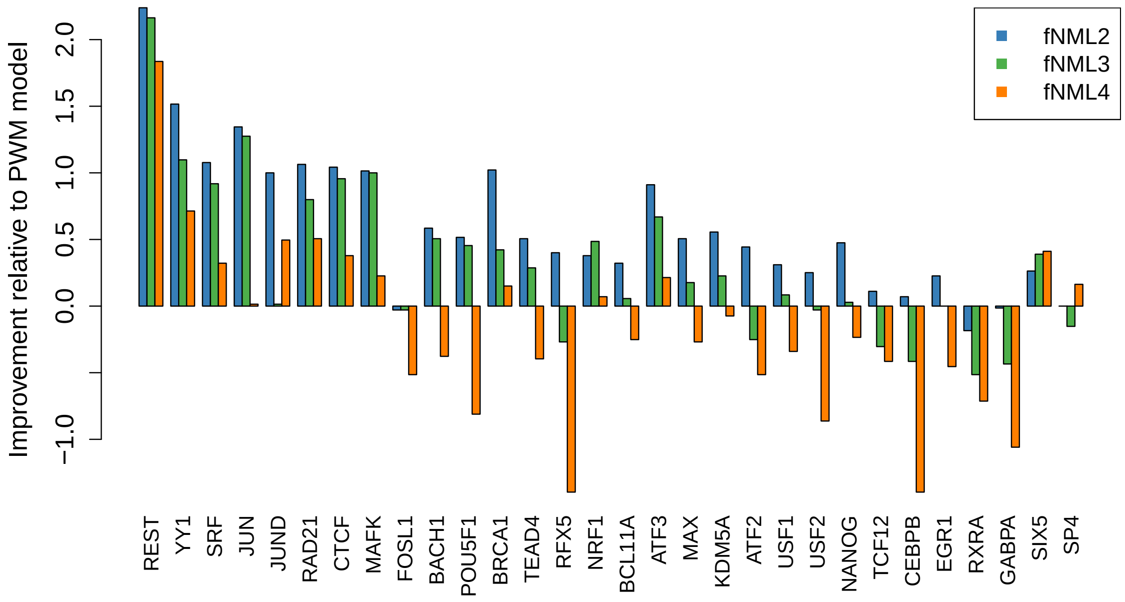


Figure C.4.: Ψ **value of modeling dependencies with PMMs of different order and fNML as model selection criterion in relation to the PWM model.**

Curriculum vitae

Personal data

Name	Ralf Eggeling
Gender	male
Nationality	German
Date of birth	14.02.1983
Place of birth	Magdeburg
Address	Leipziger Strasse 39 06108 Halle

Education

2010-present	Martin Luther University Halle-Wittenberg PhD studies on bioinformatics supervised by Prof. Dr. Ivo Grosse
2003-2010	Martin Luther University Halle-Wittenberg Diplom studies on bioinformatics
2002-2003	Civil service Gemeinde Niederndodeleben
1997-2002	Allertal-Gymnasium Eilsleben
1993-1997	Allertal-Gymnasium Völpke
1989-1993	Grundschule Dreileben

Eidesstattliche Erklärung / *Declaration under Oath*

Ich erkläre an Eides statt, dass ich die Arbeit selbstständig und ohne fremde Hilfe verfasst, keine anderen als die von mir angegebenen Quellen und Hilfsmittel benutzt und die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.
I declare under penalty of perjury that this thesis is my own work entirely and has been written without any help from other people. I used only the sources mentioned and included all the citations correctly both in word or content.

____________________________ ____

Datum / Date Unterschrift des Antragstellers / *Signature of the applicant*